Lecture Notes in Computer Science 8474

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Stefan Trausan-Matu
Kristy Elizabeth Boyer Martha Crosby
Kitty Panourgia (Eds.)

Intelligent Tutoring Systems

12th International Conference, ITS 2014
Honolulu, HI, USA, June 5-9, 2014
Proceedings

Springer

Volume Editors

Stefan Trausan-Matu
University Politehnica of Bucharest, Computer Science Department
Splaiul Independentei 313, Bucharest 060042, Romania
E-mail: stefan.trausan@cs.pub.ro

Kristy Elizabeth Boyer
North Carolina State University
890 Oval Dr., Campus Box 8206, Raleigh, NC 27695-8206, USA
E-mail: keboyer@ncsu.edu

Martha Crosby
University of Hawaii, Department of Information and Computer Sciences
1680 East-West Road, POST 317, Honolulu, HI 96822, USA
E-mail: crosby@hawaii.edu

Kitty Panourgia
Neoanalysis Ltd.
Marni 56, 10437 Athens, Greece
E-mail: kpanourgia@neoanalysis.eu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-319-07220-3 e-ISBN 978-3-319-07221-0
DOI 10.1007/978-3-319-07221-0
Springer Cham Heidelberg New York Dordrecht London

Library of Congress Control Number: 2014938384

LNCS Sublibrary: SL 2 – Programming and Software Engineering

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The 12th International Conference on Intelligent Tutoring Systems (ITS 2014) was held during June 5–9, 2014, in Honolulu, Hawaii, USA. This biennial conference focuses on research that investigates the use of intelligent systems and advanced computing technologies with close ties to interdisciplinary research for enabling, supporting, or enhancing human learning. A major direction of the series of ITS conferences is using Artificial Intelligence technologies for adapting systems to learners, modeling those learners, and providing the best-suited learning material based upon both the learner and the context. An important emphasis within the ITS community is on supporting interaction with adaptive systems as well as on the social construction of knowledge.

Reflecting the importance of this interactivity, the theme of the ITS 2014 conference was *Creating Fertile Soil for Learning Interactions*. Much as the volcanic islands of Hawaii have, over time, developed fertile ground that supports iconic biodiversity, the ITS research community is poised to see decades of rich ITS research come together to produce highly interactive systems that support a broad diversity of learner needs. With an emphasis not only on developing technologies to support learning, but on making fundamental discoveries regarding teaching and learning, ITS 2014 brought together researchers from computer science, learning sciences, cognitive and educational psychology, sociology, cognitive science, artificial intelligence, machine learning, and linguistics.

Submissions were received within three tracks. The Main Scientific Program Track was chaired by Stefan Trausan-Matu and Kristy Elizabeth Boyer; the Workshops and Tutorials Track was chaired by Min Chi and Roger Azevedo, and the Young Researchers' Track was chaired by Winslow Burleson and Tsukasa Hirashima. The Young Researchers' Track papers are included in this volume, while Workshop proceedings were prepared separately by the workshop chairs and distributed alongside the electronic proceedings at the conference.

The international response to the call for papers yielded 177 papers to the main scientific track from 28 different countries. Reviewing these submissions was a highly diverse Program Committee of 162 members. There were a minimum of three reviews per submission including at least one senior Program Committee member. Below is the number of authors and PC members from each country that participated in this year's conference by submitting their work or by reviewing papers.

ITS 2014 followed a triple-blind reviewing process: reviewers did not see author names, authors did not see reviewer names, and reviewers did not see each others' names during the review and discussion process. We worked to ensure a high quality and fair reviewing process, and we are tremendously grateful for the senior PC and regular PC members who contributed to reviewing.

Conflicts of interest were identified so that no paper was assigned to a reviewer from the same institution or who was a close collaborator of the papers' authors. The program chairs made the final decisions for acceptance on the basis of the reviews, discussions, and meta-reviews. When needed, the program chairs carefully read the papers and sought additional reviews to resolve inconsistencies.

Country	Authors	PC Members
Algeria	-	1
Australia	9	1
Austria	-	1
Brazil	51	10
Bulgaria	-	2
Canada	38	14
China	2	-
Colombia	1	-
Cyprus	2	-
Denmark	-	3
Egypt	2	-
France	18	12
Germany	10	8
Greece	3	1
India	5	-
Ireland	-	1
Italy	5	5
Japan	32	10
Korea	-	2

Country	Authors	PC Members
Lebanon	1	-
Mexico	1	2
Netherlands	4	3
New Zealand	2	3
Philippines	7	2
Poland	1	-
Portugal	2	1
Qatar	1	-
Romania	2	2
Saudi Arabia	2	1
Slovakia	-	1
Slovenia	7	-
Spain	7	3
Switzerland	4	2
Taiwan	5	4
Turkmenistan	-	-
United Kingdom	3	13
United States	286	58

Of the 177 submissions to the main scientific track, 31 were accepted as long papers (17.5%). Additionally, 45 submissions were accepted as short papers, representing high quality work that was deemed by the reviewers to be perhaps slightly less mature than the work accepted as long papers. Both long papers and short papers were presented as oral presentations at the conference. Finally, 42 submissions were accepted as posters which were presented as interactive poster exhibits at the conference. One special panel, "Grand Challenges for Intelligent Tutoring Systems in STEM: Progress and Perspectives" was organized by Xiangen Hu, Benjamin Nye, Art Graesser, Neil Heffernan, Kurt VanLehn, and Beverly Woolf.

Long papers were provided 10 pages, short papers 6 pages, and poster papers 2 pages in the proceedings. Authors could optionally purchase up to 2 additional pages for each paper, resulting in long papers occupying up to 12 pages, short papers up to 8 pages, and poster papers up to 4 pages in this volume.

The papers within the main scientific track span a range of topics, and have been organized into groups in these proceedings in a necessarily subjective way. The major topics reflect the sessions in which the conference presentations were

organized: affect and metacognition; ITS scaling and assessment; collaborative learning; dialogue and discourse; data mining and student behavior; graphical representations and learning; game-based learning and simulation; dynamic hints and scaffolds; student strategies and problem solving.

Topic	Sub-missions	Accepted	Acceptance Rate	PC Members
Privacy and security in e-learning environments	-	-	-	3
Recommender systems for learning	7	3	0.43	32
Co-adaptation between technologies and human learning	7	3	0.43	20
Informal learning environments, learning as a side effect of interactions	9	3	0.33	14
Multi-agent and service-oriented architectures for learning and tutoring environments	9	6	0.67	11
Ontological modeling, Semantic web technologies and standards for learning	10	6	0.60	26
Non conventional interactions between artificial intelligence and human learning	10	8	0.80	10
Ubiquitous and mobile learning environments	10	6	0.60	11
Virtual pedagogical agents and learning companions	18	11	0.61	28
Instructional design principles or design patterns for educational environments	19	15	0.79	13
Dialogue and discourse during learning interactions	21	15	0.71	26
Simulation-based learning and serious games	22	17	0.77	30
Authoring tools and development methodologies for advanced learning technologies	22	14	0.64	26
Collaborative and group learning, communities of practice and social networks	26	15	0.58	39
Empirical studies of learning with technologies, understanding human learning on the Web	29	22	0.76	32
Modeling of motivation, metacognition, and affect aspects of learning	30	21	0.70	37
Domain-specific learning technologies, e.g., language, mathematics, reading, science, medicine, military, and industry.	35	23	0.66	18
Educational exploitation of data mining and machine learning techniques	43	35	0.81	34
Adaptive support for learning, models of learners, diagnosis and feedback	55	38	0.69	54
Intelligent tutoring	92	68	0.74	62

We wish to thank of all the authors, the members of the Program Committee and the external reviewers, the Steering Committee and in particular Claude Frasson and Stefano Cerri for their advice and help, and the Organizing Committee. Such an event would not have been possible without their commitment, professional effort and patience. We also wish to thank the creators and maintainers of the Easychair online conference management system, without which the review process and proceedings creation would have been tremendously difficult. Easychair's reliable and expansive set of functionality was a great help.

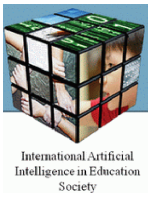
We hope that you enjoy these proceedings. It has been a great pleasure to serve the ITS research community by assembling them.

April 2014

Stefan Trausan-Matu
Kristy Elizabeth Boyer
Martha Crosby
Kitty Panourgia

With sincere thanks to the sponsors of the conference, including:

The Association for the Advancement of Artificial Intelligence (AAAI)



Conference Committee

Conference and General Chair

Martha Crosby	University of Hawaii, USA

Program Chairs

Stefan Trausan-Matu	University Politehnica of Bucharest and Research Institute for Artificial Intelligence, Romania
Kristy Elizabeth Boyer	North Carolina State University, USA

Organization Chair

Kitty Panourgia	Neoanalysis, Greece

Workshops and Tutorials Chairs

Min Chi	North Carolina State University, USA
Roger Azevedo	North Carolina State University, USA

Young Researchers' Track Chairs

Winslow Burleson	Arizona State University, USA
Tsukasa Hirashima	Hiroshima University, Japan

Award Chair

Kalina Yacef	University of Sydney, Australia

Organization Committee

Organization Chair

Kitty Panourgia — Neoanalysis, Greece

Local Organization Committee Chair

Michael-Brian Ogawa — University of Hawaii, USA

Members

Alexia Kakourou
Katerina Milathianaki
Eleni Vradi
Anna Mihail
Manolis Vasmanolis

Site Architect

Isaak Tselepis

Steering Committee

Chair

Claude Frasson	University of Montreal, Canada

Members

Stefano Cerri	LIRMM: University of Montpellier II and CNRS, France
Isabel Fernandez-Castro	University of the Basque Country, Spain
Gilles Gauthier	University of Quebec at Montreal, Canada
Mitsuru Ikeda	Japan Advanced Institute of Science and Technology, Japan
Marc Kaltenbach	Bishop's University, Canada
Judith Kay	University of Sidney, Australia
Alan Lesgold	University of Pittsburgh, USA
James Lester	North Carolina State University, USA
Roger Nkambou	University of Quebec at Montreal, Canada
Giorgos Papadourakis	Technological Educational Institute of Crete, Greece
Fabio Paragua	Federal University of Alagoas, Brazil
Elliot Soloway	University of Michigan, USA
Daniel Suthers	University of Hawaii, USA
Beverly Woolf	University of Massachussets, USA

Program Committee

Senior Program Committee

Esma Aimeur	University of Montreal, Canada
Vincent Aleven	Carnegie Mellon University, USA
Ivon Arroyo	University of Massachusetts Amherst, USA
Kevin Ashley	University of Pittsburgh, USA
Jacqueline Bordeau	TELU-UQAM, Canada
Bert Bredeweg	University of Amsterdam, Netherlands
Paul Brna	University of Leeds, UK
Peter Brusilovsky	University of Pittsburgh, USA
Stefano A. Cerri	LIRMM: University of Montpellier and CNRS, France
Tak-Wai Chan	National Central University, Taiwan
Albert Corbett	Carnegie Mellon University, USA
Elisabeth Delozanne	LIP6-Université Pierre et Marie Curie, France
Vania Dimitrova	University of Leeds, UK
Benedict Du Boulay	University of Sussex, UK
Isabel Fernandez-Castro	University of the Basque Country, Spain
Claude Frasson	University of Montreal, Canada
Art Graesser	University of Memphis, USA
Peter Hastings	DePaul University, USA
Neil Heffernan	Worcester Polytechnic Institute, USA
W. Lewis Johnson	Alelo Inc., USA
Kenneth Koedinger	Carnegie Mellon University, USA
Jean-Marc Labat	Université Paris 6, France
Susanne Lajoie	McGill University, Canada
H. Chad Lane	University of Southern California, USA
James Lester	North Carolina State University, USA
Diane Litman	University of Pittsburgh, USA
Rose Luckin	The London Knowledge Lab, UK
Gordon McCalla	University of Saskatchewan, Canada
Tanja Mitrovic	University of Canterbury in Christchurch, New Zealand
Riichiro Mizoguchi	Japan Advanced Institute of Science and Technology, Japan
Jack Mostow	Carnegie Mellon University, USA
Roger Nkambou	University of Quebec at Montreal, Canada
Toshio Okamoto	University of Electro-Communications, Japan
Ana Paiva	University of Porto, Portugal
Niels Pinkwart	Humboldt-Universität zu Berlin, Germany

Carolyn Rosé	Carnegie Mellon University, USA
Ryan S.J.D. Baker	Columbia University, USA
Kurt VanLehn	Arizona State University, USA
Julita Vassileva	University of Saskatchewan, Canada
Rosa Vicari	Universidade Federal do Rio Grande do Sul, Brazil
Gerhard Weber	University of Education Freiburg, Germany
Beverly Park Woolf	University of Massachusetts, USA
Kalina Yacef	The University of Sydney, Australia

Program Committee

Mohammed Abdel Razek	King Abdulaziz University, Saudi Arabia
Fabio Akhras	Renato Archer Center of Information Technology, Brazil
Colin Allison	University of St. Andrews, UK
Galia Angelova	Bulgarian Academy of Sciences, Bulgaria
Ana Arruarte	University of the Basque Country, Spain
Roger Azevedo	North Carolina State University, USA
Tiffany Barnes	North Carolina State University, USA
Beatriz Barros	University of Malaga, Spain
Maria Bielikova	Slovak University of Technology in Bratislava, Slovakia
Ig Ibert Bittencourt	Federal University of Alagoas, Brazil
Emmanuel G. Blanchard	Aalborg University at Copenhagen, Denmark
Stephen B. Blessing	University of Tampa, USA
Joost Breuker	University of Amsterdam, The Netherlands
Winslow Burleson	Arizona State University, USA
Nicola Capuano	University of Salerno, Italy
Luigia Carlucci Aiello	Sapienza Università di Roma, Italy
Chih-Kai Chang	National University of Tainan, Taiwan
Min Chi	Carnegie Mellon University, USA
Chih-Yueh Chou	Yuan Ze University, Taiwan
Mark Core	University of Southern California, USA
Evandro Costa	Federal University of Alagoas, Brazil
Scotty Craig	Arizona State University, USA
Alexandra Cristea	University of Warwick, UK
Sidney D'Mello	University of Notre Dame, Canada
Michel Desmarais	Ecole Polytechnique de Montreal, Canada
Cyrille Desmoulins	Université Joseph Fourrier, Grenoble, France
Philippe Dessus	LSE in Grenoble, France
Barbara Di Eugenio	University of Illinois at Chicago, USA
Darina Dicheva	Winston-Salem State University, USA
Peter Dolog	Aalborg University, Denmark
Robert Farrell	IBM T.J. Watson Research Center, USA
Mark Floryan	University of Massachussets Amherst, USA

Davide Fossati	Carnegie Mellon University in Qatar, Qatar
Nobuko Fujita	University of Toronto, Canada
Vasco Furtado	UNIFOR, Brazil
Stephen Gilbert	Iowa State University, USA
Ashok Goel	Georgia Institute of Technology, USA
Abdelkader Gouaïch	LIRMM, France
Yusuke Hayashi	Hiroshima University, Japan
Cecily Heiner	Southern Utah University, USA
Tsukasa Hirashima	Hiroshima University, Japan
Ulrich Hoppe	University Duisburg-Essen, Germany
Seiji Isotani	University of Sao Paulo, Brazil
Patricia Jaques	UNISINOS, Brazil
Heisawn Jeong	Hallym University, Republic of Korea
Clement Jonquet	University of Montpellier – LIRMM, France
Imène Jraidi	University of Montreal, Canada
Akihiro Kashihara	University of Electro-Communications, Japan
Kathy Kikis-Papadakis	IACM/FORTH, Greece
Nguyen-Thinh Le	Clausthal University of Technology, Germany
Philippe Lemoisson	CIRAD, France
Stefanie Lindstaedt	Graz University of Technology & Know-Center, Austria
Vincenzo Loia	University of Salerno, Italy
Vanda Luengo	Université Joseph Fourier Grenoble, France
Tatsunori Matsui	Waseda University, Japan
Manolis Mavrikis	London Knowledge Lab, UK
Riccardo Mazza	University of Lugano/University of Applied Sciences of Southern Switzerland, Switzerland
Alessandro Micarelli	Roma Tre University, Italy
Kazuhisa Miwa	Nagoya University, Japan
Paul Mulholland	The Open University, UK
Chas Murray	Carnegie Learning, Inc., USA
Wolfgang Nejdl	L3S and University of Hannover, Germany
Germana Nobrega	Universidade de Brasília, Brazil
Amy Ogan	Carnegie Mellon University, USA
Alexandra Poulovassilis	University of London, UK
Liana Razmerita	Copenhagen Business School, Denmark
Genaro Rebolledo-Mendez	University of Veracruz, Mexico
Ma. Mercedes T. Rodrigo	Ateneo de Manila University, Philippines
Ido Roll	University of British Columbia, Canada
John Stamper	Carnegie Mellon University, USA
Daniel Suthers	University of Hawaii, USA
Akira Takeuchi	Kyushu Institute of Technology, Japan
Pierre Tchounikine	University of Grenoble, France
Gheorghe Tecuci	George Mason University, USA

Thanassis Tiropanis	University of Southampton, UK
Wouter Van Joolingen	University of Twente, The Netherlands
Amali Weerasinghe	University of Canterbury, UK
Diego Zapata-Rivera	Educational Testing Service, USA
Ramon Zatarain	Instituto Tecnólgico de Culiacán, Mexico

Additional Reviewers

Adewoyin, Oluwabunmi
Alizadeh, Mehrdad
Alvarez, Ainhoa
Atapattu, Thushari
Biancalana, Claudio
Bredeweg, Bert
Buffum, Philip
Chavez-Echeagaray, Maria-Elena
Chien, Tzu-Chao
Chiru, Costin-Gabriel
Chuang, Chiayuan
Conejo, Ricardo
Dascalu, Mihai
De Maio, Carmen
Demmans Epp, Carrie
Diad de Ilarraza, Arantza
Elorriaga, Jon A.
Falakmasir, Mohammad Hassan
Ferrero, Bego
Foss, Jonathan
Furtado, Elizabeth
Gaeta, Angelo
Gonzalez-Sanchez, Javier
Grafsgaard, Joseph
Grawemeyer, Beate
Green, Nick
Harsley, Rachel
Hayashi, Yugo
Hayashi, Yusuke
Hidalgo-Pontet, Yoalli
Horiguchi, Tomoya
Hosseini, Roya
Hsu, Shihhsun
Kojima, Kazuaki
Kompan, Michal
Larrañaga, Mikel
Limongelli, Carla
Longhi Rossi, Luiz Henrique
Lynch, Collin
Manske, Sven
Maritxalar, Montse
Matsuda, Noriyuki
Mayorga, José Ignacio
Miranda, Sergio
Mitchell, Christopher
Mitrovic, Tanja
Morita, Jyunya
Ogata, Hiroaki
Orciuoli, Francesco
Ostrow, Korinn
Pierri, Anna
Pinheiro, Vladia
Primo, Tiago
Randolph, David
Rebedea, Traian
Ritrovato, Pierluigi
Rodríguez, Fernando
Sansonetti, Giuseppe
Seaton, Jennifer
Selmi, Mouna
Shi, Lei
Shubeck, Keith
Stampfer, Eliane
Stepanyan, Karen
Trella, Monica
Tvarozek, Jozef
Urretavizcaya, Maite
Vasconcelos, Eurico
Walker, Erin
Wylie, Ruth
Xiong, Xiaolu
Yeh, Yen-Cheng

Table of Contents

Affect

Multimodality and Metacognition

Collaborative Learning

Data Mining and Student Behavior

Dialogue and Discourse

Generating Hints, Scaffolds, and Questions

Game-Based Learning and Simulation

Graphical Representations and Learning

Student Strategies and Problem Solving

Scaling ITS and Assessment

Posters

Young Researchers' Track

Sensor-Free Affect Detection for a Simulation-Based Science Inquiry Learning Environment

Luc Paquette[1], Ryan S.J.D. Baker[1,2], Michael A. Sao Pedro[2], Janice D. Gobert[2], Lisa Rossi[3], Adam Nakama[2], and Zakkai Kauffman-Rogoff[2]

[1] Teachers College, Columbia University, New York, NY
[2] Worcester Polytechnic Institute, Worcester, MA
[3] Georgia Institute of Technology, Atlanta, GA
paquette@tc.columbia.edu, baker2@exchange.tc.columbia.edu,
{mikesp,jgobert,nakama}@wpi.edu, lrossi@gatech.edu,
zakkai@gmail.com

Abstract. Recently, there has been considerable interest in understanding the relationship between student affect and cognition. This research is facilitated by the advent of automated sensor-free detectors that have been designed to "infer" affect from the logs of student interactions within a learning environment. Such detectors allow for fine-grained analysis of the impact of different affective states on a range of learning outcome measures. However, these detectors have to date only been developed for a subset of online learning environments, including problem-solving tutors, dialogue tutors, and narrative-based virtual environments. In this paper, we extend sensor-free affect detection to a science microworld environment, affording the possibility of more deeply studying and responding to student affect in this type of learning environment.

Keywords: Educational data mining, affect detection, affective computing.

1 Introduction

It is well recognized that affect interacts with engagement and learning in complex ways [1, 2, 3, 4, 5, 6]. Learning software such as ITSs offer great opportunities to study those interactions due to their fine-grained interaction logs and their capacity to track students' actions at multiple levels. In recent years, this research has been facilitated by the use of sensor-free affect detectors that can automatically infer a range of student affective states from student interactions. Sensor-free detectors have been developed for three kinds of ITSs to date: problem-solving ITSs where answers are straightforward (e.g. [7, 8, 9]), dialogue tutors where the student iterates towards an answer (e.g. [10, 11]), and narrative-based virtual environments where the student explores a complex environment (e.g. [12]). One key finding is that, though the principles of affect detection are largely the same, the student behaviors associated with each affect often differ considerably based on the design of the learning environment being used. For instance, affect detection in problem-solving tutors tends to

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 1–10, 2014.

focus on timing, pauses, and patterns of errors, and the contexts in which they occur. In game-like virtual environments such as Crystal Island, affect detectors have been built using counts of how many times the player engaged in meaningful actions such as viewing books, and whether the student has completed important milestones [12]. In dialogue tutors, affect detection tends to focus on the actual content of student dialogue acts and how the content changes over time. Given this coupling between student behaviors indicative of affective states and the learning environment in which they are demonstrated, it is important to study those behaviors in a broader range of learning environments to make sensor-free affect detection more feasible.

In this paper, we study how to automatically detect student affect in the Inq-ITS inquiry learning environment [13] in which students use simulation and support tools to engage in inquiry. We do this by using a combination of data mining and ground-truth labels that were obtained from field observations of affect. When compared with other systems, Inq-ITS's simulation microworlds offer a less constrained learning environment than problem-solving [7, 8, 9] or dialogue tutors [10, 11], allowing more exploratory behaviors. At the same time, simulation microworlds are more constrained than virtual environments, such as Crystal Island [12] and EcoMUVE [14], where students have a lot of freedom to explore the virtual world which can lead to a wider range of ways that affect can manifest in behaviors.

Prior research on affect in simulation microworlds has provided evidence of a range of different affective states associated with learning. For example, relatively high amounts of boredom, an undesirable affect associated with both gaming the system [1] and off-task behavior [15], has been observed in some simulation microworlds [15]. The availability of sensor-free affect detectors for this type of environment would enable more in-depth studies of similar relationships, providing a better understanding of how affect impacts learning in these rich learning contexts.

2 Inq-ITS Learning Environment

The Inq-ITS learning environment (formerly known as Science Assistments [13]) is a web-based environment in which students conduct inquiry with interactive simulations aligned to middle school Physical, Life, and Earth Science content described in the NGSS standards [16]. Activities have a driving question pertinent to a science topic, and require students to address the question by conducting an investigation using a simulation and other inquiry support tools.

For example, a driving question in a Phase Change activity asks students to determine if one of three factors (size of a container, amount of ice to melt, and amount of heat applied to the ice) affects various measurable outcomes (e.g., melting or boiling point). Students address this by conducting inquiry, i.e., formulating a hypothesis, collecting data to test it with the simulation, analyzing the data, warranting their claims, and communicating their findings. Before making a hypothesis, students can first explore the simulation. More information about Inq-ITS can be found in [13, 17].

3 Method

3.1 Data Collection

Data on student affect was collected from 326 students who conducted inquiry within the Inq-ITS system in 2011 in 11 different 8th grade classes from 3 schools in Massachusetts. Students came from a diverse population (Table 1).

Table 1. Demographic information for the three schools in our data set

	First school	Second school	Third school	State average
Hispanic students	3%	6%	40%	10%
African-American students	0%	2%	17%	8%
Asian-American students	3%	12%	12%	6%
Caucasian Students	89%	79%	28%	76%
Students at or above proficient level on the MCAS science test	53%	63%	10%	39%
Students receiving reduced or free lunch	5%	16%	83%	34%

Four expert field observers coded student affect and engaged/disengaged behaviors while students used the software. Here, we focus on the affect codes. The observers based their judgment of a student's affect on the student's work context, actions, utterances, facial expressions, body language, and interactions with teachers or fellow students [cf. 18, 19]. Within an observation, each observer coded affect on five categories [1]: boredom, confusion, frustration, engaged concentration (the affect associated with the flow state [cf. 1]) and "?" (an affect different from the coding scheme and situations when coding was impossible/irrelevant such as when a student went to the bathroom).

The coders used the HART app for Google Android handheld computers [8], which implements the Baker-Rodrigo Observation Method Protocol (BROMP) [1, 20], a protocol for coding affect and behavior during use of educational software. All coding was conducted by the second, fifth, sixth, and seventh authors. These coders were previously trained by two expert coders. Pairs of coders achieved inter-rater reliability (Kappa) of 0.72 (second and sixth authors, affect), 0.60 (second and seventh, affect) and 0.60 (fifth author and additional expert coder, affect). This degree of reliability is on par with Kappas reported by past projects that have assessed the reliability of detecting naturally occurring emotional expressions [1, 18, 21, 22].

As mandated in BROMP [20], students were coded in a pre-chosen order, with each observation focusing on a specific student. To obtain the most representative indication possible of student affect, only the current student's affect was coded. At the beginning of each class, an ordering of observation was chosen based on the class layout and was enforced using the hand-held observation software. A total of 4155 observations were made across the 326 students. Each observation lasted up to twenty seconds, with observation time automatically coded by the handheld software. If affect and behavior were determined before twenty seconds elapsed, the coder moved to the next observation. If two distinct affective states occurred during a single

observation, only the first state observed was coded. Each observation was conducted using peripheral vision or side-glances to reduce disruption [cf. 1, 20, 22, 23].

From the initial 4155 observations, 1214 (from 205 students, with an average of 5.92 observations per students and a standard deviation of 5.94) were used in the final analyses. Of the 2941 discarded observations: 1146 were coded as "?"; 331 were made while the student had been inactive for more than 5 minutes; and 1464 were made when the student was not currently involved in a science inquiry task (for example, when the student was answering other multiple-choice test questions [e.g. 24]). Within the 1214 remaining observations, the affective states had the following frequencies: engaged concentration was observed 896 times (82.50%), boredom 109 times (10.03%), confusion 44 times (4.05%), and frustration 38 times (3.50%).

3.2 Feature Distillation

In order to distill a feature set for our affect detectors, student actions within the software were synchronized to the field observations. During data collections, both the handheld computers and the Inq-ITS server were synchronized to the same internet NTP time server. Actions during the 20 seconds prior to data entry by the observer were considered as co-occurring with the observation. A total of 127 features were developed using the actions that co-occurred with or preceded the observation.

Our main feature set was based on the 73 features distilled by Sao Pedro et al. in [24], which looked at the different types of actions the students can make while they use Inq-ITS. Of the action types distilled in [24], we kept those that occurred in our data set: hypothesis variable changes, simulation variable changes, simulation pauses, incomplete trials run, complete trials run, all trials run and all relevant actions. We note that Sao Pedro et al. [24] did not include student interactions during the analysis stage of the inquiry process. We included analysis stage interactions to enable affect detection in that stage and created 7 new features to summarize those interactions.

To compute values for the previously described features, we accumulated lists of each type of relevant action during the 60 seconds prior to an observation to capture the student's behavior immediately before it. For each of those lists, like [24], we calculated the minimum, maximum, average, median, standard deviations, and sum of the time spent on each action, as well as a count of the number of actions in the list. Since some observations were made when the student had been inactive for more than 60 seconds, we repeated the same process to create a second set of features using lists from the 5 actions prior to the observation. This combination accounted for 112 of the features distilled from our dataset.

We created two features related to the time elapsed since the last student action: a binary feature indicating whether the student has been inactive for the last 60 seconds, a potential indicator of off-task behavior [cf. 8], and the time elapsed between the last action of the student and the moment of the observation.

Bayesian knowledge tracing (BKT) was used to distill features indicating whether students knew how to apply two inquiry skills, designing controlled experiments and testing stated hypothesis [24]. Three features were computed for each skill: the probability that the skill was known before the most recent practice opportunity, the probability

the skill was known afterwards, and the probability that the student would correctly apply the skill on the most recent practice opportunity. In addition, we computed the ratio of positive and negative assessments during the last 5 student actions.

An additional 3 features were distilled in relation to the different stages of the inquiry process: whether the student had explored the microworld before making a hypothesis, whether the student had completed the current stage at least once in a past activity and the time elapsed so far during the current inquiry stage.

Finally, in the version of Inq-ITS (Science Assistments) for which the interaction data were collected, each time a student enters a stage for the first time for the current activity, the system shows a text box containing orienting instructions for each stage of inquiry. We created three features related to this text box: whether it is currently open, the time elapsed since it was opened (if it is still opened), and whether the student closed it during the 20-seconds of actions co-occurring with the observation.

3.3 Machine Learning Algorithms

We built separate detectors for four affective states: boredom, confusion, frustration, and engaged concentration for three stages of inquiry: hypothesizing, collecting data, and analyzing data. Thus, each affective state was predicted separately – e.g. BORED was distinguished from NOT BORED (i.e., all other affective states) within each inquiry stage (i.e., BORED/NOT BORED in hypothesizing, BORED/NOT BORED while collecting data, etc.). Separate detectors were created for each stage because they each have specific actions associated with its user interface. As such, the patterns of actions related to each affective state may differ between stages. For the specific case of engaged concentration, cases where students were off-task were considered NOT ENG. CONC., since this reflects engaged concentration with something other than learning or Inq-ITS (e.g. the day's classroom gossip). Also, no detectors were built for the "exploring" stage due to the low number of observations (only 23). Table 2 shows the frequency of each affective state.

Each detector was evaluated using leave-one-out student-level cross-validation. In this process, for each student, a detector is built using data from every other student before being tested on that student. By cross-validating at this level, we increase confidence that detectors we build with a specific feature set will be accurate for new students. In addition, re-sampling was used to make the class frequency more equal for detector development (e.g. 96.15% of the observations were labeled as "not frustrated" during hypothesizing). However, all performance calculations were made with reference to the original dataset, as in [12].

Table 2. Frequency of the affect observation across the four stages of inquiry

	Hypothesizing	Experimenting	Analyzing
BORED	35 (11.22%)	43 (8.14%)	28 (7.98%)
CONFUSED	13 (4.17%)	19 (3.60%)	10 (2.85%)
FRUSTRATED	12 (3.85%)	13 (2.46%)	10 (2.85%)
ENG. CONC.	220 (70.51%)	390 (73.86%)	271 (77.21%)

We fit sensor-free affect detectors using three common classification algorithms that have been successful for building affect detectors in the past [8, 9]: J48 decision trees, JRip, and step regression (linear regression with a step function). By fitting the detectors using multiple algorithms, we can select the best algorithm for each affective state, as manifested in the relationship between the distilled features and the affect labels (linear, small clusters, etc.). Detector performance was assessed using two metrics: Cohen's Kappa [25] and A' computed as the Wilcoxon statistic [26]. Cohen's Kappa assesses the degree to which the detector is better than chance at identifying the student's affective state for a specific observation. A Kappa of 0 indicates that the detector performs at chance, and a Kappa of 1 indicates that the detector performs perfectly. A' is the probability that the algorithm will correctly identify whether an observation is an example of a specific affective state. A' is equivalent to the area under the ROC curve in signal detection theory, and is approximated by W [26]. A model with an A' of 0.5 performs at chance, and a model with an A' of 1.0 performs perfectly. A' was computed at the observation-level.

Feature selection for machine learning was conducted using two semi-automated procedures. First, we applied forward selection, a process in which the feature that most improves model performance is added repeatedly until adding additional features no longer improves performance. During forward selection, cross-validated Kappa and A' on the original non-resampled dataset were used. Kappa was used as the main performance metric for selecting a feature, but an alternate feature was selected when the model's A' was judged to be unusually low when compared to the value of Kappa. Then, backward elimination was applied on the sets of features generated by the forward selection algorithm to determine whether a simpler model could achieve better or equivalent performance, thereby reducing model over-fitting.

4 Results

We evaluate the degree to which the detectors for each construct within each inquiry stage can identify their respective affect. Detectors' performance over all four constructs and across all inquiry tasks was better than chance (A' = .50, Kappa = 0.0) and comparably well to past sensor-free detectors of affect. Table 3 shows the performance of the 12 detectors we built and provides a list of the features used in each detector. Descriptions of each feature (from F1 to F47) are provided in Table 4. The average student cross-validated Kappa was 0.354 and the average A' was 0.720. This is above the average Kappa of 0.296 and A' of 0.682 obtained in a study with similar validation [9] within the ASSISTments problem-solving ITS for math. The detectors described in [12] for a virtual environment achieved an average accuracy that was 16% better than the base rate (approximately comparable to a Kappa of 0.16). The detectors for Cognitive Tutor Algebra from [8] achieved an average Kappa of 0.30.

Another positive aspect of our detectors is that they were cross-validated at the student-level, and developed using a diverse population (Table 1). As such, it is likely that they will generalize to new students across the entire population of Inq-ITS users.

Table 3. Each of the models and their student-level cross-validated performances

	Hypothesizing	Experimenting	Analyzing
BORED	J48 F2, F7, F31, F38 Kappa = 0.305 A' = 0.699	JRip F3, F15, F19, F35, F36 Kappa = 0.252 A' = 0.704	J48 F1, F10, F22, F37, F45, F47 Kappa = 0.438 A' = 0.767
CONFUSED	J48 F2, F14, F31, F45 Kappa = 0.327 A' = 0.704	JRip F2, F3, F9, F16, F20 Kappa = 0.355 A' = 0.777	J48 F20, F28, F33, F38 Kappa = 0.319 A' = 0.724
FRUSTRATED	JRip F2, F5, F31, F42, F44, F46 Kappa = 0.301 A' = 0.688	J48 F8, F11, F18, F30, F34, F39, F46 Kappa = 0.486 A' = 0.762	J48 F13, F23, F24, F26, F30, F32 Kappa = 0.379 A' = 0.729
CONCENTRATED	Step regression F3, F4, F6, F12, F29, F36, F43 Kappa = 0.336 A' = 0.715	J48 F17, F21, F27, F38, F41 Kappa = 0.313 A' = 0.638	Step regression F17, F23, F25, F34, F40 Kappa = 0.431 A' = 0.738

Table 4. List of all the features used in the final detectors

F1: The number of hypothesis variables changed in the last 60 seconds.
F2: The mean of all time taken to change one of the hypothesis variable in the last 60 seconds.
F3: The sum of all time taken to change one of the hypothesis variable in the last 60 seconds.
F4: The number of hypothesis variable changed in the last 5 student actions.
F5: The maximum of all time taken to change one of the hypothesis variable in the last 5 student actions.
F6: The median of all time taken to change one of the hypothesis variable in the last 5 student actions.
F7: The standard deviation of all time taken for hypothesis variable changes in the last 5 student actions.
F8: The minimum of all time taken to change one of the simulation variable in the last 60 seconds.
F9: The maximum of all time taken to change one of the simulation variable in the last 60 seconds.
F10: The median of all the time taken to change the value of a simulation variable in the last 60 seconds.
F11: The mean of all time taken to change one of the simulation variable in the last 60 seconds.
F12: The sum of all time taken changing a simulation variable in the last 60 seconds.
F13: The mean of all time taken to change one of the simulation variable in the last 5 student actions.
F14: The sum of all the time spent on completed trials run in the last 60 seconds.
F15: The minimum of all the time taken executing an incomplete trial in the last 60 seconds.
F16: The number of incomplete trials run in the last 5 student actions.
F17: The sum of all time spent executing trials in the last 60 seconds.
F18: The maximum of all time spent executing a trial in the last 5 student actions.
F19: The sum of all time taken executing trials in the last 5 student actions.
F20: The number of simulation pauses in the last 5 student actions.
F21: The mean of all time spent on simulation pauses in the last 5 student actions.
F22: The mean of all the time taken to execute one of the analysis action in the last 60 seconds.
F23: The sum of all time taken to execute any analysis action in the last 60 seconds.
F24: The number of analysis actions amongst the last 5 student actions.
F25: The mean of all time taken to execute any analysis action in the last 5 student actions.
F26: The standard deviation of all time taken to execute any analysis action in the last 5 student actions.
F27: The number of relevant actions executed in the last 60 seconds.
F28: The minimum of all time taken to execute any relevant action in the last 60 seconds.
F29: The median of all time taken to execute any relevant action in the last 60 seconds.
F30: The standard deviation of all time taken to execute any relevant action in the last 60 seconds.
F31: The sum of all the time taken to execute any relevant action in the last 60 seconds.

F32: The number of relevant actions amongst the last 5 student actions.
F33: The median of all time taken to execute any relevant action in the last 5 student actions.
F34: The standard deviation of all time taken to execute any relevant action in the last 5 student actions.
F35: The probability of knowing how to design controlled exp. before the most recent practice opportunity.
F36: The probability of knowing how to design controlled exp. after the most recent practice opportunity.
F37: The probability of correctly designing a controlled exp. on the most recent practice opportunity.
F38: The probability of knowing how to test stated hypothesis before the most recent practice opportunity.
F39: Whether the student was inactive in the software for the last 60 seconds.
F40: The time elapsed since the last user action at the moment of the observation.
F41: Whether the student entered the exploration stage during this activity.
F42: Whether the student has completed the current stage at least once in a previous activity.
F43: The time elapsed since the start of the current stage.
F44: Whether the text box is currently opened.
F45: The time elapsed since the explanation text box was opened, if it is still opened.
F46: Whether the student closed the text box during the observation.
F47: The ratio of positive and negative assessments by the system for the last 5 student actions.

5 Discussion and Conclusion

In this paper, we presented 12 sensor-free detectors that detect boredom, confusion, frustration, and engaged concentration in the different stages of inquiry in the Inq-ITS environment [17]. This work represents the first automated sensor-free detectors of student affect in simulation microworlds built. Conducting affect detection in a simulation microworld such as Inq-ITS presents different challenges than in other online learning environments. The absence of action-by-action assessment of correctness as in problem-based tutors (e.g. [8]) and the lack of on-demand help (e.g. [8, 9, 10]) hinder the engineering of features similar to those that have proven effective in problem-solving tutors and dialogue tutors such as Cognitive Tutor [8], ASSISTments [9] and AutoTutor [10]. However, other features such as the time spent on different types of actions, the probability that the student knew two key skills [24], and whether the student was inactive in the last 60 seconds, proved useful for this challenge (Table 4).

The non-uniform user interface for the different stages of inquiry also proved to be an important consideration for the generation of affect detectors. Each stage has specific types of actions associated with it and thus patterns of actions related to each affect differ in each stage. This is a general problem for affect detection in learning environments where the student-computer interaction can change considerably from moment to moment. An additional challenge comes from having many observations that co-occur with actions from two stages. In those situations, the interpretation, as an indicator of a specific affect, might differ for the same type of actions depending on whether the action occurred shortly before changing stages or right after changing stage. For these reasons, we created different detectors for each stage of the inquiry process in Inq-ITS. As can be seen in Table 3, few of the best features for individual detectors were reused across multiple stages for the same affect. No features were reused across the BORED detectors, F2 and F20 were reused for CONFUSED, F30 and F46 for FRUSTRATED, and F17 for CONCENTRATED.

The detectors proposed in this paper can be used to study whether specific features of the Inq-ITS system have an impact on the occurrence of affective states. For example, a

brief analysis indicates that in our dataset (collected on a prior version of Inq-ITS), 23 out of the 38 observations of frustration (60.53%) occurred when a text box was open or shortly after it was closed. This is more than one would expect as only 36.90% of all the observations matched this condition, and this feature has subsequently been changed in Inq-ITS.

By developing automated detectors that can identify boredom, confusion, frustration, and engaged concentration, we can take a step towards allowing Inq-ITS to effectively adapt to the full range of student's interaction choices during learning and develop interventions that target very specific kinds of disengaged behaviors, as has been successfully done to improve learning in other systems [as in 27, 28, and 29] to offset negative affect states such as boredom.

Acknowledgments. This research was supported by National Science Foundation grant DRL #1008649 awarded to Janice Gobert & Ryan Baker. We thank Michael Wixon, Ermal Toto, and Francis McGeever for their help in pre-processing the data and Jaclyn Ocumpaugh for BROMP training.

References

1. Baker, R.S.J.D., D'Mello, S.K., Rodrigo, M.M.T., Graesser, A.C.: Better to Be Frustrated than Bored: The Incidence, Persistence, and Impact of Learners' Cognitive-Affective States During Interactions with Three Different Computer-Based Learning Environments. International Journal of Human-Computer Studies 68(4), 223–241 (2010)
2. Baker, R.S.J.d., Moore, G.R., Wagner, A.Z., Kalka, J., Salvi, A., Karabinos, M., Ashe, C.A., Yaron, D.: The Dynamics Between Student Affect and Behavior Occurring Outside of Educational Software. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, pp. 14–24. Springer, Heidelberg (2011)
3. D'Mello, S.K., Taylor, R., Grasser, A.C.: Monitoring Affective Trajectories During Complex Learning. In: Proceedings of the 29th Annual Cognitive Science Society, pp. 203–208 (2007)
4. Dragon, T., Arroyo, I., Woolf, B.P., Burleson, W., el Kaliouby, R., Eydgahi, H.: Viewing Student Affect and Learning Through Classroom Observation and Physical Sensors. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 29–39. Springer, Heidelberg (2008)
5. Lee, D.M.C., Rodrigo, M. M.T., Baker, R.S.J.d., Sugay, J.O., Coronel, A.: Exploring the Relationship Between Novice Programmer Confusion and Achievement. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, pp. 175–184. Springer, Heidelberg (2011)
6. Sabourin, J., Rowe, J.P., Mott, B.W., Lester, J.C.: When Off-Task in On-Task: The Affective Role of Off-Task Behavior in Narrative-Centered Learning Environments. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 534–536. Springer, Heidelberg (2011)
7. Conati, C., Maclaren, H.: Empirically Building and Evaluating a Probabilistic Model of User Affect. UMUAI 19, 267–303 (2009)
8. Baker, R.S.J.d., et al.: Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. In: Proceedings of EDM 2012, pp. 126–133 (2012)
9. Pardos, Z., Baker, R.S.J.d., San Pedro, M.O.Z., Gowda, S.M., Gowda, S.: Affective States and State Tests: Investigating how Affect Throughout the School Year Predicts End of Year Learning Outcomes. In: Proceedings of LAK 2013, pp. 117–124 (2013)
10. D'Mello, S.K., Craig, S.D., Witherspoon, A.W., McDaniel, B.T., Graesser, A.C.: Automatic Detection of Learner's Affect from Conversational Cues. UMUAI 18, 45–80 (2008)

11. Litman, D.J., Forbes-Riley, K.: Recognizing Student Emotions and Attitudes on the Basis of Utterances in Spoken Tutoring Dialogue with Both Humans and Computer-Tutors. Speech Communication 48(5), 559–590 (2006)
12. Sabourin, J., Mott, B., Lester, J.: Modeling Learner Affect with Theoretically Grounded Dynamic Bayesian Networks. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, pp. 286–295. Springer, Heidelberg (2011)
13. Gobert, J., Sao Pedro, M., Baker, R., Toto, E., Montalvo, O.: Leveraging Educational Data Mining for Real Time Performance Assessment of Scientific Inquiry Skills within Microworlds. JEDM 4(1), 111–143 (2012)
14. Metcalf, S.J., Kamarainen, A., Grotzer, T.A., Dede, C.J.: Ecosystem Science Learning via Multi-User Virtual Environments. In: AERA Conference (2011)
15. Hershkovitz, A., Baker, R.S.J.d., Gobert, J., Nakama, A.: A Data-Driven Path Model of Student Attributes, Affect, and Engagement in a Computer-Based Science Inquiry Microworld. In: Proceedings of the ICLS (2012)
16. NGSS Lead States: Next Generation Science Standards: For States, By States.The National Academies Press, Washington (2013)
17. Sao Pedro, M., Baker, R., Gobert, J., Montalvo, O., Nakama, A.: Leveraging Machine-Learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skill. UMUAI 23, 1–39 (2013)
18. Bartel, C.A., Saavedra, R.: The Collective Construction of Work Group Moods. Administrative Science Quarterly 45, 197–231 (2001)
19. Planalp, S., DeFrancisco, V.L., Rutherford, D.: Varieties of Cues to Emotion in Naturally Occurring Situations. Cognition and Emotion 10(2), 137–153 (1996)
20. Ocumpaugh, J., Baker, R.S.J.d., Rodrigo, M.M.T.: Baker-Rodrigo Observation Method Protocol (BROMP) 1.0 Training Manual version 1.0. Technical Report, New York, NY: EdLab, Manila, Philippines: Ateneo Laboratory for the Learning Sciences (2012)
21. Litman, D.J., Forbes-Riley, L.: Recognizing Student Emotions and Attitudes on the Basis of Utterances in Spoken Tutoring Dialogues with Both Human and Computer Tutors. Speech Communication 48(5), 559–590 (2006)
22. Rodrigo, M.M.T., et al.: Comparing Learners' Affect While Using an Intelligent Tutoring Systems and a Simulation Problem Solving Game. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 40–49. Springer, Heidelberg (2008)
23. Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., Wagner, A.Z.: Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System". In: Proceedings of ACM CHI 2004: Computer-Human Interaction, pp. 383–390 (2004)
24. Sao Pedro, M., Baker, R., Gobert, J., Montalvo, O., Nakama, A.: Levaraging Machine-Learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skill. User Modeling and User-Adapted Interaction 23, 1–39 (2013)
25. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement 20(1), 37–46 (1960)
26. Hanley, J., McNeil, B.: The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve. Radiology 143, 29–36 (1982)
27. Woolf, B.P., Arroyo, I., Cooper, D., Burleson, W., Muldner, K.: Affective Tutors: Automatic Detection of and Response to Student Emotion. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) Advances in Intelligent Tutoring Systems. SCI, vol. 308, pp. 207–227. Springer, Heidelberg (2010)
28. Lehman, B.A., et al.: Inducing and Tracking Confusion with Contradictions During Complex Learning. IJAIED 22(2), 85–105 (2013)
29. Rai, D., Arroyo, I., Stephens, L., Lozano, C., Burleson, W., Woolf, B.P., Beck, J.E.: Repairing Deactivating Negative Emotions with Student Progress Pages. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 795–798. Springer, Heidelberg (2013)

Towards Automatically Detecting Whether Student Is in Flow

Po-Ming Lee[1,3], Sin-Yu Jheng[2], and Tzu-Chien Hsiao[2,3,4,*]

[1] Institute of Computer Science and Engineering, National Chiao Tung University, Taiwan (R.O.C.)
[2] Institute of Biomedical Engineering, National Chiao Tung University, Taiwan (R.O.C.)
[3] Department of Computer Science, National Chiao Tung University, Taiwan (R.O.C.)
[4] Biomedical Electronics Translational Research Center and Biomimetic Systems Research Center, National Chiao Tung University, Taiwan (R.O.C.)
labview@cs.nctu.edu.tw

Abstract. Csikszentmihalyi's flow theory states the components (e.g., balance between skill and challenge) that lead to an optimal state (referred to as flow state, or under flow experience) of intrinsic motivation and personal experience. Recent research has begun to validate the claims stated by the theory and extend the provided statements to the design of pedagogical interactions. To incorporate the theory in a design, automatic detector of flow is required. However, little attention has been drawn to this filed, and the detection of flow is currently still dominated by using surveys. Hence, within this paper, we present an automated detector which is able to identify the students that are in flow. This detector is developed using a step regression approach, with data collected from college students learning linear algebra from a step-based tutoring system.

Keywords: Student Modeling, Flow Theory, Educational Data Mining, Intelligent Tutoring System.

1 Introduction

Personal experiences are essential to pedagogical interactions. Hence, to improve personal experience, many studies have strived to increase the sensitivity and responsiveness of intelligent tutoring systems (ITSs) to various affects of students. On the other hand, Csíkszentmihályi's flow theory states the components that may lead to an optimal state (referred to as flow state, or under flow experience) of intrinsic motivation and personal experience [7]. When the flow theory is applied to education, numerous empirical studies on teaching, including teaching in high school classrooms by using traditional approaches (i.e. not ITS) [18, 19] and also teaching by using tutoring systems (TSs) [10, 14, 16], have reported that students engage in learning activities the most when they perceive both challenges and their skills as high.

The learning contents provided to students should be perceived as challenging yet not too difficult, for ensuring an optimal experience [18]. But in practice, learning

* Corresponding Author.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 11–18, 2014.

contents are usually non-adaptive, which are likely to fail in producing flow for most students [17]. Because these learning contents that maintain at a specific difficulty level constantly may be monotonous for students with high skill, and frustrating for students with low skills. Fortunately, modern ITSs [20] that are capable of accurately identifying student's condition (e.g., competencies, emotion states, or flow), may be able to provide adaptive learning contents by selecting specific problems of appropriate properties to strike a balance between the perceived challenge and a student's skill level [7, 15]. However, despite the recent advances in affect detection and competencies detection, the development of automatic flow detector has been lack of attention. Hence, this study presents a flow detector designed to identify learners that are in the flow state, when interacting with a step-based TS for linear algebra (LA).

2 Method

2.1 Participants

The dataset was collected over a period of two months. Participants were 78 college students required to have a basic understanding of high-school algebra and not have taken any college-level linear-algebra courses. Each student took from two to three weeks to complete the study over multiple sessions. In total, 55 students completed the study.

2.2 Domain and Procedure

The step-based TS used in this study is called Tempranillo. Within Tempranillo, students complete LA problems and are formatively assessed based on a knowledge component (KC) model, providing information about their knowledge to their teachers, while being assisted with scaffolding, help, and feedback.

Our work used the "linear transformations" and the "orthogonality" of LA domain as covered in the first-year college LA course. The fifteen primary KCs were: Definition of Linear Transformation (KC1), Definition of Kernel (KC2), Definition of Image and Range (KC3), Theorem 4.2.1 in [12] (KC4), Theorem 4.2.4 in [12] (KC5), Similarity (KC6), Definition of Distance Between X and Y (KC7), Theorem 5.1.1 in [12] (KC8), Cauchy-Schwarz Inequality (KC9), Orthogonality (KC10), Scalar and Vector Projections (KC11), Orthogonal Complement (KC12), Fundamental Subspaces (KC13), Theorem 5.1.1 in [12] (KC14), $W = U \oplus V$ (KC15).

All participants experienced an identical procedure and presented with same materials. The procedure was as following: 1) a background survey; 2) read a textbook covering the target domain knowledge; 3) took a pretest; 4) solved the same fifteen training problems in the same order on Tempranillo; and 5) took a posttest. The pretest and posttest were identical. A KC-based score for each KC application was given by identifying all relevant KCs over all test questions. In the following sections, the evaluation of the competence of each student is provided based on the sum of all of these KC-based scores. The tests contained 36 test questions which cover 41 KC occurrences. All test scores were normalized to fall in the range of [0,1] for comparison purposes.

3 Flow Detector

3.1 Label Generation

To build a model to detect whether a student is in flow, an operational definition of flow that can be used as training label (i.e. a "ground truth" label of the construct being detected) is required. This study operationalizes flow as the difference between a student's perception of skills and challenge, by using the approach of describing the progress of the individual in terms of challenges and skills during an activity because challenge and skills have been reported to be reliable indicators for measuring flow [13]. This study adopts the probe (5-point Likert scale ratings) used in [14] to measure challenge and skills as primary data for our repeated measures of flow during the learning tasks. During the experiment, participants work through learning tasks comprising steps. At the end of each of the steps, the probe is presented to record the participants' perceptions of challenge and skill.

Data collected by using these probes are used to map the flow condition of each student through the two-dimensional skills–challenge space (termed "flow space") during a learning task (see Fig. 1). On categorizing a student's flow conditions, Csikszentmihalyi's three-channel model of flow is used. This is consistent with his ideas that the flow channel represents the movement of a learner through an activity. The each skills–challenge pairs in Fig. 1 is categorized into three flow conditions that are frustration, flow, and boredom. Students were considered to be in flow when their rating of challenge and skill scores is (1,2), (2,3), (3,4), (4,5), (1,1), (2,2), (3,3), (4,4), (5,5), (2,1), (3,2), (4,3), or (5,4) (defined by that $|$ challenge – skills $| \leqq 1$); to be frustrated when challenge was greater than skills, as following: (1,3), (2,4), (3,5), (1,4), (2,5), or (1,5) (defined by that challenge – skills $\geqq 2$); and to be bored when challenge is lower than skills, as following: (3,1), (4,2), (5,3), (4,1), (5,2), and (5,1) (defined by that challenge – skills $\leqq 2$). Based on this operational definition, 333 of the 580 labels in this study are labeled as in flow. Of the remaining 247 labels treated as not in flow, 187 are labeled as boredom, and 60 are labeled as frustration.

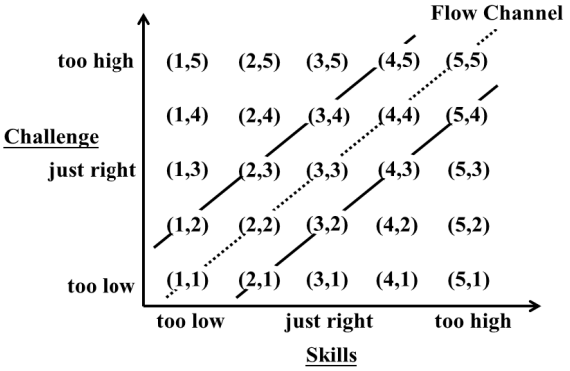


Fig. 1. Coordinate pane formed by the skills-challenge axis

3.2 Data Features

This study selects a set of action-level features based on a combination of theory and prior work. In particular, prior research on validating flow theory [15] and developing automatic affect detectors [9, 21] influenced the design of features in this study.

The first category of basic features focused on the basic properties of a step, as following: (1) type of the step (providing definition/receiving scaffolding hints/solving problem), (2) step result (no response required/correct step/error step), (3) step number (the number of steps that have been done previously on the current task), (4) the proportion of correct steps, (5) the proportion of help requests, and (6) the proportion of answers that were incorrect and received bug messages.

The second category of basic features focused on overall response time and time spent processing tutor-provided assistance, as following: (7) average response time, (8) the average unitized response time (in the standard deviations above or below the mean for students on the current task), (9) the proportion of actions that involved a fast response after the student received a bug message (bug messages indicate why the system thinks the student made an error), (10) the proportion of slow responses after a bug message, (11) the proportion of fast responses after requesting a hint, (12) the proportion of slow responses after requesting a hint, (13) the proportion of fast responses after receiving a hint and entering a correct answer, and (14) the proportion of slow responses after receiving a hint and entering a correct answer.

The third category of basic features focused on the affect information [21] of a student, as following: (15) average keystroke duration, (16) average mouse click duration, (17) standard deviation of keystroke duration, (18) standard deviation of keystroke duration, (19) standard deviation of mouse click duration, (20) average keystroke latency, (21) average mouse click latency, (22) standard deviation of keystroke latency, (23) standard deviation of mouse click latency, (24) valence, and (25) arousal.

To assess the affective state of a student (i.e. (24) and (25)), the Self-Assessment Manikin (SAM), an affective rating system devised by Lang [11], was used to acquire affective ratings. The SAM is a non-verbal pictorial assessment that is designed to assess the two emotional dimensions: valence and arousal directly by means of two sets of graphical manikins (the third dimension (i.e. dominance) is typically ignored in latest studies). The SAM has been extensively tested and has also been used in diverse theoretical studies and applications [3-5]. The SAM takes a very short time to complete (5 to 10 seconds), and there is little chance of confusion with terms as in verbal assessments. The SAM was also reported to be capable of indexing cross-cultural results, and the results obtained using a Semantic Differential scale. The SAM was presented to the students each time right before the presentation of the skills-challenge probes.

Some of these features relied upon cut-off thresholds. This study chooses an optimized cut-off threshold using a procedure discussed in the next section.

3.3 Detector Development

This study develops flow detectors for detecting students that are frustrated, boredom, or in flow using 2-class detection models built by using step-regression (not step-wise

regression). For building each 2-class detection model, step regression firstly fits a linear regression (LR) model to detect the labels of the flow condition (i.e. boredom, flow, or frustration) using the features collected from pedagogical interactions during the experiment. Then, all students for whom the LR detected values equal to or higher than a pre-chosen threshold are assessed to be frustrated, in flow, or bored. For the choice of the thresholds, 0.5 is used because it is standard convention for 2-class classification models (0.5 is halfway between 0 and 1). This study takes numerical detections of flow conditions and transforms them into a binary detection of whether a student is bored, in flow, or frustrated, which can be compared to the labels initially derived from the reported challenge and skills.

The models of detecting flow conditions are validated using 10-Fold CV. In each of the 10-Fold CV, the data points are divided into 10 groups. Each of the groups serves successively as a test set, whereas the remaining 9 groups serve as a training set to build a model. The cross-validated performance assesses the model's predictive performance when applied to new data, which is an indicator of the model's ability to generalize. Two criterion are used to determine goodness for each model: (1) Area Under the ROC Curve (AUC), and (2) Cohen's Kappa (or κ). The AUC and κ are used as indicators to access that the possibility that successful classifications are occurred by chance [2].

This study develops detectors by using all the features discussed above in section 3.2. Some of the features that are depend on cut-off parameters (e.g., how many seconds differentiates a "fast response" from a "slow response"). These parameters are optimized by selecting a best cut-off threshold judged by using the AUC values achieved by each of the step-regression models of single-feature. To reduce the possibility of over-fitting[1], this study reduces the parameter space of models before fitting full models by using Akaike criterion [1] for model selection. In addition, all the co-linear features are also excluded. For attribute selection, this study applies forward selection to find the best model. In a forward selection process, the best single-feature model is chosen, and then the feature that improves the model most is repeatedly added into the model until no more features can be added to improve the model.

4 Results

The best-fitting models for each feature set are shown in Table 1. The first column in Table 1 shows the detection target of the models. The second column in Table 1 shows the built models. The third and firth columns in Table 1 list the AUC and κ values of the corresponding models. The model built to detect students that are in flow achieves an acceptable cross-validated κ of 0.33 (which is 33% better than the baseline performance [6] that is achieved by chance). The AUC value for the model is 0.64, which indicates that the model can differentiate a student that is in flow from not being in flow, at 64% of the time. This level of performance is significantly better than chance ($p < .001$), and may be considered to be sufficient to enable fail-soft intervention.

[1] A set of features that does not generalize well from old data to new data.

Table 1. Step regression models with cross-validated AUC and κ (higher values of model coefficients correspond to the target of detections)

Target of Detection	Model	AUC	κ
Boredom	0.0004 * AVG_MouseClickDuration + 0.0439 * Step-Number - 0.0689 * Valence + 0.0674 * Arousal + 0.0247	0.86[c]	0.50[c]
In Flow	-0.0004 * AVG_MouseClickDuration - 0.036 * Step-Number + 0.048 * Valence + 0.5509	0.64[c]	0.33[c]
Frustration	0.0252 * Valence - 0.0596 * Arousal + 0.3375	0.91[c]	0.49[c]

[a] $p < .05$, [b] $p < .01$, [c] $p < .001$ on rejecting null hypothesis (i.e. classifier predicting at random)

Table 1 also reveals that the models built to detecting boredom and frustration achieve moderately better cross-validated AUC and κ than the model built to detect students that are in flow. In that 0.91 and 0.86 of AUC are achieved by the built models for detecting frustration and boredom, respectively. Furthermore, 0.49 and 0.50 of κ are achieved for detecting frustration and boredom, respectively.

The features that constitute the three models are similar, and that all of the models are quite simple. In all of the models, a common feature is valence. In addition, arousal is also a common feature of the models for detecting frustration and boredom. Because the flow conditions are related to personal experience, it seems reasonable that valence and arousal would be associated with these conditions. A positive coefficient for valence indicates that students who are in a positive emotion state are more likely to be in flow; whereas a negative coefficient for the number of steps shows that more steps done by the students, the less likely the students to be in flow. For the coefficients of average mouse click duration in Table 1, although the values are relatively small, exist in the models for detecting flow and boredom. The average mouse click duration has been previously shown to predict affective states [21]; as such, it makes sense that this feature may be related to the flow conditions. The results indicate that the average mouse click duration is positively correlated to boredom but negatively correlated to flow. Furthermore, the positive coefficient for the number of steps in the model for detecting boredom indicates that the longer the mouse click durations are, the more likely a student is bored.

5 Discussion and Conclusions

This study presents models that can distinguish with reasonable accuracy whether a student is in flow, boredom, and frustration. The flow conditions are operationally defined as the difference between the challenges received in the tutor and the perceived skills level. These models are developed in the context of pedagogical interactions between human students and Tempranillo (i.e. a tutor for LA), and are validated based on 10-Fold CVs. The built models can identify a student that is bored 86% of the time, performing 50% better than chance; identify a student that is in flow 64% of

the time, performing 33% better than chance; and identify a student that is frustrated 91% of the time, performing 49% better than chance. These models are built based on four features, including the two affective dimensions and the two features related to the interactions with the learning software. The results are in line with theory [7] that suggests the relationship between emotion related constructs and flow conditions, and also support the use of affect detectors on detecting flow conditions [15]. The finding of the relationship between mouse click duration [21] and flow conditions also suggests that the students' usage of standard input devices is a feature which may contain abundant information for developing future low-cost affect detectors. The flow detectors have considerable potential usefulness for developing ITSs because traditional non-adaptive learning contents are likely to fail to promote flow experience of students during learning, and that the traditional methods (i.e. surveys) to assess students' flow conditions required the students to be interrupted from the performing activities. An ITS with automatic detector of flow condition can identify students' conditions related to flow construct and offer them remediation specific to their needs, helping a student to maintain in flow experience to hold the intrinsic motivation that is necessary for achieving optimal learning [8]. Furthermore, the improvement of the detectors' performance may lead to flow-path research in ITSs, that is, tracing the path of a student when performing a task through process points shown in Fig. 1. The exploration of the built models' generality to other learning domains and types of tutors may also be an important area of future work.

Acknowledgements. This work was fully supported by the Taiwan Ministry of Science and Technology under grant numbers NSC-102-2220-E-009-023 and NSC-102-2627-E-010-001. This work was also supported in part by the UST-UCSD International Center of Excellence in Advanced Bioengineering sponsored by the Taiwan Ministry of Science and Technology I-RiCE Program under grant number NSC-101-2911-I-009-101; and in part by "Aim for the Top University Plan" of the National Chiao Tung University and Ministry of Education, Taiwan, R.O.C.

References

1. Akaike, H.: A New Look at the Statistical Model Identification. IEEE Transactions on Automatic Control 19, 716–723 (1974)
2. Ben-David, A.: About the Relationship between Roc Curves and Cohen's Kappa. Eng. Appl. Artif. Intell. 21, 874–882 (2008)
3. Bolls, P.D., Lang, A., Potter, R.F.: The Effects of Message Valence and Listener Arousal on Attention, Memory, and Facial Muscular Responses to Radio Advertisements. Communication Research 28, 627–651 (2001)
4. Bradley, M.M.: Emotional Memory: A Dimensional Analysis. In: van Goozen, S.H.M., van de Poll, N.E., Sergeant, J.A. (eds.) Emotions: Essays on Emotion Theory, pp. 97–134. Lawrence Erlbaum, Hillsdale (1994)
5. Chang, C.: The Impacts of Emotion Elicited by Print Political Advertising on Candidate Evaluation. Media Psychology 3, 91–118 (2001)

6. Cohen, J.: A Coefficient of Agreement for Nominal Scales. Educational and Psychological Measurement 20, 37–46 (1960)
7. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper Perennial (1991)
8. Egbert, J.: A Study of Flow Theory in the Foreign Language Classroom. Canadian Modern Language Review/La Revue Canadienne des Langues Vivantes 60, 549–586 (2004)
9. Epp, C., Lippold, M., Mandryk, R.L.: Identifying Emotional States Using Keystroke Dynamics. In: Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems, pp. 715–724. ACM, Vancouver (2011)
10. Kort, B., Reilly, R., Picard, R.W.: An Affective Model of Interplay between Emotions and Learning: Reengineering Educational Pedagogy-Building a Learning Companion. In: Proceedings of the IEEE International Conference on Advanced Learning Technologies, pp. 43–46. IEEE (2001)
11. Lang, P.J.: Behavioral Treatment and Bio-Behavioral Assessment: Computer Applications. In: Sidowski, J., Johnson, J., Williams, T. (eds.) Technology in Mental Health Care Delivery Systems, pp. 119–137. Ablex Pub. Corp., Norwood (1980)
12. Leon, S.J.: Linear Algebra with Applications. Pearson Education (2007)
13. Novak, T.P., Hoffman, D.L.: Measuring the Flow Experience among Web Users. Interval Research Corporation 31 (1997)
14. Pearce, J.M., Ainley, M., Howard, S.: The Ebb and Flow of Online Learning. Computers in Human Behavior 21, 745–771 (2005)
15. San Pedro, M.O.Z., Baker, R.S.J.d., Gowda, S.M., Heffernan, N.T.: Towards an Understanding of Affect and Knowledge from Student Interaction with an Intelligent Tutoring System. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 41–50. Springer, Heidelberg (2013)
16. Sedighian, K.: Challenge-Driven Learning: A Model for Children's Multimedia Mathematics Learning Environments. In: Conference of Educational Multimedia & Hypermedia & Educational Telecommunications (1997)
17. Sessink, O.D., Beeftink, H.H., Tramper, J., Hartog, R.J.: Proteus: A Lecturer-Friendly Adaptive Tutoring System. Journal of Interactive Learning Research 18, 533–554 (2007)
18. Shernoff, D.J., Csikszentmihalyi, M.: Flow in Schools: Cultivating Engaged Learners and Optimal Learning Environments. In: Gilman, R., Huebner, E.S., Furlong, M.J. (eds.) Handbook of Positive Psychology in Schools, pp. 131–146. Routledge/Taylor & Francis Group (2009)
19. Shernoff, D.J., Csikszentmihalyi, M., Shneider, B., Shernoff, E.S.: Student Engagement in High School Classrooms from the Perspective of Flow Theory. School Psychology Quarterly 18, 158 (2003)
20. Vanlehn, K.: The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education 16, 227–265 (2006)
21. Zimmermann, P., Guttormsen, S., Danuser, B., Gomez, P.: Affective Computing–a Rationale for Measuring Mood with Mouse and Keyboard. International Journal of Occupational Safety and Ergonomics 9, 539–551 (2003)

To Quit or Not to Quit: Predicting Future Behavioral Disengagement from Reading Patterns

Caitlin Mills[1], Nigel Bosch[2], Art Graesser[3], and Sidney D'Mello[1,2]

[1,2] Departments of Psychology and Computer Science, University of Notre Dame
Notre Dame, IN 46556, USA
{cmills4,pbosch1,sdmello}@nd.edu
[3] Department of Psychology and Institute for Intelligent Systems, University of Memphis,
Memphis, TN 38152, USA
a-graesser@memphis.edu

Abstract. This research predicted behavioral disengagement using quitting behaviors while learning from instructional texts. Supervised machine learning algorithms were used to predict if students would quit an upcoming text by analyzing reading behaviors observed in previous texts. Behavioral disengagement (quitting) at any point during the text was predicted with an accuracy of 76.5% (48% above chance), before students even began engaging with the text. We also predicted if a student would quit reading on the first page of a text or continue reading past the first page with an accuracy of 88.5% (29% above chance), as well as if students would quit sometime after the first page with an accuracy of 81.4% (51% greater than chance). Both *actual* quits and *predicted* quits were significantly related to learning, which provides some evidence for the predictive validity of our model. Implications and future work related to ITSs are also discussed.

Keywords: engagement, disengagement, affect detection, reading, ITSs.

1 Introduction

One of the benefits afforded by intelligent tutoring systems (ITSs) and other advanced learning technologies is the students' ability to move at their own pace through learning sessions. In many systems, students have choice over the topics and activities they engage in. Importantly, they can also choose how long to spend on each one. However, one caveat to this type of choice is that disengagement can occur before activities or topics are completed, leaving vital information unseen. Therefore, identifying when disengagement will occur may help inform timely interventions, such as temporarily suppressing choice or providing motivational messages to persist [1], as well as development of educational materials that keep students engaged in order to achieve learning goals.

There has been a growing interest in automatically detecting students' affective states and engagement during learning (see [2] for a review). One focus, in particular, has been on identifying behaviors associated with engagement/disengagement during

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 19–28, 2014.

learning because of the necessity of engagement for learning [3]. In fact, previous research has had success in modeling and detecting various types of disengaged behaviors within ITSs [4–9]. For example, an automatic detector for "gaming" the system can reliably detect when students exploit the system to get correct answers [4]. Another detector also made it possible to recognize if a student is purely off-task or engaging in on-task conversation [10]. These types of detectors have led to helpful design interventions, as well as more accurate student models of learning [11].

Previous work has also been able to classify different levels of engagement using log files. Cocea and Weibelzahl [12] classified 10-minute intervals of a learning session as one of three levels of engagement: engaged, disengaged, or neutral. Ground truth was achieved from labels provided by expert human coders. This study reported accuracies 71% greater than baseline (Cohen's kappa = .713) using features extracted from log file information, such as reading behaviors (i.e., average time, number of pages) and test information (i.e., average time, number of tests, correct answers). This model displayed impressive accuracies for diagnosing students' current level of engagement during the specified 10-minute intervals and appears to generalize across multiple learning environments [13]; however, predictors of future engagement have not yet been established.

All of the detectors mentioned thus far have focused on a specific aspect of engagement (or disengagement), such as behaviors like gaming the system. Indeed, engagement has been operationalized in numerous ways due to its multi-faceted nature [14]. Specifically, engagement can be thought of as encompassing three distinct components: (a) affect (e.g., positive and negative feelings), (b) behavior (e.g., persistence, effort), and (c) cognition (e.g., goals, self-regulated behaviors) [15]. Typically, disengagement detectors target some combination of these three components, initially relying on external coders to make some inference about the cognitive/affective components based on student behavior or self-report measures. One problem then, as noted by Baker and Rossi [14], is that models of engagement are difficult to validate beyond face validity because engagement is complex, and ground truth is achieved via human judgments, which are inherently subjective.

The current research focus is on a behavioral indicator of disengagement. Specifically, we build a predictor of behavioral disengagement, which we operationally define as the point at which a student opts to stop interacting with (quits) a given activity within a learning session. Importantly, this operationalization of behavioral engagement does not require any external human coders to initially establish ground truth. A distinguishing aspect of this work is that our model is *predictive* in that disengagement on the current activity N is predicted from interaction patterns observed during the previous N-1 activities instead of *diagnostic*, where actions in N are used to detect disengagement in N after it occurs. A predictive model can ostensibly be used to prevent the onset of disengagement, which is advantageous since disengagement and boredom are long-lasting persistent negative states [16].

The instructional reading task in the present research is a self-paced learning task where students control the pace and time spent on each text. Self-paced reading is an important component within a number of interactive learning technologies and ITSs, such as in Operation ARA!, iSTART, and ELM-ART [17–19]. For example,

in Operation ARA!, students read an electronic textbook before engaging in tutorial dialogs. We use sensor-free information from previous activities (i.e., log files of reading patterns) to predict quitting before the current text ever begins. The ability to unobtrusively predict when quitting behaviors will occur provides the foundation for effective design of interventions to keep students engaged.

2 Methods

2.1 Data Collection

Participants. Data was obtained from 173 undergraduate students from a private university in the Midwest and a large public Mid-south university in the US who participated for course credit.

Texts. Students spent a total of 30 minutes completing reading from instructional texts. The reading task consisted of eight texts on scientific research methods topics (disguised measures, gathering data, hypotheses, scientific method, construct validity, variables, criterion of precision, expectancy bias) adapted from a popular textbook [20]. Texts had an average length of 1068 words (SD = 35.7) with a Flesch-Kincaid Grade Level score of about 13.5, which is indicative of some difficulty. Order of topics was counterbalanced across students.

Procedure. Students completed an informed consent and a short trial to familiarize themselves with the interface. Each student was then left alone in a small room for 30 minutes with the reading interface. No other devices or distractions, such as a watch or cell phone were permitted. Students were presented with a blank screen with a button labeled READ to begin the reading task. A text was presented once a student selected the READ button. Texts were presented one page at a time with 77 words per page. Students could use the right and left arrow keys to navigate through the text with the ability to move backward to previous pages or forward to the next page. Students had the capability of quitting the text at any point in time by pressing the 'C' key ("Change to a different text"). If students hit the 'C' key, a new text would appear. Students could press the 'C' key up to seven times and receive a new text (eight texts). Only data from the first time students viewed each text were analyzed in order to avoid familiarity biases after seeing a text multiple times. In sum, over the course of the 30 minutes, students were able to read as much or as little of each text as they chose.

As a learning measure, students completed a posttest involving 48 multiple-choice questions (six per text) about the information from the eight texts after the reading session. Questions were developed in adherence to the Graesser-Person question-asking taxonomy [21]. The questions targeted specific sections in the text, such that answers were not apparent unless the targeted section of text was read.

Quitting Behaviors. Students' reading time information (e.g., how long they spent on each page) was collected during the reading task. Every text was classified as *Quit*, *Completed*, or *Timeout* based on how the student interacted with the text. Instances labeled Quit consisted of texts that students started reading, but hit the 'C' key to exit the text before reaching the end of the text. Completed instances were texts that were read by students in their entirety. Finally, an instance was labeled as Timeout if the

learning session was interrupted in the midst of reading due to the 30 minute time limit, and therefore could not be classified as Quit or Completed. The instances (texts) that were labeled as Timeout were removed from the dataset because we were not interested in a forced exit from a text. In total, there were 911 instances used to build models, where students either quit (n = 311) or completed (n = 600) a text after beginning to read it for the first time, thereby yielding a 34% rate of quitting. On average, students quit texts after reading 32.9% of the pages (SD = 28.3%).

2.2 Model Building

Feature Engineering and Selection. A total of 18 features were computed from reading behaviors and reading times. For each text analyzed (text N), two types of features were extracted: previous text information (text N-1) and cumulative previous texts information [e.g., features from all previous texts (1 to N-1) averaged]. No feature used any information from the current text being classified or any text that was viewed later, which is essential for predictive modeling. Table 1 contains a list of the features that were computed based on the logged reading behaviors (e.g., reading times, quit behaviors).

Using a backward feature selection method, features from the *previous text* feature set were removed one at a time depending on model performance after removing a feature[1]. If model performance declined, the feature was retained for the final model. Next, features from the *cumulative previous texts* feature set were removed in the same manner. Finally, backward selection was used on the combined set of remaining features from the two feature sets to produce a final set of features for each classification task. There were no features that correlated higher than .80 or higher, which was used as a threshold to remove correlated features.

Supervised Classification. We used supervised machine learning to build predictors for three different classification tasks. The first task attempted to classify if a student would quit at any point during a particular text vs. completely read the text. The second task attempted to classify if a student would quit on the first page of the text vs. continue reading past the first page (even if they might eventually quit at some point). Finally, the third task aimed to classify if a student would quit at any point past page one vs. completely read the text. Six binary classification algorithms provided in Rapid Miner were used for each of the models, including Bayes Net, RIPPER (JRip implementation), C4.5 (J48 implementation), Naïve Bayes, SMO, and VFI.

Model Validation. All models were evaluated using leave-one-student-out cross-validation, in which k-1 students are used in the training data set. The model is then tested on the student who was not used in the training data. This process is repeated until every student has been used as the testing set one time. The average results from the k iterations provide an estimation of classification accuracy. Cross-validating at the student level increases confidence that models will be more generalizable when applied to new students because the testing and training sets are independent.

[1] We also tested models using all 18 features, which exhibited worse performance (assessed via Cohen's Kappa) than each of the three final models using the selected features.

Table 1. Description of features and indication of which final model(s) each was included (+)

Features	Quitting on Any Page vs. Completing	Quitting on Page 1 vs. Continuing	Quitting After Page 1 vs. Completing
Previous Text Only			
Page 1 Reading Time (RT)		+	+
Quit On Page 1 (Yes/No)		+	+
Location of Quit (First 3 Pages, After 3 Pages, None)		+	+
Max Page Number Seen			
Median Page Reading Time (RT)			
Minimum Page Reading Time			
Maximum Page Reading Time			
Proportion of Text Seen		+	+
Reading Time 1 Page Before Exit	+	+	
Proportion of Pages < 5s Reading Time	+	+	+
Total Reading Time		+	+
Text Exit (Quit/Completed)	+		+
Cumulative Previous Texts Seen			
Maximum Page Number Seen	+	+	
Median Page Reading Time		+	
Minimum Page Reading Time			
Maximum Page Reading Time			
Proportion of Pages < 5s Reading Time	+	+	
Total Reading Time			

Metrics. Classification accuracy was evaluated using precision, recall [22], and Cohen's kappa [23]. Precision represents the percentage of texts classified as Quit that were actually Quit. Recall represents the percentage of texts that were actually Quit and also correctly classified as Quit. Cohen's kappa takes base rates into consideration and indicates the degree to which the model is better than chance (kappa of 0) at correctly predicting whether the text will be Quit or Completed. A kappa value of -0.5 or 0.5 would indicate the model is classifying -50% worse or 50% better than chance, respectively. We also report percent correctly classified (accuracy), but caution that this should be interpreted cautiously since class imbalance tends to inflate accuracy.

3 Results and Discussion

3.1 Quitting on Any Page vs. Completing the Text

The first classification was to attempt to predict whether a student would quit a text at any point or complete the text. The six classifiers were used to predict quitting based on the features extracted from text(s) previously presented to the student (see above). The best model for predicting overall quitting behavior used the Bayes Net algorithm. The kappa for this model indicates the model's performance is 48.4% higher than chance. Five features were used in this best model (indicated in Table 1). Model fit statistics are presented in Table 2.

Table 2. Performance measures for the three classification tasks

	Quit Class		Completed/ Continued Class			
	Precision	**Recall**	**Precision**	**Recall**	**Kappa**	**Accuracy**
Any Page	64.8%	68.2%	83.1%	80.8%	.484	76.5%
First Page	38.7%	33.0%	92.9%	94.4%	.293	88.5%
Subsequent Pages	67.5%	60.5%	85.9%	89.2%	.514	81.4%

We also examined the confusion matrix for this predictor (Table 3). It is notable that both true positives and true positives were higher than false positives or false negatives. Given a prediction of Quit, odds were nearly 2:1 (64.8% precision) that the prediction is correct (a "hit" rather than a "false alarm"), and so an intervention can be given with a good degree of confidence.

3.2 Quitting on the First Page vs. Continuing

The next classification task attempted to predict if students would quit on the first page vs. continue reading, which occurred 10% of the time. Predicting these instances may provide information for more immediate interventions before quitting occurs on page one. For this task, Quit labels were restricted to the cases where students quit the text on the first page. Any quit past page one is classified as a *Continue Past Page One*. The best classifier was a Bayes Net algorithm using 10 features (see Table 1). Performance measures are provided in Tables 2 and 3, respectively.

This model was able to classify texts where students quit on the first page 29.3% higher than chance using information from previous text(s). Although this predictor does not perform as well as the previous model, this model provides an important classification at a relatively small window size (page level). The confusion matrix for

the first page Quit model illustrates the class imbalance well. Due to the large proportion of Continue Past Page One instances (.903), Quit instances were not likely to be detected as well as Quit instances on any page. Interventions given based on these predictions must be especially cautious, using a "fail soft" approach. The low precision (38.7%) implies that less than half of the Quit predictions will be correct, due largely to the class imbalance.

Table 3. Confusion matrices for the three classification tasks

Any Page	**Predicted Quit**	**Predicted Completed**	**Priors**
Actual Quit	0.68 (hit)	0.32 (miss)	0.34
Actual Completed	0.19 (false alarm)	0.81 (correct rejection)	0.66
First Page	**Predicted Quit**	**Predicted Continued**	**Priors**
Actual Quit	0.33 (hit)	0.67 (miss)	0.10
Actual Continued	0.06 (false alarm)	0.94 (correct rejection)	0.90
Subsequent Pages	**Predicted Quit**	**Predicted Completed**	**Priors**
Actual Quit	0.61 (hit)	0.39 (miss)	0.27
Actual Completed	0.11 (false alarm)	0.89 (correct rejection)	0.73

3.3 Quitting after the First Page vs. Completing the Text

The third classification task attempted to predict quitting once students read past the first page vs. completing. Since 10% of texts were quit on page one, it is also useful to understand when students will quit after reading past the initial first page. Classifying quitting once students read past page one will allow interventions to target students who are moving through the text (past the initial page), yet decide to stop before completing the entire text.

The cases where students quit on the first page were not included in this task, leaving 223 instances labeled as Quit and 600 labeled as Completed. The best classifier was a C4.5 classifier, which was able to perform 51.4% higher than chance (see Tables 2 and 3 for performance summary). Interestingly, this model differed from the first two classifications tasks, as only the features containing information from the previous text were included in the model (see Table 1). Precision for this model was 67.5% and had a lower proportion of false alarms than in the "Any Page" model, indicating some potential for use with interventions.

3.4 Predictive Validity

We also examined the relationship between posttest performance and quitting. First, we correlated students' proportion of correct responses on the posttest (posttest performance)

with their proportion of actual quits, Pearson's $r = -.314$, $p < .001$. Indeed, this negative correlation provides some validation for the use of quitting as a measure of behavioral disengagement, as disengagement is associated with negative learning [5].

It is also important to establish whether posttest performance was related to our model's predicted quits. Students' posttest performance was also correlated with the proportion of predicted quits, based on model classification (i.e., Quit vs. Finished using the Bayes Net algorithm), $r = -.332$, $p < .001$. This correlation gives us some confidence in our model's predictive validity, since our predicted quits are negatively related to learning as well.

Finally, we also investigated the relationship between actual quits and predicted quits at the student level. The proportion of students' actual quits was highly correlated with the proportion of predicted quits (as predicted using the Bayes Net algorithm), $r = .934$, $p < .001$. This positive relationship gives us further confidence in our predictor, as students' quitting behavior was closely tied to the model's predictions.

4 General Discussion

We developed three models of quitting by analyzing log files from previous texts: (1) any point during a text vs. completing the text (kappa of .484), (2) on the first page vs. continuing reading (kappa of .293), and (3) past the first page vs. reading to completion (kappa of .514). Importantly, we are attempting to predict future behavior before the activity is even started from easily available reading measures, so this form of modest kappa is expected. Additionally, the kappa values achieved using these predictors are similar to those found in previous disengagement detectors [24, 9], however meaningful comparisons of results are complicated by differences in how disengagement is conceptualized.

The features that were used in the final models reveal that reading times on key pages are important for predicting quitting. For example, reading time on the page immediately before quitting the previous text was included in two of the final models and the proportion of pages with reading time less than five seconds was included as a feature in all three final models. Furthermore, the reading time on the first page was included in two out of three final models. Previous quitting behavior was also relevant in these predictors. In fact, students previously quitting on the first page, as well as what section of the text they quit (first three pages, after first three pages, or completed) were also relevant features in two of the final models. These predictors indicate that past (reading) behavior can be a good indicator of future behavior.

Predicting quitting behaviors may open up new avenues for interventions and instructional designs in order to facilitate better learning. When disengagement behaviors, such as gaming the system, are detected, a system can reactively respond by reintroducing the content that is missed due to gaming for improved learning [11]. The predictors presented in this paper are an initial step for interventions that can occur *proactively*, since the prediction is made before the text is even read. For example, the utility of the text can be highlighted as a potential motivator to continue if quitting is predicted [25]. Or the system might suggest a change of topics or that the student may take a short break.

It is important to note that these models are not without limitations. First, these models were fit using an instructional reading task, which may not generalize to other learning environments. Second, our results cannot be generalized beyond the current sample. Third, since this study was conducted in the lab, future work should investigate the effectiveness of similar models using log files from actual ITS learning sessions. Future work should also include attempts to combine these reading behavior features with other trait-based features, such as prior knowledge and interest, which might further improve prediction rates. This paper provides initial groundwork on predicting behavioral disengagement via quitting behaviors, but we believe further development of these types of models are promising for adaptive ITSs to intervene before the moment of disengagement occurs.

Acknowledgment. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. Kelly, K.M., Heffernan, N., D'Mello, S., Namais, J., Strain, A.: Added Teacher-Created Motiational Video to an ITS. In: The Twenty-Sixth International FLAIRS Conference, pp. 503–508. AAAI Press, Menlo Park (2013)
2. Calvo, R.A., D'Mello, S.: Affect detection: An interdisciplinary review of models, methods, and their applications. IEEE Trans. on Affect. Comput. 1, 18–37 (2010)
3. Pekrun, R., Linnenbrink-Garcia, L.: Academic emotions and student engagement. In: Handbook of Research on Student Engagement, pp. 259–282. Springer (2012)
4. Baker, R.S., Corbett, A.T., Koedinger, K.R.: Detecting student misuse of intelligent tutoring systems. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 531–540. Springer, Heidelberg (2004)
5. Beck, J.E.: Using response times to model student disengagement. In: Proceedings of the ITS 2004 Workshop on Social and Emotional Intelligence in Learning Environments, pp. 13–20 (2004)
6. D'Mello, S., Cobian, J., Hunter, M.: Automatic Gaze-Based Detection of Mind Wandering during Reading. In: Proceedings of the 6th International Conference on Educational Data Mining, pp. 364–365. International Educational Data Mining Society (2013)
7. Forbes-Riley, K., Litman, D.: When does disengagement correlate with learning in spoken dialog computer tutoring? In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 81–89. Springer, Heidelberg (2011)
8. Rowe, J.P., McQuiggan, S.W., Robison, J.L., Lester, J.C.: Off-Task Behavior in Narrative-Centered Learning Environments. In: AIED, pp. 99–106 (2009)
9. Jang, H.: Supporting students' motivation, engagement, and learning during an uninteresting activity. UMAP 2012 100, 798 (2008)
10. Baker, R.S.J.: Modeling and understanding students' off-task behavior in intelligent tutoring systems. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1059–1068 (2007)

11. Baker, R.S.J.d., et al.: Adapting to when students game an intelligent tutoring system. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 392–401. Springer, Heidelberg (2006)
12. Cocea, M., Weibelzahl, S.: Eliciting motivation knowledge from log files towards motivation diagnosis for Adaptive Systems. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 197–206. Springer, Heidelberg (2007)
13. Cocea, M., Weibelzahl, S.: Disengagement Detection in Online Learning: Validation Studies and Perspectives. IEEE Trans. Learn. Technol. 4, 114–124 (2011)
14. Baker, R.S.J., Rossi, L.M.: Assessing the Disengaged Behaviors of Learners. Des. Recomm. Intell. Tutoring Syst. 155 (2013)
15. Fredricks, J.A., Blumenfeld, P.C., Paris, A.H.: School engagement: Potential of the concept, state of the evidence. Rev. Educ. Res. 74, 59–109 (2004)
16. D'Mello, S., Graesser, A.C.: The half-life of cognitive-affective states during complex learning. Cogn. Emot. 25, 1299–1308 (2011)
17. Brusilovsky, P., Schwarz, E., Weber, G.: ELM-ART: An intelligent tutoring system on World Wide Web. In: Lesgold, A.M., Frasson, C., Gauthier, G. (eds.) ITS 1996. LNCS, vol. 1086, pp. 261–269. Springer, Heidelberg (1996)
18. McNamara, D.S., Levinstein, I.B., Boonthum, C.: iSTART: Interactive strategy training for active reading and thinking. Behav. Res. Methods Instrum. Comput. 36, 222–233 (2004)
19. Millis, K., Forsyth, C., Butler, H., Wallace, P., Graesser, A.C., Halpern, D.: Operation ARIES!: A serious game for teaching scientific inquiry. Serious Games Edutainment Appl., 169–195 (2011)
20. Rosenthal, R., Rosnow, R.L.: Essentials of behavioral analysis: Methods and data analysis. McGraw-Hill, New York (1984)
21. Graesser, A.C., Person, N.K.: Question asking during tutoring. Am. Educ. Res. J. 31, 104–137 (1994)
22. Davis, J., Goadrich, M.: The relationship between Precision-Recall and ROC curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240 (2006)
23. Cohen, J.: A coefficient of agreement for nominal scales. Educ. Psychol. Meas. 20, 37–46 (1960)
24. Baker, R.S.J., De Carvalho, A.: Labeling student behavior faster and more precisely with text replays. In: Proceedings of the 1st International Conference on Educational Data Mining, pp. 38–47 (2008)
25. Jang, H.: Supporting students' motivation, engagement, and learning during an uninteresting activity. J. Educ. Psychol. 100, 798 (2008)

Predicting Affect from Gaze Data during Interaction with an Intelligent Tutoring System

Natasha Jaques[1], Cristina Conati[1], Jason M. Harley[2], and Roger Azevedo[3]

[1] University of British Columbia, 2366 Main Mall,
Vancouver, BC, Canada V6T1Z4
{jaquesn,conati}@cs.ubc.ca
[2] McGill University, 845 Sherbrook Street West,
Montreal, QC, Canada H3A0G4
Jason.Harley@mail.mcgill.ca
[3] North Carolina State University, 2310 Stinson Drive,
Raleigh, NC, USA 27695
razeved@ncsu.edu

Abstract. In this paper we investigate the usefulness of eye tracking data for predicting emotions relevant to learning, specifically boredom and curiosity. The data was collected during a study with MetaTutor, an intelligent tutoring system (ITS) designed to promote the use of self-regulated learning strategies. We used a variety of machine learning and feature selection techniques to predict students' self-reported emotions from gaze data features. We examined the optimal amount of interaction time needed to make predictions, as well as which features are most predictive of each emotion. The findings provide insight into how to detect when students disengage from MetaTutor.

1 Introduction

Emotions play a critical role in human behavior, thought, motivation, and social interaction [21]. An affect-adaptive interface can react and adapt to clues about the user's emotional state; such systems can increase task success [30], motivation [19], and user satisfaction [18]. Affect sensitivity can be especially beneficial in educational contexts, where maintaining positive emotions can lead to increased learning [21].

Our study focuses on predicting feelings of boredom and curiosity experienced during learner interactions with MetaTutor, an ITS designed to support effective self-regulated learning (SRL) [4]. The main contribution of our work is that we explore the usefulness of eye tracking data alone in predicting learner affect in MetaTutor via machine learning. The only other research that has used eye-tracking data to predict emotions has been limited to using hand engineered heuristics to generate gaze-based interventions [29], or has focused on non-gaze features such as pupil dilation [20] [29]. Unlike pupil dilation, gaze features provide insight into the user's attention to various interface elements, and are not sensitive to changes in luminosity. A second contribution is that we investigate curiosity, an emotion not frequently studied in the affective computing literature. Curiosity is considered an emotion related to interest [27],

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 29–38, 2014.

and was included based on Pekrun's research into academic emotions [22].We are aware of few other studies that include curiosity [9][11][24]. Finally, by uncovering which features are most predictive of each emotion, we gain insights into effective methods for constructing an affect-adaptive MetaTutor.

2 Related Work

Emotions experienced in an academic setting are related to students' motivation and academic achievement [21]. For example, boredom is linked to decreased task success, while engagement is associated with user satisfaction [13]. Further, the presence of an empathetic and supportive tutor or pedagogical agent has been shown to enhance learning [32], and reduce stress [23]. For these reasons, researchers have begun investigating how to detect and respond to learners' emotional states. Conati and Maclaren [7] used information about learners' personalities and interaction logs to model emotions using a Dynamic Bayesian Network (DBN). Forbes-Riley et al. predicted disengagement from acoustic and dialog features [13].

Physiological sensors, including wireless skin conductance bracelets, pressure sensitive seat cushions, and accelerometers, have been used to predict affect in an ITS context [3]. By combining several data sources, including heart rate, skin conductance, posture, questionnaires and interaction logs, Sabourin and colleagues achieved prediction accuracies of 75% for boredom and 85% for curiosity [24]. Affect can also be detected with a single sensor; D'Mello and colleagues. obtained 60%, 64%, and 70% accuracy in predicting boredom using facial expressions, dialog, and posture, respectively [9].

Eye gaze has been used to detect affect. Findings from psychological research have suggested that blinking often or a lack of fixations on interface text may help predict boredom [28], and that increased pupil diameter may be indicative of stronger emotion [20], [29]. This finding was incorporated in an affect-sensitive ITS that responded in real time to heuristic signs of boredom, such as decreased pupil size or wandering gaze [29]. Gaze Tutor [10] also uses heuristics to respond to gaze, by lsending an intervention message if a student does not look at the tutor or the pedagogical content for ten seconds. In the broader domain of education, eye gaze has been used to predict learning gains [6] [17], problem solving [2], and reading performance [26].

Most closely related to our study is the work by Harley, Bouchet and Azevedo [15] on correlating the emotions experienced during interactions with MetaTutor with output from FaceReader 5.0 software. Because the FaceReader emotions do not map directly to the academic emotions of the study, the authors had to develop their own mapping scheme, but still achieved 75.6% agreement. This suggests that the emotion self-reports collected during the MetaTutor study closely matched participants' actual behavior [15]. Unfortunately, positive emotions (including curiosity) declined over the course of the interaction, demonstrating a need for affective interventions [15].

3 MetaTutor User Study

MetaTutor is an adaptive ITS designed to encourage students to employ metacognitive SRL strategies, while teaching concepts about the human circulatory system [5]. SRL is the ability to manage learning through monitoring and strategy use, and can be a powerful predictor of students' learning gains and academic success [25]. For this reason, the MetaTutor learning environment (Fig. 1) contains an overall learning goal (OLG) and subgoal completion bar (at the top of the screen), for setting and viewing progress toward learning objectives. There are four pedagogical agents (PAs) which appear in turn in the top right corner of the screen. The learning strategies palette (LSP) is located beneath the PAs, and allows the user to initiate interactions such as requesting an evaluation of her current understanding of content [19]. Finally, MetaTutor's text and image contents are displayed in the center of the screen, and are organized via the table of contents (TOC) on the far left.

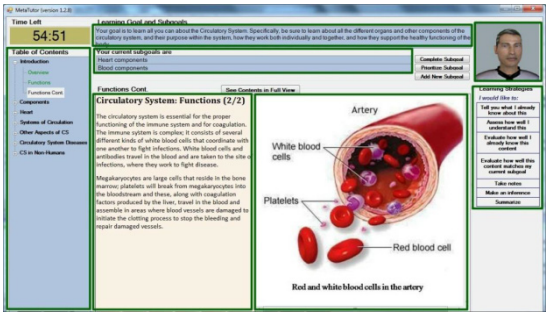


Fig. 1. The MetaTutor Interface

The data used in this analysis was collected from a study of 67 undergraduate students with a variety of academic program concentrations which were not necessarily related to MetaTutor's content. Participants used Meta Tutor for approximately 90 minutes while they were recorded using a number of sensors, including a Tobii T60 eye tracker [15]. Participants also self-reported their concurrent emotions using an Emotions-Value questionnaire (EVQ) developed by researchers at McGill University. The EVQ consists of 19 basic and learning-centered emotion items, and is based on a modified subscale of Pekrun's Academic Emotions Questionnaire [21]. Each item consists of a statement about an emotion (e.g., "Right now I feel bored"), and was rated on a 5-point Likert scale where 1 indicated "strongly disagree" and 5 indicated "strongly agree". The EVQ was filled out at the beginning, and every 14 minutes thereafter during the one hour learning session with Meta Tutor, for a total of 5 self-reports per student. For the purposes of this study, we will focus on two of the most strongly reported emotions (those most frequently rated as 4 or 5 on the Likert scale): boredom (M = 2.60, SD = 0.69) and curiosity (M = 2.93, SD = 0.71) [15].

4 Eye Tracking Data Analysis

The gaze data in this study was collected using a Tobii T60 eye tracker, and takes the form of *fixations* on a single point, and *saccades,* which are the paths between two consecutive fixations. Following the data validation process described in [6], we discarded participants with too few valid gaze samples overall, and were left with a total of 51 participants for analysis. The data was then processed into aggregate features using EMDAT, an open source package for gaze data analysis[1]. The extracted features include application-independent gaze features related to the number of fixations, fixation duration, and saccade length, as well as the angle between two consecutive saccades (the relative path angle) and the angle between a saccade and the horizontal plane (absolute path angle) [6]. We did not include features related to pupil dilation because the data was collected in a room with a window.[2]

In addition to application-independent features, we include features related to specific Areas of Interest (AOIs) within the MetaTutor interface. Following [6], we defined seven AOIs (which are outlined in green boxes in Fig. 1): Text Content, Image Content, Overall Learning Goal (OLG), Subgoals, Learning Strategies Palette (LSP), Agent, and Table of Contents (TOC). We include features such as the duration of the longest fixation on a given AOI, the proportion of fixations and time spent on an AOI, and the number and proportion of gaze transitions between each pair of AOIs. We also include time to first fixation on the AOI, time to last fixation, and the total fixation time. In total, we have 166 features.

5 Machine Learning Experiments

We treat predicting boredom and curiosity as two separate binary classification problems. Although boredom and curiosity could be considered mutually exclusive states, the data does not support this approach. While there was a significant negative correlation between the ratings of boredom and curiosity ($r = -.333$, $p < .001$), in 18% of the self-reports both curiosity and boredom were rated as present simultaneously, and in 13% they were both absent.

Classification labels were based on the EV self-reports. We did not include the first round of reports, because they were collected before participants began using the learning environment. Ratings of 3 or higher were labeled as Emotion Present (EP), and ratings of less than 3 were labeled as Emotion Absent (EA), as in [15]. For classification, we used 10-fold cross validation (CV), and four algorithms available in the Weka data mining toolkit: Random Forests (RF), Naïve Bayes, Logistic Regression, and Support Vector Machines (SVM), chosen because they showed the most promising performance in initial tests. We also use 10-fold CV to tune the parameters of the algorithms. Results are reported in terms of both accuracy (percentage of correctly classified data points), as well as Cohen's kappa, a measure of classification

[1] `http://www.cs.ubc.ca/~skardan/EMDAT/index.html`
[2] Pupil dilation is more sensitive to luminance than to affect [31].

performance that accounts for correct predictions occurring by chance [8]. Kappa scores are 1 when classification labels exactly match the ground truth values, and 0 if the predictions were no more accurate than chance. A good kappa score for trained human judges rating emotion might be .5 [13] or .6 [8], while a typical score for a machine predicting emotion might be .3 [8] [16] [1].

Due to the small size of our dataset and the large number of features available, our classifiers will tend to over-fit the training data without an effective feature reduction method. We tested two techniques, Principal Component Analysis (PCA) and Wrapper Feature Selection (WFS), using 10-fold CV, and performing feature selection using only the training data. PCA reduces the dimension of a feature set by creating components based on highly correlated subsets of features [12]. WFS finds useful subsets of features by testing them with a specific classifier [16]. In order to obtain more robust feature sets with WFS, we performed nested cross validation, by further subdividing each training fold into another 10 train/test sets, performing wrapper selection on each, and using those features that were selected in more than 10% of the sub-folds. We found that WFS achieved better results overall, and that the features selected are more interpretable than PCA components. For these reasons, we focus on WFS when reporting results in the rest of the paper.

6 Results

In this section we present the results of several classification experiments. We begin by training classifiers using all available self-reports and gaze features computed using various time intervals preceding each report. We discuss the features chosen as most predictive by WFS, and the effectiveness of predicting reports independently.

6.1 Predicting Self-reports across the Interaction

Our first experiment involved training classifiers to predict the affective labels derived from any self-report, regardless of when it was generated. We wished to determine the amount of gaze data preceding the self-report that should be used for prediction. Many studies make use of a window of 20 seconds for affect labeling [14]. In a study of the same dataset, Harley et al., [15] used a 10 second window. We tested window lengths ranging from 100% of the available data (14 minutes) to 1% (8 seconds), and the results are shown in Fig. 2.

We used a 4 (classifier) x 6 (window length) General Linear Model (GLM) to analyze the results, treating the score obtained for one train/test split as a single data point. We ran four of these models, one with each of boredom accuracy, boredom kappa, curiosity accuracy, and curiosity kappa as the dependent variables, and applied Bonferroni corrections to adjust for family-wise error. In cases where the accuracy and kappa results are analogous, we present only the accuracy results. We compare the results to a majority-class baseline using t-tests with a Bonferroni adjustment.

The GLM results for both emotions were similar; there were no significant effects of classifier or interaction effects, likely because we have already restricted our focus

to the best classifiers. There was, however, a significant main effect of window length for both boredom, $F(5,216) = 8.390$, $\eta^2 = .163$, $p < .001$, and curiosity, $F(5,216) = 7.382$, $\eta^2 = .146$, $p < .001$. The curiosity results significantly exceeded the baseline at a window of 100% or 14 minutes ($M = 63.45\%$, $SD = 1.03$), $t(3) = 7.12$, $p < .05$. For boredom it was a window of 100%, ($M = 55.79\%$, SD = 1.95), $t(3) = 3.92$, $p < .05$, and a window of 75% or 10.5 minutes ($M = 57.83\%$, $SD = 1.35$), $t(3) = 8.29$, $p < .01$. Although certain classifiers (like RF) can still achieve good performance with a small interval of data, in general it seems that more gaze data generates better results. A large body of previous affect prediction research has focused on using a 20 second interval for affect labeling [14]. This study provides empirical evidence that this type of interval may not always be appropriate, depending on the data used for prediction.

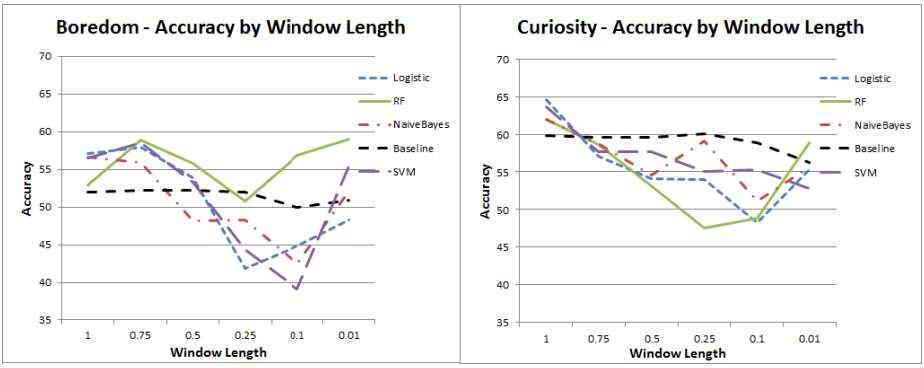


Fig. 2. Accuracy as a function of the fraction of interaction time used for training classifiers

6.2 Important Features

In this section we examine the features that were selected by the WFS process. We focus on the features selected most frequently for the windows that achieved the best performance, reasoning that these features must have been the most informative.

The general trends that seem to have emerged are depicted in Fig. 3, where arrows indicate gaze transitions and circles indicate features related to the AOI itself (circle size increases with the number of these features found). One trend is that students who are engaged (curious and/or not bored) make frequent use of the table of contents (TOC). This is evidenced by the fact that increased fixation length in the TOC and more TOC-to-TOC transfers are predictive of curiosity, while lower TOC fixation rate and fewer OLG-to-TOC and TOC-to-LSP transfers indicate boredom.

Engagement also appears to be linked to use of the image and Overall Learning Goal (OLG) AOIs. For example, bored students spend a smaller proportion of time and number of fixations on the image, and have fewer image-to-image and text-to-image transfers. They also have a shorter maximum fixation on the OLG, and fewer image-to-OLG, text-to-OLG, and OLG-to-subgoals transfers.

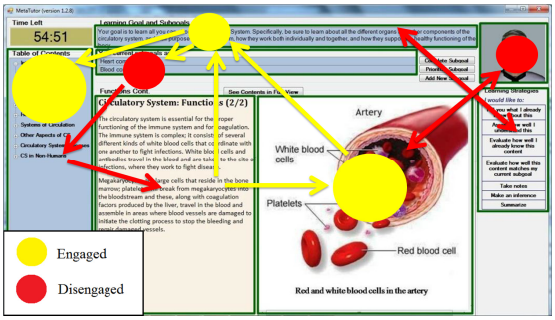


Fig. 3. Depiction of gaze trends for engaged and disengaged students

While the features listed so far provide evidence that engaged students look between the text, image, OLG, and TOC in a way that may suggest strategic learning, disengaged students seem to have frequent, scattered gaze transitions, without remaining focused on a single AOI. This is evidenced by subgoals-to-TOC, TOC-to-text, OLG-to-LSP and LSP-to-OLG transitions all being predictive of boredom or a lack of curiosity, while with the exception of frequent fixations on the subgoals, no features related to prolonged attention to the remaining AOIs were found.

Finally, attention to the agent may be associated with disengagement. Curious students fixate for a shorter time on the Agent, and have fewer Agent-to-image and image-to-Agent transfers. This is especially interesting given findings from [6] on the same dataset, which showed that the Agent was the only AOI not predictive of learning gains.

6.3 Time-Dependent Effects on Prediction

In our initial tests, the data from all self-reports was collected together and used in the training set with no indication of the point during the interaction when the report occurred. In the following tests we treat each self-report time as its own classification problem, to see if this timing information can improve performance. We use a full 14-minute window for prediction, based on findings from the previous section.

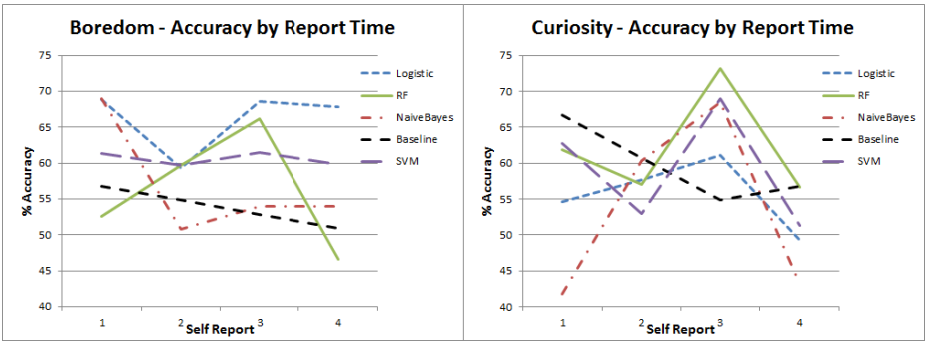


Fig. 4. Accuracy as a function of the self-report time

We conducted a similar 4 (classifier) x 4 (report time) General Linear Model on the results, which are shown in Fig. 4. For boredom, there were no significant effects, although on average the classifiers significantly exceeded the baseline, $t(163) = 2.68$, $p < .01$, with Logistic Regression achieving the highest average of 66.17% (kappa = .306), and peaking at report 1 (68.83%, kappa = .330). For curiosity, we found a main effect of report time for both kappa and accuracy, $F(3,144) = 5.953$, $\eta^2 = .110$, $p < .005$. This suggests that the time of the self-report, which corresponds to the amount of time a student has been interacting with MetaTutor, strongly affects the relationship between gaze and affect. Tukey post-hoc analysis revealed that self-report 3 ($M = 67.96$, $SD = 15.55$) was significantly better than all other reports. It was also the only report in which the classifiers significantly surpassed the baseline, $t(39) = 5.313$, $p < .001$, with Random Forests achieving a peak accuracy of 73.17% (kappa = .416).

Note that in addition to the effect of report time detected for curiosity, the average results obtained for boredom by restricting focus to a single self-report were also markedly higher than those obtained when all report times are classified together, as in the previous section. Overall, the results of this section seem to indicate that the relationship between gaze and affect varies over time. If this were true, we would expect that different gaze features would be more informative at different report times. Indeed, we examined the features chosen by WFS for each report, and found that there was considerable variability. Fig. 5 groups features into categories based on their AOI, and shows how the relevant features change along with time spent with MetaTutor. For example, the subgoals become highly relevant for predicting curiosity at report three, but otherwise are hardly chosen at all. We are not certain of the cause of this effect, however the changing importance of the features demonstrates that different patterns of behavior are indicative of the emotions over time.

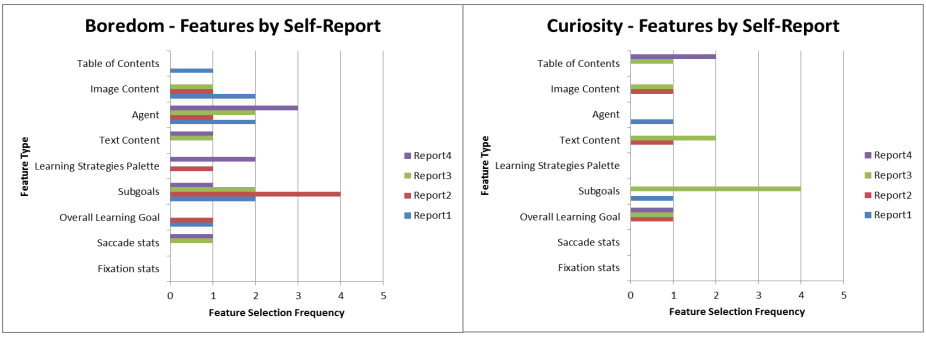


Fig. 5. The features found to be most predictive by wrapper feature selection change, depending on progress through MetaTutor

7 Conclusions and Future Work

The findings from this study demonstrate that eye gaze data alone is a useful tool for predicting boredom and curiosity in MetaTutor. The best results obtained, 69% (kappa = .33) for boredom and 73% (kappa = .42) for curiosity, are notable in the field of affect prediction, where near-perfect results are not the reality [13], and achieving

higher accuracies often requires combining multiple sources of user information [24]. We also present empirical evidence to contradict the assumption that a short interval of a few seconds is always most appropriate when predicting affect. Finally, we have found that temporal information about a students' progress through MetaTutor can lead to increased accuracy, so the relationship between gaze and affect in MetaTutor may be dependent on timing.

In the future we plan to leverage additional data sources collected during the MetaTutor study in order to predict affect, such as Electrodermal Activity (EDA), since it is related to emotional arousal [3]. Once we are able to reliably detect student affect, we can leverage this information in order to develop interventions that will help increase task success, engagement, and user satisfaction.

Acknowledgments. We would like to thank Daria Bondareva for her work on the data processing and validation portion of this project. This research was supported by NSERC, SSHRC, NSF, and Microsoft Research.

References

1. AlZoubi, O., D'Mello, S., Calvo, R.: Detecting naturalistic expressions of nonbasic affect using physiological signals (2012)
2. Anderson, J.R., Gluck, K.: What role do cognitive architectures play in intelligent tutoring systems. Cognition & Instruction, 227–262 (2001)
3. Arroyo, I., Cooper, D.G., Burleson, W., Woolf, B.P., Muldner, K., Christopherson, R.: Emotion sensors go to school. In: AIED, July 6-10, pp. 17–24. IOS Press, Brighton (2009)
4. Azevedo, R., Harley, J., Trevors, G., et al.: Using trace data to examine the complex roles of cognitive, metacognitive, and emotional self-regulatory processes during learning with multi-agent systems. In: IHMLT, pp. 427–449. Springer (2013)
5. Azevedo, R., Johnson, A., Chauncey, A., Burkett, C.: Self-regulated learning with MetaTutor: Advancing the science of learning with MetaCognitive tools. In: New Science of Learning, pp. 225–247. Springer (2010)
6. Bondareva, D., Conati, C., Feyzi-Behnagh, R., Harley, J.M., Azevedo, R., Bouchet, F.: Inferring Learning from Gaze Data during Interaction with an Environment to Support Self-Regulated Learning. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 229–238. Springer, Heidelberg (2013)
7. Conati, C., Maclaren, H.: Empirically building and evaluating a probabilistic model of user affect. UMUAI 19(3), 267–303 (2009)
8. D'Mello, S., Graesser, A.: Mind and body: Dialogue and posture for affect detection in learning environments. FAIA 158, 161 (2007)
9. D'Mello, S., Graesser, A., Picard, R.W.: Toward an affect-sensitive AutoTutor. IEEE Intelligent Systems 22(4), 53–61 (2007)
10. D'Mello, S., Olney, A., Williams, C., Hays, P.: Gaze tutor: A gaze-reactive intelligent tutoring system. IJHCS 70(5), 377–398 (2012)
11. D'mello, S., Craig, S., Gholson, B., Franklin, S., Picard, R., Graesser, A.: Integrating affect sensors in an intelligent tutoring system. Affective Interactions, 7–13 (2005)
12. Field, A.: Discovering statistics using SPSS. Sage publications (2009)

13. Forbes-Riley, K., Litman, D., Friedberg, H., Drummond, J.: Intrinsic and extrinsic evaluation of an automatic user disengagement detector for an uncertainty-adaptive spoken dialogue system. In: NAACL: Human Language Technologies, pp. 91–102 (2012)
14. Gutica, Conati: Student Emotions with an Edu-Game: A Detailed Analysis (2013)
15. Harley, J.M., Bouchet, F., Azevedo, R.: Aligning and Comparing Data on Emotions Experienced during Learning with MetaTutor. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 61–70. Springer, Heidelberg (2013)
16. Hussain, M.S., Calvo, R.A.: Multimodal affect detection from physiological and facial features during ITS interaction. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 472–474. Springer, Heidelberg (2011)
17. Kardan, S., Conati, C.: Exploring gaze data for determining user learning with an interactive simulation. In: Masthoff, J., Mobasher, B., Desmarais, M.C., Nkambou, R. (eds.) UMAP 2012. LNCS, vol. 7379, pp. 126–138. Springer, Heidelberg (2012)
18. Klein, J., Moon, Y., Picard, R.W.: This computer responds to user frustration: Theory, design, and results. Interacting with Computers 14(2), 119–140 (2002)
19. Kort, B., Reilly, R., Mostow, J., Picard, R.: Experimentally augmenting an intelligent tutoring system with human-supplied capabilities: Adding human-provided emotional scaffolding to an automated reading tutor that listens. In: ICMI, p. 483 (2002)
20. Muldner, K., Christopherson, R., Atkinson, R., Burleson, W.: Investigating the utility of eye-tracking information on affect and reasoning for user modeling. In: Houben, G.-J., McCalla, G., Pianesi, F., Zancanaro, M. (eds.) UMAP 2009. LNCS, vol. 5535, pp. 138–149. Springer, Heidelberg (2009)
21. Pekrun, R., Goetz, T., Titz, W., Perry, R.P.: Academic emotions in students' self-regulated learning and achievement: A program of qualitative and quantitative research. Educational Psychologist 37(2), 91–105 (2002)
22. Pekrun, R.: Emotions as drivers of learning and cognitive development. In: New Perspectives on Affect and Learning Technologies, pp. 23–39. Springer (2011)
23. Prendinger, H., Ishizuka, M.: The empathic companion: A character-based interface that addresses users'affective states. APAI 19(3-4), 267–285 (2005)
24. Sabourin, J., Mott, B., Lester, J.C.: Modeling learner affect with theoretically grounded dynamic bayesian networks. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, pp. 286–295. Springer, Heidelberg (2011)
25. Sabourin, J., Shores, L., Mott, B., Lester, J.: Predicting student self-regulation strategies in game-based learning environments, 141–150 (2012)
26. Sibert, J.L., Gokturk, M., Lavine, R.A.: The reading assistant: Eye gaze triggered auditory prompting for reading remediation, pp. 101–107. ACM Press (2000)
27. Silvia, P.J.: Interest—The curious emotion. Current Directions in Psychological Science 17(1), 57–60 (2008)
28. Smilek, D., Carriere, J.S., Cheyne, J.A.: Out of Mind, Out of Sight Eye Blinking as Indicator and Embodiment of Mind Wandering. Psych. Sci. 21(6), 786–789 (2010)
29. Wang, H., Chignell, M., Ishizuka, M.: Empathic tutoring software agents using real-time eye tracking. ETRA, 73–78 (2006)
30. Wang, N., Johnson, W.L., Mayer, R.E., Rizzo, P., Shaw, E., Collins, H.: The politeness effect: Pedagogical agents and learning outcomes. IJHCS 66(2), 98–112 (2008)
31. Wang, W., Li, Z., Wang, Y., Chen, F.: Indexing cognitive workload based on pupillary response under luminance and emotional changes. IUI, 247–256 (2013)
32. Zimmerman, B.J.: Self-efficacy: An essential motive to learn. Contemporary Educational Psychology 25(1), 82–91 (2000)

It's Written on Your Face: Detecting Affective States from Facial Expressions while Learning Computer Programming

Nigel Bosch[1], Yuxuan Chen[1], and Sidney D'Mello[1,2]

[1] Departments of Computer Science
[2] Psychology, University of Notre Dame, IN 46556, USA
{pbosch1,ychen18,sdmello}@nd.edu

Abstract. We built detectors capable of automatically recognizing affective states of novice computer programmers from student-annotated videos of their faces recorded during an introductory programming tutoring session. We used the Computer Expression Recognition Toolbox (CERT) to track facial features based on the Facial Action Coding System, and machine learning techniques to build classification models. Confusion/Uncertainty and Frustration were distinguished from all other affective states in a student-independent fashion at levels above chance (Cohen's kappa = .22 and .23, respectively), but detection accuracies for Boredom, Flow/Engagement, and Neutral were lower (kappas = .04, .11, and .07). We discuss the differences between detection of spontaneous versus fixed (polled) judgments as well as the features used in the models.

1 Introduction

Learning computer programming is an early obstacle for students pursuing a computer science (CS) degree [1]. The difficult nature of computer programming and lack of prior knowledge of novice students can create a particularly frustrating and confusing learning experience. One of the strategies that can be adopted to help with the burden of effectively teaching a large number of novice students is the use of intelligent tutoring systems (ITSs). As has been seen in other domains like computer literacy [2], it is likely that incorporating awareness of student affect into a computer programming ITS would lead to increased proficiency, particularly for novice students. Of course, an affect-aware ITS can never respond to affect if it cannot detect affect. We demonstrate a method for detecting the affect of novice programming students in a computerized learning environment using videos of students' faces.

Related Work. Affect detection can be done using various types of data sources, such as interaction data, speech, and physiology [3]. Facial-feature based affect detection is attractive because there is a strong link between facial features and affective states [4], it is more independent of learning environment or content (compared to interaction features), and it does not require expensive hardware, as webcams are ubiquitous on laptops and mobile devices.

In previous research on affect detection from facial features, Kapoor et al. [5] used multimodal data channels including facial features from video to predict Frustration in

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 39–44, 2014.

an automated learning companion. They were able to predict when a user would self-report Frustration with 79% accuracy (chance being 58%). Hoque et al. [6] used facial features and temporal information in videos to classify smiles as either frustrated or delighted. They were able to accurately distinguish between Frustrated and Delighted smiles correctly in 92% of cases. They also found differences between posed (acted) facial expressions and naturally induced facial expressions. Only 10% of Frustrated cases included a smile in acted data, whereas smiles were present in 90% of cases of naturally occurring Frustration.

The Computer Expression Recognition Toolbox (CERT) [7] is a computer vision tool used for automatic detection of 19 Action Units (AUs, codes describing specific facial muscle activations) as well as head pose and position information. It also supplies measures of three unilateral (one side of the face only) AUs, as well as "Fear Brow" and "Distress Brow," which indicate the presence of combinations of AU1 (Inner Brow Raiser), AU2 (Outer Brow Raiser), and AU4 (Brow Lowerer). CERT has been tested with databases of both posed facial expressions and spontaneous facial expressions, achieving accuracies of 90.1% and 79.9% respectively when discriminating between video frames with the presence vs. absence of particular AUs.

Whitehill et al. [8] have used CERT to detect engagement in a learning session. They obtained fine-grained judgments of engagement from external observers and achieved an accuracy of 71.8% when classifying instances of engagement vs. no engagement using AUs detected by CERT. Additionally, they found a correlation (r = .42) between fine-grained difficulty self-reports from students and AUs detected by CERT.

Grafsgaard et al. [9] used CERT to detect the overall level of Frustration (self-reported on a Likert scale) present in a learning session with modest results (R^2 = .24). Additionally, they have achieved good agreement between the output of CERT AU recognition and human-coded ground truth measurements of AUs (Cohen's kappa >= .68 for several key AUs), thereby providing additional evidence of the validity of CERT for automated AU detection.

Current Approach. This paper differs from the previous work in that our detectors will be applied at a finer granularity (15-second intervals), recognizing instances of affective states within a learning session rather than the level reported for the entire session. Additionally, the affective states we track are predominately learning-centered rather than the more commonly detected basic emotions (e.g. anger, sadness). Confusion is especially unique in that to our knowledge automatic confusion detection has not previously been done at a fine-grained level using facial features.

The facial expressions in the present study are naturalistic expressions, which have been shown to be more difficult to detect than posed expressions [10]. Despite the difficulty, we propose that facial features can be an effective method of automatically distinguishing particular affective states others in the domain of computer programming. To explore this we will answer these research questions: 1) Which affective states can be detected? 2) Can detection be improved by considering the type of affect judgment that is made? 3) Which features are most useful for detecting affective states?

2 Method

Data Collection. The data was collected from 99 computer programming novices who used a computerized learning environment designed to teach the basic elements of computer programming in the Python language. After the learning session, students viewed synchronized videos of their face and on-screen activity that had been recorded during the learning session, and retrospectively self-reported affective states at fixed points in the learning session. These periods were chosen to correspond with interaction events, such as typing code or viewing a new exercise, as well as idle periods (periods of time with no interaction events for more than 15 seconds). Students were also allowed to make spontaneous affect judgments at any point in the retrospective affect judgment process if they chose to. This retrospective affect judgment protocol allows for judgments to be made on the basis of a combination of the students' facial expressions, contextual cues (via screen capture), and their memories of the learning session [11]. We found that five affective states, namely Boredom (9%), Confusion (21%), Flow/Engagement (24%), Frustration (12%), and Neutral (17%), formed 83% of the affective states reported, so we focused on detecting these states (see [12] for more comprehensive details on data collection methodology used in the current study).

Computing Facial Features. We used CERT to calculate occurrence likelihoods for AUs in each video frame. The output of CERT was z-standardized within students and temporally aligned with affect judgments, then divided into segments of variable length (see below), each leading up to each affect judgment. Features were calculated by aggregating frame-level AU likelihoods across each segment using the median, maximum, and standard deviation of the 19 AUs provided by CERT. Head orientation and nose position were included as well. We also used HAAR cascades to detect the size of the face and the visibility of nose, mouth, eyes, and ears to provide additional information for situations with unusual pose or occlusion. We eliminated features exhibiting multicollinearity (variance inflation factor > 5).

Supervised Classification. We used the segment-level aggregate features to build classification models with the Waikato Environment for Knowledge Analysis (WEKA), a popular machine learning tool. Leave several out student-level cross-validation was used for model validation, with data from 66% of students randomly chosen to train classifiers and the remaining data used to test the performance of the classifiers. This ensures that the models generalize to new students since training and testing data sets are independent. The models were each trained and tested over 50 iterations with random students chosen each time to amortize random sampling error.

RELIEF-F feature ranking was used on the training data for each of the 50 iterations in order to identify the most diagnostic features prior to classification. We used 15 different classifiers and 6 different video segment sizes (2, 3, 6, 9, 12, and 15 seconds) to determine which segment size was likely to work best for a particular classification task. We then attempted to improve each of the best data configurations by either oversampling the training data (with SMOTE [13]) or downsampling the training data to equal class proportions.

3 Results and Discussion

Question 1: Which affective states can be detected? We attempted to individually classify each of the five most common affective states compared to all other affective states combined ("Other", which includes rare affective states). Cohen's kappa was used as the primary measure of performance, because it is more robust to class imbalances. Kappa measures the agreement between predicted and actual labels compared to chance (kappa = 0), with 1 reflecting perfect detection.

Classification of affective states from fixed affect ratings was not very successful. Using only fixed affect judgments, we had 6000 instances to be split into training and testing sets. Flow/Engagement was classified best (kappa = .112), with Boredom, Confusion, Frustration, and Neutral (kappas = .038, .064, .083, .070) classifications barely above chance. Classification of these data was expected to be difficult because they were selected at fixed points that were mostly independent of any facial activity. To improve the efficacy of classifiers we built models made using the spontaneous affect judgments, as discussed next.

Question 2: Can detection be improved by considering the type of affect judgment that is made? Because spontaneous affect judgments come from points in time of the student's choice, they may represent noticeable facial features in the video streams, as previously documented [14]. These judgments would likely make a more viable task for facial-feature based affect detection than the fixed judgments.

Only Confusion and Frustration had at least 100 spontaneous affect ratings, so we only consider those two states for further analysis. For the "Other" affective states, we sampled randomly from the fixed affective state judgments (5 times for each of the 50 iterations). Spontaneous Confusion judgments thus composed 21% of 582 instances with the other affective states in the fixed distribution as well, while Frustration composed 12% of 527 instances.

Using spontaneous judgments in this manner we were able to detect Confusion and Frustration much more effectively (kappa = .221 and .232, respectively). A simple logistic classifier yielded the best model for detecting Confusion, while an updatable naïve Bayes classifier was the most effective for Frustration. The best segment size for both of these was short (2 seconds for Confusion, 3 seconds for Frustration). Feature selection was used to select 50% of features for Confusion and the best 25% for Frustration detection. A more detailed look at the performance of these classification models can be found by examining the confusion matrices in Table 1.

Table 1. Confusion matrix for Confusion and Frustration spontaneous judgments

	Predicted Confusion	**Predicted Other**	**Priors**
Actual Confusion	0.50 (hit)	0.50 (miss)	0.21
Actual Other	0.25 (false alarm)	0.75 (correct rejection)	0.79
	Predicted Frustration	**Predicted Other**	
Actual Frustration	0.40 (hit)	0.60 (miss)	0.12
Actual Other	0.13 (false alarm)	0.87 (correct rejection)	0.88

Both models were impressive in terms of the low false alarm rate (i.e., Other affective states incorrectly detected as Confusion or Frustration). These models accurately detected half of the Confusion instances and nearly half of the Frustration instances, while properly rejecting most of the Other affective states, despite class imbalances.

Question 3: Which features were most useful for detecting affective states? We examined the features that were automatically selected for the spontaneous affect judgment classifiers. While both classification tasks used AU45 (Blink) features frequently, the other features differed between Confusion and Frustration. Particularly notable were the presence of unilateral (one side of the face) features for Frustration detection as well as head pose features (Yaw). Distress Brow, which appears frequently for Confusion detection, indicates evidence for AU1 (Inner brow raiser) or a combination of AU1 and AU4 (Brow lowerer). This feature has been found to be predictive in prior research involving manual coding of AUs as well [4]. Additionally, it appears as though Confusion was manifested more in absolute values of facial features, while Frustration was more easily detected from standard deviation features.

4 General Discussion

Despite the complexities of affect detection of naturally occurring learning-centered states, we were able to achieve some success in building fully-automated facial-feature based detectors of spontaneous confusion and frustration in a manner that generalizes to new students. Our current detection accuracy is modest at best, but affect detection is inherently an imperfect science and current detection rates are comparable with what is achieved for automatic detection of naturalistic affect from alternate modalities in a student-independent fashion [10].

Results for spontaneous judgments were much improved over the fixed judgments, as was expected. A similar phenomenon occurs when examining the inter-rater reliability between human judges manually coding emotions or AUs in video [14]. These results suggest that future work collecting video data for building affect detectors might be better served by focusing only on spontaneous affect judgments.

Some limitations of this study include (1) the relative infrequency of spontaneous judgments, (2) the relatively small sample size, and (3) lack of generalizability of results beyond the current sample. In future work we plan to train detectors using interaction data from the students' learning sessions, and incorporate those detectors with video-based detectors to create a more powerful multimodal affect classifier. We hope that accurate affect detection for novice computer programmers will lead to more effective computerized learning environments capable of responding to the momentary affective episodes of students so they may learn to their fullest potential.

Acknowledgment. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958) and the Bill & Melinda Gates Foundation. Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. Rodrigo, M.M.T., Baker, R.S.J.d., Jadud, M.C., Amarra, A.C.M., Dy, T., Espejo-Lahoz, M.B.V., Lim, S.A.L., Pascua, S.A.M.S., Sugay, J.O., Tabanao, E.S.: Affective and behavioral predictors of novice programmer achievement. SIGCSE Bulletin 41, 156–160 (2009)
2. D'Mello, S., Lehman, B., Sullins, J., Daigle, R., Combs, R., Vogt, K., Perkins, L., Graesser, A.: A time for emoting: When affect-sensitivity is and isn't effective at promoting deep learning. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 245–254. Springer, Heidelberg (2010)
3. Zeng, Z., Pantic, M., Roisman, G.I., Huang, T.S.: A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence 31, 39–58 (2009)
4. McDaniel, B.T., D'Mello, S.K., King, B.G., Chipman, P., Tapp, K., Graesser, A.C.: Facial features for affective state detection in learning environments. In: Proceedings of the 29th Annual Cognitive Science Society, pp. 467–472 (2007)
5. Kapoor, A., Burleson, W., Picard, R.W.: Automatic prediction of frustration. International Journal of Human-Computer Studies 65, 724–736 (2007)
6. Hoque, M.E., McDuff, D.J., Picard, R.W.: Exploring Temporal Patterns in Classifying Frustrated and Delighted Smiles. IEEE Transactions on Affective Computing 3, 323–334 (2012)
7. Littlewort, G., Whitehill, J., Wu, T., Fasel, I., Frank, M., Movellan, J., Bartlett, M.: The computer expression recognition toolbox (CERT). In: 2011 IEEE International Conference on Automatic Face Gesture Recognition and Workshops (FG 2011), pp. 298–305 (2011)
8. Whitehill, J.R.: A stochastic optimal control perspective on affect-sensitive teaching. PhD dissertation, University of California, San Diego (2012)
9. Grafsgaard, J.F., Wiggins, J.B., Boyer, K.E., Wiebe, E.N., Lester, J.C.: Automatically Recognizing Facial Indicators of Frustration: A Learning-Centric Analysis (2013)
10. D'Mello, S., Kory, J.: Consistent but modest: a meta-analysis on unimodal and multimodal affect detection accuracies from 30 studies. In: Proceedings of the 14th ACM International Conference on Multimodal Interaction, pp. 31–38. ACM, New York (2012)
11. D'Mello, S., Graesser, A., Picard, R.W.: Toward an affect-sensitive AutoTutor. IEEE Intelligent Systems 22, 53–61 (2007)
12. Bosch, N., D'Mello, S., Mills, C.: What Emotions Do Novices Experience during Their First Computer Programming Learning Session? In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 11–20. Springer, Heidelberg (2013)
13. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research 16, 321–357 (2011)
14. Graesser, A.C., McDaniel, B., Chipman, P., Witherspoon, A., D'Mello, S., Gholson, B.: Detection of emotions during learning with AutoTutor. In: Proceedings of the 28th Annual Meetings of the Cognitive Science Society, pp. 285–290 (2006)

Impact of Agent Role on Confusion Induction and Learning

Blair Lehman and Arthur Graesser

University of Memphis, Memphis, TN 38152, USA
{balehman,graesser}@memphis.edu

Abstract. The presentation of contradictory information to trigger deeper processing and increase learning has been investigated in a variety of ways (e.g., conversational agents, worked examples). However, the impact of information source (e.g., expertise, gender) and the relationship between the contradicting sources (e.g., status level) has not been investigated to the same degree. We previously reported that confusion can successfully be induced and learning increased when contradictory information was presented by two conversational agents (tutor, peer student). In the present experiment we investigated contradictions posed by two peer student agents. Self-reports of confusion and learner responses to embedded forced-choice questions revealed that the contradictions still successfully induced confusion. There were, however, differences in the nature of confusion induction based on the inter-agent relationship (i.e., student-student vs. tutor-student). Learners performed better on transfer tasks when presented with contradictions compared to a no-contradiction control, but only when they were successfully confused.

Keywords: confusion, contradiction, affect, tutoring, animated pedagogical agents, intelligent tutoring systems, learning.

1 Introduction

To understand a concept it is important to learn why a particular strategy or explanation is correct and why alternatives are incorrect. However, it is often difficult for learners to understand both aspects. One method to help learners reach this level of understanding is the presentation of contradictory information [1-5]. Contradictory information has been presented in a variety of contexts, such as conversational agents [1,3], sources within a text [6], and worked examples [2,4,5], to create cognitive conflict (see Limón [7] for review), cognitive disequilibrium [8-10], and confusion [1,3]. In all instances, the contradictory information is expected to increase learning by causing learners to stop, think, and deliberate over which alternative is correct in an effort to resolve their current cognitive and affective conflict.

There are two important considerations when presenting contradictions to increase learning. First, the contradiction must be highlighted such that learners are aware that there is a contradiction and that the two alternatives are not compatible (i.e., cannot both be correct) [7]. Unfortunately, learners often dismiss the contradiction and do not

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 45–54, 2014.

engage in the beneficial cognitive activities required to compare the competing alternatives and determine which one is correct. Learners can ignore the contradiction, reject or deny the validity of one alternative, exclude one alternative from the explanation of a concept, or reinterpret one alternative so that the two alternatives are no longer in conflict [11]. Thus, it is important to present contradictions that are salient to learners within a context that requires their resolution.

The second issue to consider when presenting contradictions to increase learning is the sources of the contradicting alternatives. Research on the presentation of contradictions within a text has found that contradictions actually draw more attention to the source of information [6]. Participants have been found to have more fixations and longer gaze times on the sources of information (e.g., person A vs. person B) while reading and increased citations of sources when writing summaries compared to when sources agreed. Attention to sources can lead to source evaluation, which has been found to increase comprehension [12-15]. In fact, learners who performed better on comprehension assessments were found to evaluate information sources more while reading than those who performed less well [13].

Contradictions have also been found to be an effective catalyst for deeper reasoning when presented by conversational agents. In a series of experiments, conversational agents presented contradictions during trialogues (i.e., three-party conversations) to induce confusion and promote learning [1,3]. One agent served as a tutor, whereas the other agent served as a peer student agent. Learners who were successfully confused by the contradictions performed significantly better on measures of learning and transfer tasks compared to when the agents agreed. However, the effectiveness of confusion induction was consistently found to differ depending on which agent was correct (i.e., tutor vs. student) when the agents disagreed. This finding raises the question as to how agent role (e.g., status, status differential, gender, etc.) impacts confusion induction and learning. Baylor and Kim [16] have indeed reported that agent roles in learning environments that do not pose contradictions can impact both motivation and learning.

The present research is an initial attempt to determine the impact of agent role when contradictions are presented. To completely address this question, research should examine confusion induction and learning when agent role differs (tutor, peer student) and is the same (peer student, tutor or expert) as well as when agent characteristics (e.g., gender, race, age) are varied. The present research replicates Lehman et al. [3], but with two peer student agents instead of a tutor agent and peer student agent. Three research questions are investigated in the present research. When contradictions are presented by two peer student agents, will confusion be successfully induced (question 1) and will learning increase (question 2)? Finally, the third research question will address the similarities and differences between confusion induction and learning outcomes when contradictions are presented by agents of different status (tutor, student, [3]) compared to agents of the same status (two students). The impact of agent role will be investigated within a learning environment that diagnoses flaws in research case studies to help learners better understand research methods concepts.

2 Methods

2.1 Manipulation

We experimentally induced confusion with a contradictory information manipulation over the course of learning research methods concepts (e.g., replication, control group, validity). This was achieved by having the two student agents (male student and female student, see Figure 1) stage a disagreement on an idea and eventually invite the human learner to intervene (note that student agent refers to the animated agents, the actual human learner is referred to as learner). This confusion induction method has been found to successfully induce confusion when contradictions were posed by tutor and student agents in previous experiments [1,3].

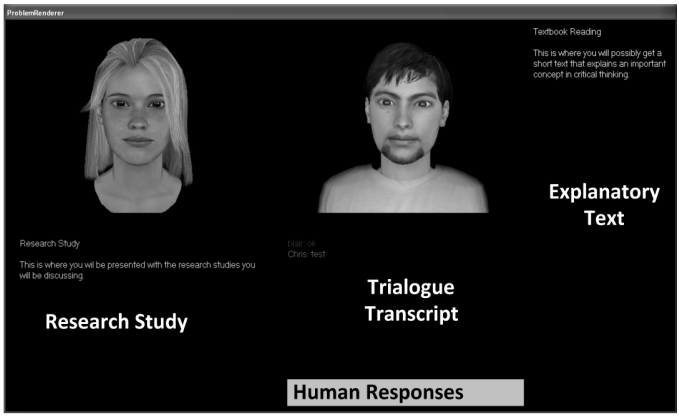


Fig. 1. Screenshot of learning environment interface

Contradictions were introduced during trialgoues (three-party conversations) identifying flaws in sample research studies. Some studies had subtle flaws while others were flawless. There were four contradictory information conditions. In the *true-true* condition, both student agents agreed and presented correct opinions. In the *true-false* condition, the female student agent presented a correct opinion and the male student agent disagreed by presenting an incorrect opinion. In contrast, the male agent presented a correct opinion and it was the female agent that disagreed with an incorrect opinion in the *false-true* condition. Finally, in the *false-false* condition, both agents agreed but the opinions that they presented were incorrect. It should be noted that all misleading information was corrected after learners completed all four trialogues and posttests and that learners were fully debriefed at the end of the experiment.

2.2 Participants and Design

Participants were 32 undergraduate students from a mid-south university in the US and participated for course credit. The experiment had a within-subjects design with four conditions (*true-true*, *true-false*, *false-true*, *false-false*). Learners completed one

trialogue in each of the four conditions with a different research methods topic in each session (4 in all). Order of conditions and topics and assignment of topics to conditions was counterbalanced across learners with a Graeco-Latin Square.

2.3 Procedure

The experiment occurred over two phases: (1) knowledge assessments and trialogues and (2) a retrospective affect judgment protocol.

Knowledge Tests. Research methods knowledge was assessed with flaw identification tasks before and after trialogues (pretest and posttest, respectively). The flaw identification tasks consisted of a description of a previously unseen study and learners were asked to identify flaw(s) in the study by selecting as many items as they wanted from a list of eight research methods topics. The list included four topics that could potentially be flawed (discussed in the trialogues) and four distractor topics (not discussed in the trialogues). Learners also had the option of selecting that there was no flaw, although each study contained one flaw. The pretest involved the presentation of four case studies that each contained one flaw. The flaw in each case study corresponded to one of the topics discussed in the trialogues.

The posttest consisted of both near and far transfer versions of the studies that were presented in the trialogues. The near transfer studies differed from the studies in the trialogues on surface features, whereas the far transfer studies differed on both surface and structural features. Each topic discussed during the trialogues had one near and one far transfer study, resulting in eight transfer studies in all on the posttest.

Trialogues. First, learners signed an informed consent and then completed the pretest. Learners then began the first of four trialogues. A webcam and a commercially available screen capture program (Camtasia Studio™) recorded learners' face and screen, respectively, during the trialogues.

Each trialogue began with a description of a study, which learners read and then began the discussion with the agents. The excerpt in Table 1 is an example trialogue. This is an excerpt from the *true-false* condition, where the female (Mary) and male (Chris) student agents are discussing a flawed study with Bob (learner). The discussion of each study involved five trials. For example, in Table 1 the dialogue turns 2 through 5 represent one trial. Each trial consisted of the student agents asserting their opinions (turns 2 and 3), prompting the learner to intervene with a forced-choice (FC) question (turn 4), and obtaining the learner's response (turn 5).

This cycle was repeated in each trial, with each trial becoming increasingly more specific about the scientific merits of the study. The trialogue in Table 1 discusses a study that uses an inappropriate control group. Trial 1 broadly asked if learners would change their behavior based on the results of the study ("Would you recommend the diet pill to a friend?"), while Trial 2 addressed whether or not there is a problem in the methodology of the study ("Do you think the methodology of the study was good or problematic?"). Trial 3 began to specifically address the problem in the study (turns 2-5). Trial 4 then directly addressed the appropriateness of the control group, (turns 6-9). After Trial 4, learners were presented with an explanatory text to potentially alleviate

their confusion, which was adapted from the electronic textbook that accompanies the *Operation ARA!* ITS [17]. Finally, Trial 5 repeated the forced-choice question posed to learners in Trial 4 without the presentation of the agents' opinions. For the present paper only Trials 1-4 are relevant and will be included in subsequent analyses. Learners then completed the posttest after discussing the four studies.

Table 1. Excerpt of trialogue from *true-false* condition

Turn	Speaker	Dialogue
Participants took this new diet pill and reported losing 10 pounds in the first month. None of the participants exercised or ate super healthy or anything, they just acted normally. The researchers even compared the pill to another group who didn't take it and just acted how they normally do. The group that didn't take the pill didn't lose any weight over a month. So the pill works about ten times as well as not doing anything. <*Case study*>		
1	Mary	Chris, are these two groups similar in every way except taking the pill or not? <*Advance dialogue*>
2	Chris	Umm, I think they were the same in every way. <*Assert opinion*>
3	Mary	Well, the two groups don't seem the same to me. <*Disagree with Chris*>
4	Chris	Looks like we still disagree. Bob were these two groups exactly the same or different? <*Contradiction & forced-choice question*>
5	Bob	Different <*Response*>
6	Chris	I don't think they need to make any changes to the control group. <*Assert opinions*>
7	Mary	No, I don't think so, it needs to change. <*Disagree with Chris*>
8	Mary	It's too bad that we still disagree. Bob, should the control group have taken some kind of a pill or should it stay the same? <*Contradiction & forced-choice question*>
9	Bob	stay same <*Response*>

Retrospective Affect Judgment Protocol. Learners completed a retrospective affect judgment protocol [18] after completing the posttest. Videos of learners' face and screen were synchronized and learners made affect ratings while viewing these videos. Learners were provided with a list of affective states (anxiety, boredom, confusion, curiosity, delight, engagement/flow, frustration, surprise, and neutral) with definitions. Affect judgments occurred at 14 pre-specified points (e.g., after contradiction presentation, after forced-choice question, after learner response) in each trialogue (56 in all). In addition to these pre-specified points, learners were able to manually pause the videos and provide judgments at any time.

3 Results and Discussion

The analyses were conducted in three phases: self-report confusion ratings, forced-choice (FC) question response accuracy, and transfer test performance. We conducted these analyses in order to determine the impact of agent role (tutor-student vs. student-student) on confusion induction and learning. The results from the current experiment were compared to previous findings from an experiment that involved trialogues with tutor and student agents (tutor-student experiment) [3]. Mixed-effects linear or logistic regression models were constructed for each dependent measure,

with one exception, to compare the experimental conditions (*true-false, false-true, false-false*) to the no-contradiction control condition (*true-true*).

3.1 Self-report Confusion Ratings

In the tutor-student experiment confusion was reported more often when learners were in the *true-false* and *false-false* conditions compared to the *true-true* condition. Confusion self-report ratings for the first four trials of each trialogue were investigated for the present experiment. A mixed-effects logistic regression revealed that in the student-student experiment, learners also reported more confusion in the *true-false* and *false-false* conditions than when in the *true-true* condition, $\chi^2(3) = 6.90$, $p = .038$. Table 2 shows the coefficients for the models along with the mean proportional occurrence of confusion. These findings suggest that confusion induction can still be successful when contradictions were presented by two peer student agents. It is interesting, however, that the same pattern of findings emerged when both agents had the same status level. Ostensibly, the contradiction in the *true-false* and *false-true* conditions should evoke the same degree of confusion, but this was not the case. This suggests that other characteristics of the agents may need to be taken into consideration (e.g., gender, perceived knowledge).

Table 2. Proportional occurrence of trialogue dependent measures

	Induction Condition				**Coefficient (B)**		
	Tr-Tr	*Tr-Fl*	*Fl-Tr*	*Fl-Fl*	*Tr-Fl*	*Fl-Tr*	*Fl-Fl*
Confusion Self-Report	.113	.159	.118	.150	**.490**	.221	**.421**
FC Question							
Trial 1	.688	.563	.500	.500	-.530	-.795	-.787
Trial 2	.844	.594	.656	.406	**-1.28**	**-1.05**	**-2.06**
Trial 3	.750	.563	.656	.406	**-.847**	-.452	**-1.48**
Trial 4	.656	.688	.719	.500	.126	.298	-.667

Notes. Tr: True; Fl: False; Tr-Tr was the reference group for each model, hence coefficients for this condition are not shown in the table. Bolded cells refer to significant effects at $p < .05$.

3.2 Forced-Choice Question Response Accuracy

Two analyses were conducted to investigate FC question response accuracy during trialogues. First, we constructed four mixed-effects logistic regressions to investigate response accuracy in each trial (see Table 2). In the tutor-student experiment, learners were less likely to respond correctly when in the experimental conditions as the trialogues became increasingly more specific (i.e., Trials 2-4) compared to the no-contradiction control condition. This reduction in correct responses is hypothesized to display confusion and uncertainty. A similar pattern emerged in the present student-student experiment with learners being less likely to respond correctly when in the experimental conditions compared to the no-contradiction control condition in Trials 2 ($\chi^2(3) = 13.6$, $p = .002$) and 3 ($\chi^2(3) = 8.66$, $p = .017$). The one exception was that the *false-true* condition did not differ from the *true-true* condition in Trial 3. Interestingly, when the trialogue specifically addressed the flaw in the study (Trial 4)

in the present experiment, the experimental conditions did not differ from the no-contradiction control condition, $\chi^2(3) = 3.95$, $p = .134$. Performance in Trial 4 was then the primary difference between the two experiments.

Second, we investigated response accuracy compared to random guessing (or chance) in each condition with the hypothesis that responses similar to random guessing would display confusion and uncertainty. Since the questions adopted a two-alternative format, random guessing would yield a score of .5. In the tutor-student experiment this analysis revealed the general pattern that *true-true* performed above chance and *false-false* performed below chance, whereas *true-false* and *false-true* generally remained at chance level. One-sample t-tests comparing learner responses to .5 (chance) revealed the following overall pattern: *true-true* and *false-true* were significantly greater than chance and *true-false* and *false-false* were statistically indistinguishable from chance. There were two exceptions to this pattern: (a) *true-false* was greater than chance on Trial 4 and (b) *false-true* was at chance level in Trial 1.

There are two overall differences when the patterns from the tutor-student and student-student experiments are compared. First, learner responses in the *false-false* condition were found to remain at chance level in the present experiment, suggesting that learners may have been more skeptical of incorrect agent opinions, even when the agents agreed. Second, learners responded above chance levels in the *false-true* condition. This is a somewhat perplexing finding given that responses in the *true-false* condition were generally still at chance level. Even though the agents had the same status level, there may have been other agent characteristics (e.g., gender, perceived knowledge) or trialogue characteristics (e.g., which agent stated their opinion first) that influenced learner responses.

3.3 Transfer Task Performance

Learner performance on both transfer tasks was assessed with hits (correctly identifying the presence of a flaw) to investigate learning. In the previous tutor-student experiment, performance on multiple-choice knowledge assessments was used to measure learning. The results from that experiment revealed that learners only benefited from the presentation of contradictions when they were successfully confused during the trialogues. Two analyses were conducted to investigate learning in the present student-student experiment.

First, mixed-effects logistic regressions revealed that there were not significant condition differences on either transfer task (Near Transfer: $\chi^2(3) = 4.95$, $p = .176$, Far Transfer: $\chi^2(3) = 1.41$, $p = .703$). This finding was consistent with the previous tutor-student experiment and is likely due to the fact that confusion induction success was not taken into consideration. The second analysis then involved dividing learners into low- and high-confusion cases based on a median split of self-report confusion ratings. Mixed-effects logistic regression models were constructed to investigate the induction condition × confusion (low, high) interaction (see Table 3). A significant model was found for the near transfer task ($\chi^2(7) = 11.1$, $p = .067$), but not for the far transfer task ($\chi^2(7) = 6.26$, $p = .255$). The main effect for confusion was not significant for either model (p's > .1).

The interaction was probed by regressing near transfer hits for the low- and high-confusion cases separately. The model for low-confusion cases was not significant,

$\chi^2(3) = .435$, $p = .467$. However, the model for the high-confusion cases was significant, $\chi^2(3) = 10.5$, $p = .008$. When learners were in the *true-false* and *false-true* conditions, they performed significantly better on the near transfer task than in the *true-true* condition. It is possible that this increased performance was actually due to increased guessing. To address this issue, we investigated false alarms (incorrectly identifying the presence of a flaw) for the near transfer case studies. The induction condition × confusion model for false alarms was not significant, so the learning effect cannot be attributed to guessing.

Despite the fact that different types of assessments were used (multiple-choice questions vs. transfer tasks), the findings in the present experiment are very similar to those in the tutor-student experiment. It appears to be critical that learners are successfully confused to benefit from the presentation of contradictory information.

Table 3. Proportional occurrence of transfer test performance

	Induction Condition				**Coefficient (B)**		
	Tr-Tr	*Tr-Fl*	*Fl-Tr*	*Fl-Fl*	*Tr-Fl*	*Fl-Tr*	*Fl-Fl*
Near Transfer							
Low Confusion	.571	.533	.471	.500	-.168	-.414	-.311
High Confusion	.273	.647	.600	.222	**1.84**	**1.60**	-.273
Far Transfer							
Low Confusion	.350	.200	.412	.385	-.800	.433	.092
High Confusion	.455	.500	.214	.500	.171	-.160	.244

Notes. Tr: True; Fl: False; Tr-Tr was the reference group for each model, hence coefficients for this condition are not shown in the table. Bolded cells refer to significant effects at $p < .05$.

4 Conclusion

Contradictory information has been used to increase learning with different methods of presentation (e.g., [1-5,7]). This strategy is expected to be effective because it creates a state of mental discomfort through occurrences of cognitive conflict [7], cognitive disequilibrium [8-10], and confusion [1,3], which then trigger learners to engage in effortful cognitive activities (e.g., reflection, problem solving) that ultimately bring about deeper comprehension [19-20]. The present experiment continues this line of research, but also addresses the less researched issue of the sources of contradictions. We have conducted an experiment that, when compared with the findings of a previous experiment [3], allows for the impact of source to be investigated.

Overall we have found that the presentation of contradictory information by two peer student agents can still successfully induce confusion and had a positive impact on learning. Findings for self-reported confusion mirrored the pattern when contradictions were presented by tutor and student agents [3]. The patterns differed, however, when response accuracy was investigated. This more objective measure of uncertainty and confusion indicated that learners were influenced by agent role and the inter-agent relationship in the trialogue. Although, it was the case that similar learning |patterns were found regardless of the inter-agent relationship. In both experiments learners performed better when in the contradictory information conditions

(*true-false*, *false-true*) when they were successfully confused. This finding across both experiments is consistent with impasse-driven theories of learning [20] and also cognitive conflict research (e.g., [7,21]) in which learners must be triggered through awareness of the conflict to begin engaging in the cognitive activities that benefit learning.

It was not the case, however, that the two conditions in which the agents disagreed (*true-false*, *false-true*) were identical in all respects in the present experiment. In particular, the *true-false* and *false-true* conditions differed on self-reported confusion and forced-choice question response accuracy. Given that the two agents had the same status level (peer student), it could be expected that similar patterns would emerge for both conditions. The findings for the *true-false* condition adhere to the expected pattern with increased self-reported confusion and response accuracy at chance level, whereas the *false-true* condition did not differ from the *true-true* condition on self-reported confusion and generally responded above chance level. This suggests that status level is not the only agent characteristic that should be considered and trialogue characteristics (e.g., which agent stated their opinion first) may need to be considered as well. For example, in the present experiment the agents differed on gender. When there is no clear authority figure or expert learners may align with an agent based on other characteristics. Research has suggested that agent gender and ethnicity in relation to learner gender and ethnicity can impact the learning experience (e.g., [16,22-23]). There were too few participants in the present sample to investigate these differences, but future research will need to investigate additional characteristics and how they impact the effectiveness of the presentation of contradictions to trigger deeper processing and ultimately have a positive impact on learning.

Acknowledgements. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. D'Mello, S., Lehman, B., Pekrun, R., Graesser, A.: Confusion can be beneficial for learning. Learning and Instruction 29, 153–170 (2014)
2. Grosse, C., Renkl, A.: Finding and fixing errors in worked examples: Can this foster learning outcomes? Learning and Instruction 17, 612–634 (2007)
3. Lehman, B., D'Mello, S., Strain, A., Mills, C., Gross, M., Dobbins, A., et al.: Inducting and tracking confusion with contradictions during complex learning. International Journal of Artificial Intelligence in Education 22, 71–93 (2013)
4. McLaren, B.M., et al.: To err is human, to explain and correct is divine: A study of interactive erroneous examples with middle school math students. In: Ravenscroft, A., Lindstaedt, S., Kloos, C.D., Hernández-Leo, D. (eds.) EC-TEL 2012. LNCS, vol. 7563, pp. 222–235. Springer, Heidelberg (2012)
5. Tsovaltzi, D., Melis, E., McLaren, B.: Erroneous examples: Effects on learning fractions in a web-based setting. International Journal of Technology Enhanced Learning 4, 191–230 (2012)

6. Braasch, J., Rouet, J.-F., Vibert, N., Britt, M.: Readers' use of source information in text comprehension. Memory & Cognition 40, 450–465 (2012)
7. Limón, M.: On the cognitive conflict as an instructional strategy for conceptual change: A critical appraisal. Learning and Instruction 11, 357–380 (2001)
8. Festinger, L.: A theory of cognitive dissonance. Row Peterson, Evanston (1957)
9. Graesser, A., Lu, S., Olde, B., Cooper-Pye, E., Whitten, S.: Question asking and eye tracking during cognitive disequilibrium: Comprehending illustrated texts on devices when devices breakdown. Memory & Cognition 33, 1235–1247 (2005)
10. Piaget, J.: The origins of intelligence. International University Press, New York (1952)
11. Chinn, C., Brewer, W.: An empirical test of a taxonomy of responses to anomalous data in science. Journal of Research in Science Teaching 35, 623–654 (1998)
12. Bråten, I., Strømsø, H., Britt, M.: Trust matters: Examining the role of source evaluation in students' construction of meaning within and across multiple texts. Reading Research Quarterly 44, 6–28 (2009)
13. Goldman, S., Braasch, J., Wiley, J., Graesser, A., Brodowinska, K.: Comprehending and learning from internet sources: Processing patterns of better and poorer learners. Reading Research Quarterly 47, 356–381 (2012)
14. Strømsø, H., Bråten, I., Britt, M.: Reading multiple texts about climate change: The relationship between memory for sources and text comprehension. Learning and Instruction 20, 192–204 (2010)
15. Wiley, J., Goldman, S., Graesser, A., Sanchez, C., Ash, I., Hemmerich, J.: Source evaluation, comprehension, and learning in internet science inquiry tasks. American Educational Research Journal 46, 1060–1106 (2009)
16. Baylor, A., Kim, Y.: Simulating instructional roles through pedagogical agents. International Journal of Artificial Intelligence in Education 15, 95–115 (2005)
17. Halpern, D., Millis, K., Graesser, A., Butler, H., Forsyth, C., Cai, Z.: Operation ARA: A computerized learning game that teaches critical thinking and scientific reasoning. Thinking Skills and Creativity 7, 93–100 (2012)
18. Graesser, A., D'Mello, S.: Emotions during the learning of difficult material. In: Ross, B. (ed.) The Psychology of Learning and Motivation, vol. 57, pp. 183–225. Elsevier (2012)
19. D'Mello, S., Graesser, A.: Dynamics of affective states during complex learning. Learning and Instruction 22, 145–157 (2012)
20. VanLehn, K., Siler, S., Murray, C., Yamauchi, T., Baggett, W.: Why do only some events cause learning during human tutoring? Cognition & Instruction 21, 209–249 (2003)
21. Chan, C., Burtis, J., Bereiter, C.: Knowledge building as a mediator of conflict in conceptual change. Cognition and Instruction 15, 1–40 (1997)
22. Baylor, A., Kim, Y.: The role of gender and ethnicity in pedagogical agent perception. In: Richards, G. (ed.) Proceedings of the World Conference on E-learning in Corporate, Government, Healthcare, and Higher Education, pp. 1503–1506. AACE, Chesapeake (2003)
23. Moreno, R., Flowerday, T.: Students' choice of animated pedagogical agents in science learning: A test of the similarity-attraction hypothesis on gender and ethnicity. Contemporary Educational Psychology 31, 186–207 (2005)

Automated Physiological-Based Detection of Mind Wandering during Learning

Nathaniel Blanchard[1], Robert Bixler[1], Tera Joyce[1], and Sidney D'Mello[1,2]

[1] Department of Computer Science
[2] Department of Psychology, University of Notre Dame, Notre Dame, IN 46556
{nblancha,rbixler,tjoyce4,sdmello}@nd.edu

Abstract. Unintentional lapses of attention, or mind wandering, are ubiquitous and detrimental during learning. Hence, automated methods that detect and combat mind wandering might be beneficial to learning. As an initial step in this direction, we propose to detect mind wandering by monitoring physiological measures of skin conductance and skin temperature. We conducted a study in which student's physiology signals were measured while they learned topics in research methods from instructional texts. Momentary self-reports of mind wandering were collected with standard probe-based methods. We computed features from the physiological signals in windows leading up to the probes and trained supervised classification models to detect mind wandering. We obtained a kappa, a measurement of accuracy corrected for random guessing, of .22, signaling feasibility of detecting MW in a student-independent manner. Though modest, we consider this result to be an important step towards fully-automated unobtrusive detection of mind wandering during learning.

Keywords: skin conductance, skin temperature, mind wandering, machine learning.

1 Introduction

Almost everyone has had the experience of attempting to concentrate on a learning task and suddenly realizing that their mind has drifted elsewhere. As a result they may have missed key pieces of information and are forced to review the missed material. This phenomenon, called mind wandering (MW), can be described as involuntarily engaging in conscious off-task thoughts without the metacognitive realization that this has occurred [1]. MW has been linked to lower performance on a number of tasks including poor comprehension during reading [2] and low recall during memory encoding [3]. Furthermore, MW is difficult to address immediately because people initially lack conscious awareness of that fact that they are MW. Given the ubiquity and negative consequences of the phenomenon, it might be beneficial for intelligent tutoring systems (ITSs) and other educational technologies to detect when MW occurs and then intervene to restore attention to the task at hand. As an initial step in this direction, this paper reports research aimed at developing a fully-automated system to detect momentary occurrences of MW in a manner that generalizes to new students.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 55–60, 2014.

Related Work. MW detection is a relatively unexplored field. Drummond and Litman (2010) were one of the first to attempt automatic MW detection. They used prosodic and lexical features of student responses to a spoken ITS. Students were probed at set intervals into if they were MW. Their models were able to discriminate high and low MW with an accuracy of 64%. However, their models were only applicable to ITSs with student speech, and their validation method did not ensure generalization to new students [4].

D'Mello, Cobian, and Hunter (2013) furthered work on MW detection by building supervised classification models that automatically detected MW during reading from eye gaze features obtained with commercial eye trackers. Their model obtained a kappa, a measurement of accuracy corrected for chance, of 0.23 [5]. Though their validation method ensured generalizability to new students, their approach is limited to reading tasks. Furthermore, the use of eye tracking has some scalability concerns.

Current Study. The present study focused on detecting MW by monitoring two physiological signals: skin conductance (SC) and skin temperature (ST). These signals were collected using a wearable sensor at a fraction of the cost of commercial eye trackers. The use of physiology to track MW is motivated by the relationship between sympathetic nervous activity (captured by SC and ST) and attentional states [6].

A previous study found a higher rate of MW was related to overall lower levels of skin conductance (SC) [7]. However, this result was not leveraged to build automated MW detectors. To our knowledge, no attempt has been made to build models capable of detecting MW using SC or ST signals, nor has there been research attempting to link ST and MW. Taking a step in this direction, we collected a large data set where students were periodically probed to report instances of MW during computerized learning from instructional texts. These signals were used to create machine learning models that predicted MW.

2 Methods

Data Collection. Participants were 70 undergraduate students from a medium-sized private mid-western University in the U.S. Students were seated in front of a computer and an Affectiva Q sensor was strapped to the inside of the student's non-dominate wrist, a standard placement to measure SC [8]. The Affectiva Q [9] provides a nonintrusive way to measure SC and ST of the student at sampling rates of 8 Hz.

Students were asked to study four texts, each on key research methods topics: experimenter bias, replication, causality, and dependent variables. On average, each text contained 1500 words (SD = 10 words) with approximately 60 words per page. Students were informed that they would be asked a series of test questions on each text after reading. Before each text, students were made aware of the point value of test questions related to the text – "high-value" text questions were worth three times more than "low-value" text questions. This was the value manipulation. In addition, there were also difficult vs. easy versions of the texts equated in terms of content and length (difficulty manipulation). These manipulations were integral to a larger research study, but are not the focus of this research.

As students progressed through the texts they were instructed to report if they were MW by responding to auditory probes. Auditory probes occurred at a random point 4 to 12 seconds from the beginning of pseudo randomly chosen probe pages. These probes are classified as "within-page" probes. If students attempted to advance to the next page before the probe appeared, they were probed with an "end of page" probe. Once an auditory probe occurred, students used a keyboard to indicate MW with a "yes" or normal reading with a "no" by selecting appropriate keys on the keyboard.

Students reported MW to end of page probes 16.9% of the time (N = 108), and they reported MW to within page probes 26.1% of the time (N = 526).

Model Building. Supervised classification was conducted to detect instances of MW from physiological signals and contextual features (discussed below). Models were built using WEKA [10] and were validated at the student-level - data was randomly split on students, with 67% for training and 33% in the testing set and repeated for 25 iterations. SC and ST signals were z-score standardized at the student level and a low pass filter was applied to the SC data at 0.3 Hz to reduce noise in the signal.

To account for physiological measurements compromised by abrupt movements, the average difference between consecutive x, y, and z accelerometer readings for each student was calculated from an accelerometer in the Affectiva Q. A threshold of five times the average was used to eliminate compromised data, as has been used in previous studies [11]. In instances where this threshold was reached, data 5/8ths of a second before the movement through 5/8ths of a second after the movement was discarded.

Features were extracted from windows of signal data between the triggering of the auditory probe and a variable number of seconds before the probe. Separate datasets were constructed for window lengths of 3, 6, 12, 20, and 30 seconds.

Physiological features were extracted from the SC and ST signals included the mean, standard deviation, maximum, the ratio of maxima, and ratio of minima [12]. These statistical features were calculated for: the *standardized signal*; an approximation of the *derivation of the signal* (D1) obtained by taking the difference from one data point to the next; an approximation of D1, or the *second derivate* (D2) [13]; the *frequency*, and *magnitude* obtained from the Fast Fourier transformation [11]; the spectral density of the signal with *Welch's* method; the *autocorrelation* of the signal at lag 10, and, in models where both ST and SC of the same window were used, the *magnitude squared coherence* between the signals. Other physiological features included slope and y-intercept of the slope *coefficient* of the linear trend line [13].

In all, 43 features from the SC signal and the same 43 from ST were extracted. A separate dataset was created for each combination of window sizes of SC and ST data in order to address different temporal combinations of these signals (e.g. SC data was extracted for a window size of 3 seconds while ST was extracted for a window size of 30 seconds). Coherence statistics were used if the window sizes matched.

Context features captured the context of the learning task and included features for text, timing, and difficulty and value. Difficulty and value features included the *current difficulty* and *current value* of the text and the *previous difficulty* and *previous value* of the previous text. Timing features include *total time elapsed* since the student started the reading portion of the experiment, the *time since starting the current* text, the *average page time*, the *previous page time*, and the ratio of *previous page time to average page time*. Text features were the *total number of pages* that the student had

read since starting the reading portion of the session and the *page number* of the current text. In all, there were 11 context features.

Data treatments were applied in various combinations to determine which combination of data treatments resulted in the most accurate model. First, tolerance analysis was used to eliminate features that exhibited multicollinerity. Second, three feature selection algorithms (Gain-Ratio, Info-Gain, or ReliefF) were used (on training data only) to rank the contribution of each feature, and either 25%, 50%, 75%, or 90% of the top features were selected. Third, the data was winsorized by setting outliers greater than 3 standard deviations from the mean to the corresponding value 3 standard deviations from the mean. Fourth, downsampling was applied to the training data to obtain an equal distribution of responses by randomly removing instances of the more frequent class until the classes were balanced. Fifth, SMOTE (oversampling) was applied to the training data by adding random synthetic samples of the less frequent class until the classes were balanced. Sixth, when context features were not used, probes were eliminated if the student spent less than 4 seconds on a probe page, as the student likely either was not reading or accidently advanced prematurely.

3 Results

Table 1 presents the kappa, a measurement of accuracy which corrects for random guessing, of the best models (highest kappa). The best models were standardized and outliers were winsorized. Neither of the best models used tolerance, downsampling, or oversampling. Within page MW responses were easier to detect (kappa = .22, *SD* across iterations = .11) than end of page probes (kappa = .14, *SD* = .11). As seen from the confusion matrices in Table 2, although the best models have a high true negative rate (accurately detecting when not MW), the hit rate (correctly detecting MW) was low.

Table 1. Models with kappas

Probe Type	Features	Window (SC, ST)	No. Feat	Classifier	Kappa
Best WP	SC+ST+CF	(3, 12)	36	Filtered Classifier	0.22
Best EoP	ST	20	34	LADTree	0.14
Alt. WP	SC+CF	30	7	LADTree	0.15
Alt. EoP	ST+CF	6	23	AdaBoost M1	0.10

Note. WP – within page; EoP = end of page; Alt = Alternative;

To address the low hit rates, we considered alternate models as shown in Table 2. These models have a lower kappa for within page (kappa = .15, *SD* = .11) and end of page probes (kappa = .10, *SD* = .09), but have higher MW hit rates. Both alternative models were standardized within subjects, winsorized, used context features, and were trained with upsampling. Neither model used tolerance analysis. The use of upsampling in both models may indicate that with more positive MW reports, higher rates of MW can be detected.

Table 2. Confusion matricies for models

Model	Best Models			Alternative Models		
		Predicted			Predicted	
	Actual	*Yes*	*No*	**Actual**	*Yes*	*No*
Within page	*Yes* (.26)	.30	.70	*Yes* (.26)	.57	.43
	No (.74)	.11	.89	*No* (.74)	.38	.62
End of Page	*Yes* (.16)	.14	.86	*Yes* (.17)	.41	.59
	No (.84)	.04	.96	*No* (.83)	.28	.72

Note. Prior probabilities (base rates) are in parantheses

4 General Discussion

We investigated the possibility of detecting MW, a frequent and harmful phenomenon, from two physiological markers and aspects of the interaction context. MW detection is in its infancy; hence our immediate goal was to demonstrate the feasibility of MW detection. The major finding of this work is that SC and ST both contain information that can be used to detect MW. We acknowledge that our detection rates are modest, but consider them to be promising as an initial investigation into the possibility of unobtrusive detection of momentary instances of MW, an elusive state that is difficult to study since it is a highly internal unconscious phenomenon. Our detection is complicated by the relatively low rates of MW (23.9% of probes), which complicates supervised classification. Furthermore, we attempted to detect MW in a student-independent fashion, which is important for generalizability to new students, but more challenging due to individual differences in physiological responding [6].

MW detection has a number of possible applications. Interventions could be initiated during moments of MW in learning sessions to increase engagement. For example, an ITS that has detected MW could reevaluate the difficulty of the task the student is undertaking or could attempt to reengage the student's attention.

There are a few limitations that need to be addressed in future studies. One limitation is the relatively small data set used to train the models, so replicating the study with a larger sample is warranted. The study was conducted in a lab since we were interested in a highly controlled environment for this initial investigation. However, replication in more authentic contexts is warranted. The use of physiological sensors are also somewhat limited in terms of scalability. All participants were undergraduate students, and a large proportion (69%) identified as Caucasian – it would be advisable to retrain the models with a more diverse data set to study generalizability to diverse student populations.

In summary, although the results detailed are promising as a first start, there are multiple directions in which this research can be extended. We are working towards expanding our models to include multimodal data such as eye gaze or facial features. It is possible that by including additional modalities we will be able to achieve improved detection rates than by using any single modality. This is, of course, an empirical question that awaits further investigation.

Acknowledgement. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. Schooler, J.W., Reichle, E.D., Halpern, D.V.: Zoning out while reading: Evidence for dissociations between experience and metaconsciousness. Think. Seeing Vis. Metacognition Adults Child, 203–226 (2004)
2. Smallwood, J., McSpadden, M., Schooler, J.W.: When attention matters: The curious incident of the wandering mind. Mem. Cognit. 36, 1144–1150 (2008)
3. Smallwood, J., Schooler, J.W.: The restless mind. Psychol. Bull. 132, 946 (2006)
4. Drummond, J., Litman, D.: In the zone: Towards detecting student zoning out using supervised machine learning. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part II. LNCS, vol. 6095, pp. 306–308. Springer, Heidelberg (2010)
5. D'Mello, S., Cobian, J., Hunter, M.: Automatic Gaze-Based Detection of Mind Wandering during Reading
6. Andreassi, J.L.: Psychophysiology: Human behavior and physiological response. Routledge (2000)
7. Smallwood, J., Davies, J.B., Heim, D., Finnigan, F., Sudberry, M., O'Connor, R., Obonsawin, M.: Subjective experience and the attentional lapse: Task engagement and disengagement during sustained attention. Conscious. Cogn. 13, 657–690 (2004)
8. Feidakis, M., Daradoumis, T., Caballé, S.: Emotion measurement in intelligent tutoring systems: what, when and how to measure. In: 2011 Third International Conference on Intelligent Networking and Collaborative Systems (INCoS), pp. 807–812 (2011)
9. Picard, R.W.: Measuring affect in the wild. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, p. 3. Springer, Heidelberg (2011)
10. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. SIGKDD Explor Newsl. 11, 10–18 (2009)
11. Guo, R., Li, S., He, L., Gao, W., Qi, H., Owens, G.: Pervasive and unobtrusive emotion sensing for human mental health. In: 2013 7th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), pp. 436–439 (2013)
12. Wagner, J., Kim, J., André, E.: From physiological signals to emotions: Implementing and comparing selected methods for feature extraction and classification. In: IEEE International Conference on Multimedia and Expo, ICME 2005, pp. 940–943 (2005)
13. Setz, C., Arnrich, B., Schumm, J., La Marca, R., Troster, G., Ehlert, U.: Discriminating stress from cognitive load using a wearable EDA device. IEEE Trans. on Inf. Technol. Biomed. 14, 410–417 (2010)

Knowledge Construction with Pseudo-haptics

Akihiro Kashihara and Go Shiota

The University of Electro-Communications, Tokyo, Japan
akihiro.kashihara@inf.uec.ac.jp

Abstract. Composing a knowledge map to represent knowledge included in an instructional material is an effective way to understand the material. We currently attempt to utilize tablet media for the knowledge map composition, which allows touch operations with fingers. In particular, we address the issues of how the touch operations could be accompanied with pseudo-haptic senses and whether these senses could produce better cognitive awareness and retention of knowledge learned from the material. Our approach to these issues is to design a model of pseudo-haptic effects that demonstrates what and how cognitive awareness is obtained from pseudo-haptic senses, and to develop a tablet tool on iPad presenting the pseudo-haptic senses as modeled. In this paper, we discuss knowledge construction with the tablet tool, and report the case study.

Keywords: pseudo-haptics, knowledge map, tablet media, visual incongruity.

1 Introduction

Learning with plural senses would bring about better results than learning with single sense. Most of related work on multimedia material and multi-modal user interface for learning has focused on accompanying learning process particularly with visual and auditory senses [1].

The emergence of tablet media such as iPad, on the other hand, brings about the possibilities for providing learners with new learning experiences to be obtained from touch operations with fingers where learning process could be accompanied not only with visual and auditory senses but also with haptic sense. Generating visual representation of an instructional material such as diagram, chart, map, etc. by means of the tablet media, for example, involves touch operations that allow learners to make clear various attributes such as relationships among concepts/knowledge embedded in the material [2,3]. The touch operations bring with visual sense that could enhance an awareness of the embedded attributes to promote understanding of the instructional material and retention of knowledge learned [4]. It seems difficult to obtain such cognitive effects from only referring to the material. In addition, the touch operations could accompany the process with haptic sense in addition to visual sense. The haptic sense is expected to produce better cognitive effects. In particular, we use pseudo-haptic sense to be obtained from touch operations on iPad.

This paper discusses knowledge construction with pseudo-haptics. We also demonstrate a tablet tool that allows learners to compose a knowledge map from an instructional text with touch operations accompanied with visual and pseudo-haptic senses. The haptic sense is expected to allow them to become aware of important attributes in

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 61–68, 2014.

the material and of errors in composing the map, and is expected to enhance the retention of knowledge learned from the composed map. This paper also reports a case study with the table tool. The results suggest the possibility that the pseudo-haptic sense produces the cognitive awareness and brings about better effects on the retention rather than the visual sense.

2 Learning Experience with Touch Operations

In the field of CHI and Intelligent UI, there is a lot of work addressing the issue of how to accompany touch operations on the user interface with pseudo-haptics [5]. Pseudo-haptic sense is a kind of illusion that could occur from visual incongruity felt during operating objects [6]. For example, let us consider that a learner moves an object with drag operation on the user interface. The object generally follows the finger to move. But if the object does not follow the finger and move slowly in comparison with the finger movement, he/she would accept the visual incongruity, and have a feeling that the object is heavy. Such feeling is called pseudo-haptic sense.

In applying the pseudo-haptics to the touch operations on the tablet media, we need to analyze what kind of cognitive effects could be obtained from it. The pseudo-haptics has been exhaustively studied in related work on CHI and Intelligent UI where the main focus is what kind of pseudo-haptics could be obtained from operations in the user interfaces [6]. As far as the authors know, however, there is little work addressing the issues whether the pseudo-haptics could produce cognitive effects, and what kind of cognitive effects could be obtained from it.

We have accordingly addressed these issues in composing a knowledge map from an instructional text with touch operations on iPad. In addition, the visual operations for map composition do not always allow the learners to become aware of any attributes embedded in the text. For example, it is not easy for the learners to become aware of the importance of nodes/links even via the visual operations. The important nodes/links could be beforehand distinguished with color from others on the map. In this case, however, the learners are given the importance as the distinguished nodes/links, and they do not become aware of it by themselves. It is also hard for them to become aware of errors in the map composition process.

In this paper, we focus on cognitive awareness and retention as the cognitive effects. First, we expect that the pseudo-haptics to be presented could allow learners to become aware of important attributes in the instructional text and of errors in composing the map. Second, it could promote retention of knowledge learned from the composed map.

3 Knowledge Map Composition with Pseudo-haptics

Focusing on the cognitive awareness, we have designed the model of pseudo-haptic effects, which demonstrates what and how pseudo-haptic sense is presented during touch operations and what cognitive awareness is obtained from the presented pseudo-haptic senses. We have also developed a tablet tool that presents the pseudo-haptic

senses as modeled when the learners operate important nodes/links or make errors in composing a knowledge map from an instructional text.

Following the lessons learned from the previous work [4], we have refined the model of pseudo-haptic effects as shown in Table 1. This shows what kind of pseudo-haptic sense is presented from visual incongruity that is caused by visual movement of the node/link during touch operation with the intention of manipulating a node, a link, or the map. It also shows what cognitive awareness is brought about by the presented pseudo-haptic sense.

Table 1. Model of pseudo-haptics effects

Map manipulation	Touch operation	Visual movement	Pseudo-haptic sense	Cognitive awareness
Node movement	Drag	Delay in node movement	Node heaviness	Important knowledge
Linking	Draw	Shortening after linking	Tension between nodes	Important relationship
Link extension	Drag	Shortening after extension		
Linking	Drag	Away from the link	Repulsion between node and link	Incorrect relationship
Map shake	Shake	Link coming off	Loss of tension	Incorrect relationship
		Node vibration	Unstable force	Deficient relationship
		Label coming off	Loss of sticky force	Incorrect label

From the first to fourth rows, the pseudo-haptic effects for node or link operation are shown. For example, the first row illustrates the pseudo-haptics presentation and cognitive awareness when a learner moves a node with drag operation and accepts the incongruity from the visual movement that the node is delayed in comparison with the finger movement. He/she is expected to have a feeling that the node is heavy. This pseudo-haptic sense is also expected to provide him/her with awareness that the corresponding concept/knowledge is important. The fourth row illustrates cognitive awareness of incorrect relationship to be brought about when a learner links unrelated nodes and accepts the incongruity from the visual movement that the node to which is linked keeps away from the link. He/she is expected to feel repulsive force between the node and the link. This pseudo-haptic sense will provide him/her with awareness that making the relationships between the concepts/knowledge is incorrect. If he/she does not fee it, he/she could make the link between the nodes although it is incorrect.

In the fifth to last rows, the pseudo-haptic effects for map shake operation are shown. For example, the fifth row illustrates cognitive awareness of incorrect relationship when a learner shakes iPad after composing the map including a link between unrelated nodes and accepts the incongruity of the visual movement that the link comes off. He/she is expected to feel the loss of tensile strength between the nodes. This sense will bring about awareness that the composed relationship is incorrect.

Fig. 1 shows the user interface of the tablet tool. This tool prepares an instructional text and the corresponding correct map. The important nodes/links in the map are defined in advance by an instructor. The tool also embeds the visual movements for the pseudo-haptic senses in operating the corresponding nodes/links/labels.

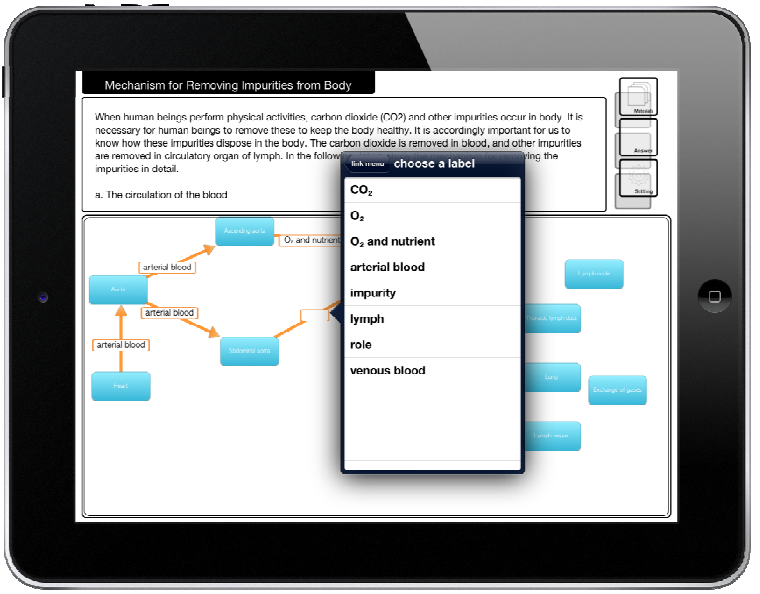


Fig. 1. User interface of the tablet tool

Fig. 2 shows an example of instructional text describing the mechanism for removing impurities from body and the corresponding correct map. Since lung and lymph nodes play a crucial role in removing carbon dioxide (CO_2) in blood and other impurities collected at lymph vessel, these two nodes are defined as important ones in the map. Since these organs also change impure matter into pure matter to transmit to other organs, the links representing the flows of the matter are defined as important ones. The importance of these nodes and links is stated in the text. When a learner operates these important nodes/links, the embedded visual movements are demonstrated. When he/she also makes incorrect links or leaves related nodes unlinked, the corresponding visual movements are also demonstrated.

The learners are required to complete a knowledge map corresponding to the correct map by carefully reading the text provided that all nodes in the correct map are beforehand presented. They are expected to compose a map by repeating two phases that are map composition and map confirmation. In the map composition phase, the learners are expected to locate/relocate the nodes and make the links between the nodes. In linking the nodes, they are also required to stick a suitable label on the link by selecting it from the menu including all labels necessary for the correct map composition. In addition, the tool detects the difference between the composed map and the correct map to identify incorrect links/labels and deficient links. In the map confirmation phase, the learners are expected to shack the composed map to confirm whether there are any errors in the links or labels and whether there are any deficient links. After confirming the map, they are expected to get back to the composition phase by means of the phase transition button in the user interface until the composed map corresponds to the correct one.

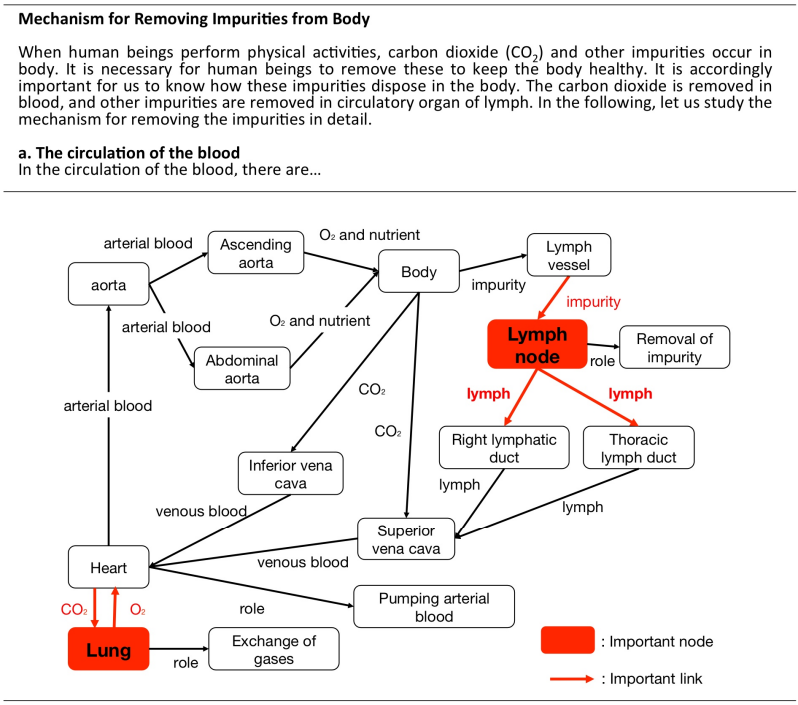


Fig. 2. An example of instructional material

4 Case Study

4.1 Preparation and Procedure

We have conducted a case study whose purpose was to ascertain whether the tablet tool could provide the pseudo-haptic senses and cognitive awareness as modeled and enhance the retention of knowledge compared to knowledge map composition only with visual operations. In order to allow the participants to compose the knowledge map without pseudo-haptics, we prepared the control tool that removed the function of demonstrating the visual movements for the pseudo-haptics from the tablet tool. Instead of this function, the control tool visualizes the importance of nodes/links and errors by coloring the important nodes/links and by giving alerts in making incorrect links, sticking incorrect labels, or leaving deficient links. In this study, we also prepared two instructional texts whose domains were the mechanism for removing impurities from body as shown in Fig. 2 and the classification of terrestrial plants, and whose correct maps (M-network and M-tree) had network and tree structure. The total numbers of nodes/links are 15/20 in M-network and 14/13 in M-tree. Out of these nodes/links, the numbers of important nodes/links defined in advance are also 2/5 in M-network and 4/3 in M-tree.

The hypotheses we set up in this study were as follows:

- H1: the tablet tool presents the pseudo-haptic senses as modeled,
- H2: the tablet tool provides the same degree of cognitive awareness as the control tool provides, and
- H3: the tablet tool enhances retention of learned knowledge compared to the control tool.

The participants were 16 graduate and undergraduate students in technology who belonged to the same university. We set two conditions, which were knowledge map composition with pseudo-haptics (With-PH) and with visual sense (With-VI).

We first prepared knowledge map composition session, in which half of the participants composed M-network with the tablet tool (With-PH) and subsequently composed M-tree with the control tool (With-VI). The remaining half of them first composed M-network with the control tool (With-VI), and subsequently composed M-tree with the tablet tool (With-PH). The time limit given for each map composition was 30 minutes. Even if the participants could not compete the knowledge map within the time limit, they were not given the correct map. Before the session, each participant was given an explanation about how to use the control tool, and was not informed beforehand that visual movement of important node/link for pseudo-haptics was demonstrated, and what the visual movement meant.

After the session, we next prepared questionnaire session, in which each participant was requested to first answer the questions for evaluating cognitive awareness to be provided with each tool and then to answer the questions for evaluating pseudo-haptics to be presented under With-PH. Table 2 shows a part of these questions. Each participant was allowed to refer to and touch his/her composed map.

On the next day, we set post-test session, in which each participant was required to reproduce the knowledge map for each instructional text within 20 minutes without any pseudo-haptic senses, any visualization of important nodes/links, and any alerts provided that all nodes in the correct map were given in advance. He/she was accordingly allowed to make the links between the given nodes to reproduce the map.

Table 2. Questionnaires for evaluating pseudo-haptics and cognitive awareness

Cognitive awareness	Questions
Q1	Select the nodes that felt important from the following: 1.Aorta 2.Ascending aorta 3.Abdominal aorta 4.Body 5.Lymph node …
Q2	Select the links that felt important from the following: 1.Heart->Aorta 2.Aorta->Ascending Aorta 3.Heart->Lung…
Pseudo-haptic sense	**Questions**
Q7	Select the node that felt heavy from the following: 1.Aorta 2.Ascending aorta 3.Abdominal aorta 4.Body 5.Lymph node …
Q8	Select the links in which you felt tensile strength from the following: 1. Heart->Aorta 2. Aorta->Ascending Aorta 3. Heart->Lung…

4.2 Results and Considerations

Table 3 shows the average data of the map composition and post-test. As for the map completion degree, there was no significant difference in completing important links between With-PH and With-VI although there was significant difference in completing the other links (two-sided t-test, $t(14)=2.66$, $p<0.05$) in M-network.

Regarding the visual movements for important nodes/links, node delay and link shortening were frequently demonstrated. Such frequent demonstration allows learners to have more chances to become aware of important knowledge and errors by themselves. As for the visual movements for errors and the alerts given for the corresponding errors in M-network, the total number of the visual movements on With-PH was more than the total number of the alerts on With-VI (two-sided Welch's t-test, $t(8)=2.05$, $p<0.10$). The average number of phase transitions was also larger on With-PH (two-sided t-test, $t(14)=1.69$, $p<0.10$). These results suggest that the M-network composition on With-PH is more complicated than the one on With-VI.

In order to ascertain the hypothesis H1, let us next examine the average ratios of the number of the participants who selected the important nodes/links in Q7/Q8 to the number of the participants who were provided with the corresponding visual movements. Each average ratio is calculated per the important node/link. From the results of the pseudo-haptics presentation, the pseudo-haptic senses for important nodes/links were presented with a very high degree. The pseudo-haptic senses for errors also tended to be presented with a higher (lower) degree in M-network (M-tree) since there were more (fewer) visual movements for errors in the map composition.

Table 3. Average data obtained from the knowledge map composition process and post-test

		M-network		M-tree	
		With-PH	With-VI	With-PH	With-VI
N		8	8	8	8
Map completion	Important links	0.95	1.00	1.00	1.00
	Other links	0.84	1.00*	1.00	1.00
Visual movements for important nodes/links	Node delay	48.38	--	46.16	--
	Link shortening (linking)	1.23	--	1.17	--
	Link shortening (extension)	43.05	--	69.25	--
Visual movements / Alerts for errors	Total	56.25^{+}	32.5	23.88	22.38
Phase transition		7.38^{+}	4.13	1.13	2.25
Pseudo-haptics presentation	Selection of Important nodes in Q7: Node heaviness	1.00	--	1.00	--
	Selection of Important links in Q8: Link tension	0.82	--	0.83	--
Cognitive awareness	Recall of the selected nodes	1.00	1.00	0.78	0.88
	Recall of the selected links	0.43	0.58	0.75	0.67
Post-test	Recall of reproduced links	0.81	0.84	0.85	0.87
	Recall of I-links/O-links	0.97/0.74	0.80/0.85	0.96/0.81	1.00/0.83

t-test, *:$p<0.05$, $^{+}$:$p<0.10$

As for H2, we calculated the recall of the selected nodes/links in Q1/Q2. The recall represents the ratio of the selected important nodes/links to all the important ones the participants could make in the map composition session. From the two-sided t-test with the recall, there were no significant differences between With-PH and With-VI. As for the incorrect relationships/labels and deficient links, there were no significant differences between each condition in almost all questions. These results suggest that cognitive awareness on With-PH is provided with the same degree as With-VI.

In order to ascertain H3, we next examine the recall of the reproduced links in the post-test. From the one-sided t-test, there was no significant difference between With-PH and With-VI. We then divided the reproduced links into the links corresponding to

the important ones (I-links) and the others (O-links). From the mixed model ANOVA with the recall in M-network, there was an interaction between With-PH/With-VI factor and I-links/O-links factor ($F(1, 31)=3.76$, $p<0.10$). From the simple main effect test, there was a tendency toward significant difference between With-PH and With-VI on I-links ($F(1,14)=3.37$, $p<0.10$). There was also significant difference between I-links and O-links on With-PH ($F(1,14)=6.50$, $p<0.05$).These indicate that the pseudo-haptics has more influence on the retention of important knowledge than the visual sense, and contributes to retaining important knowledge rather than others.

From the above results, we consider that cognitive awareness of important knowledge to be provided from the pseudo-haptics in more complicated map composition promotes the retention. We think the reasons as follows. First, the pseudo-haptics allows the learners to find out the important knowledge by themselves. The pseudo-haptics would also induce the learners to review the instructional text to find out the reason why the pseudo-haptic sense occurs. Such review process would contribute to retaining the important knowledge.

5 Conclusion

This paper has discussed knowledge construction with pseudo-haptics, which includes the model of pseudo-haptic effects and the tablet tool for composing a knowledge map with touch operations. In the case study, the three hypotheses are confirmed to some extent. The pseudo-haptics and cognitive awareness could be provided on iPad, and the cognitive awareness of important knowledge could promote the retention particularly in the context of more complicated map composition.

In future, we will conduct more detailed evaluation to refine the tablet tool.

Acknowledgments. The work is supported in part by Grant-in-Aid for Challenging Exploratory Research (No. 25560106) from the Ministry of Education, Science, and Culture of Japan.

References

1. Woolf, B.P.: Building Intelligent Interactive Tutors: Student-centered strategies for revolutionizing e-learning. Morgan Kaufmann Publishers Inc. (2008)
2. Jonassen, D.H.: Computers as Mindtools for schools, 2nd edn. Merrill Prentice Hall (2000)
3. Lajoie, S.P.: Computers As Cognitive Tools, 2nd edn. Lawrence Erlbaum Assoc. (2000)
4. Shiota, G., Kashihara, A.: Cognitive Effects of Concept Map Generation with Pseudo-Haptic Feedback. In: Proc. of ITHET 2013 in IEEE Xplore (2005), doi:10.1109/ITHET.2013.6671032
5. Lecuyer, A.: Simulating Haptic Feedback Using Vision, Teleoperators and Virtual Environments, vol. 18(1), pp. 39–53. MIT Press (2009)
6. Watanabe, K., Yasumura, M.: VisualHaptics: Generating haptic sensation using only visual cues. In: Proc. of Advances in Computer Entertainment Technology, pp. 405–405 (2008)

A Tool for Integrating Log and Video Data for Exploratory Analysis and Model Generation

Victor Girotto, Elissa Thomas, Cecil Lozano, Kasia Muldner, Winslow Burleson, and Erin Walker

Computing, Informatics & Decision Systems Engineering, Arizona State University
{victor.girotto,eethomas,cecil.lozano,katarzyna.muldner, winslow.burleson,erin.walker}@asu.edu

Abstract. Analysis of students' log data to understand their process as they solve problems is an essential part of educational technology research. Models of correct and buggy student behavior can be generated from this log data and used as a basis for intelligent feedback. Another important technique for understanding problem-solving process is video protocol analysis, but historically, this has not been well integrated with log data. In this paper, we describe a tool to 1) facilitate the annotation of log data with information from video data, and 2) automatically generate models of student problem-solving process that include both video and log data. We demonstrate the utility of the tool with analysis of student use of a teachable robot system for geometry.

Keywords: log analysis tool, cognitive modeling, intelligent tutor.

1 Introduction

With the advent of the web and the ubiquity of computing, log data from educational systems is dramatically increasing [1]. Analysis of log data that contains information about interactions with these systems helps researchers create different expert and novice models of student behavior [2, 3, 4]. These models allow intelligent systems to adapt to the learner's knowledge, ability and needs [5] by tracking interactions and making inferences about what students know or need feedback on [6].

Some forms of interaction are better captured in video, such as gestures, discussion with teachers, or even the use of non-digital materials. Analysis of video can be very time-consuming [7], and therefore a number of different tools have been developed, mainly to support annotation. AVnnotator [8] facilitates the addition of contextual information with a variety of lenses that allows users to document different categories of information to any given scene in a video file. Other tools allow exporting annotations so that researchers can manually integrate these with other analysis resources [9]. However, none of these tools integrate video and system logs.

This paper presents a tool that facilitates integrated analysis of data in these two formats. Our approach leverages behavior graphs, first proposed by Newel [10]. McLaren et al., in an approach called Bootstrapping Novice Data (BND), demonstrated how log data could be used to automatically generate behavior graphs [11]. Like McLaren,

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 69–74, 2014.

our tool supports automatic generation of behavior graphs, but extends it by supporting the integration of video code data. This enables users to annotate logs with additional information obtained through video analysis and to see their annotations automatically represented on the behavior graphs. This approach makes two contributions: 1) We incorporate video data in the automatic graph generation, and 2) We provide a visualization that makes it possible to easily explore relevant expert and novice models from the graph characteristics.

In the remainder of this paper, we describe our tool and then present a proof of concept analysis of its efficacy using data collected from student use of our *Tangible Activities for Geometry* (TAG) system. TAG is an embodied environment that supports middle school students during learning of geometry in a digitally-augmented physical space [12]. It uses a teachable agent paradigm, where students are told they will tutor a robot named Quinn to solve problems such as "Plot the point (3,1)". They can do so by moving within the physical space to give Quinn commands such as Move *N* units, or Turn *M* degrees. When students believe they have solved the problem they can check the correctness of their solution and receive feedback from TAG. In theory, students benefit from using TAG by encoding the relevant geometry concepts in an embodied way, and by making their reasoning explicit while instructing Quinn on how to solve the problems. Through the models generated by our tool, we can explore the strategies, misconceptions, preferences and influence of embodiment on student performance. Ultimately, this information can help the system in tailoring the feedback given to students during problem-solving.

2 The Analysis Tool

In order to understand how a system like TAG can provide better adaptive guidance to students, there is a need to identify student strategies and misconceptions, and how these interact with students' embodied behaviors. The analysis tool that we developed generates, for both aggregated and individual student log data, a behavior graph (defined below). The tool syncs log data with video data; a researcher can annotate the log data with video information as he or she views a student's video. The behavior graph is updated as annotations are made, giving visual insight on the relationship between video data such as gestures and actions performed. Fig. 1 shows the main interface of the application, comprised of two main sections: log/video information (items A and B) and the behavior graphs (items C and D).

The behavior graphs consist of *states* (location and orientation of the teachable agent in the coordinate plane; represented by circles) and *transitions* between states, corresponding to actions (e.g., *moving from (0,0) to (1,0)*; represented by lines). Characteristics of the underlying data are visually encoded as follows. The node size and transition thickness are proportional to how many students passed through them, with larger nodes or thicker transitions indicating more students. Color is also used: a blue node indicates the starting state; green and red indicate correct and incorrect states, respectively; and color intensity characterizes the number of students who have checked their solution at that state, resulting in a white node if no student checked the solution at that particular state. The text inside of the nodes shows the state that it represents, using the format *x*|*y*|*orientation* (e.g. 2 | -1 | 90).

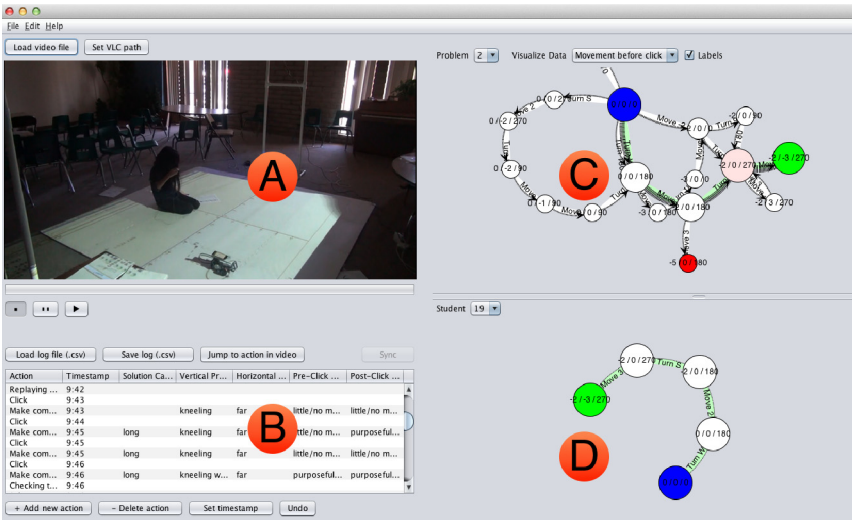


Fig. 1. The tool's main interface. (A) Video viewport. (B) Log table. (C) Aggregate behavior graph. (D) An individual student behavior graph.

The tool includes two graphs: the *aggregate graph* shows information from all students for any given problem (Fig. 1C) and the *individual graph* shows information from a single student (Fig. 1D). Given the high amount of data that needs to be represented in the graph, users can interact with the graph in several ways. It is possible to change problems and students, enable or disable labels, pan, zoom, move nodes around, and switch between the visualization of different annotations. Clicking on a node or transition displays detailed information, such as which students passed through it or how many checked for correctness.

The final aspect of this tool is its support of a seamless two-way navigation between the video and logs, making it easier to sync actions and to enable annotation of the log file based on video information. The tool automatically highlights the current action in the imported log file as the user plays the video. Alternatively, the user can move through the log file and the video will automatically sync to the log location. Users can annotate log entries using free-form text through the table seen in Fig. 1B. Each annotation is associated to its respective edge on the graph and receives a weight. The graph then encodes these weights visually by coloring its edges, using a darker green to denote a higher occurrence of this annotation, and a lighter one to denote a lower occurrence.

3 Using the Tool for Analysis: Proof of Concept

We used this tool to analyze data from a prior study [13], with the goal of better understanding student behavior in order to guide future developments of the system.

In that study, 19 subjects (8 female, 11 male) spent 45 minutes teaching Quinn how to solve point-plotting problems. Interactions were recorded both in the system's logs and in video, which were loaded into the tool. We will now describe the four exploratory analyses done using the tool.

Metacognitive Strategies. Through visual inspection of the graphs, we derived a set of metacognitive strategies that students used to solve problems. The strategies identified were: 1) W*andering*: the student follows a long path that does not lead to the solution (used by 3 participants). 2) C*hecking and resetting*: the student follows a path, checks it, and if incorrect, restarts the problem and tries a different approach (used by 11 participants). 3) C*onstant checking*: the student checks the answer after most actions (used by 3 participants). 4) *Intelligent novice*: the student takes a slightly long path to the correct solution (used by 13 participants). 5) *Expert*: the student moves directly towards the correct solution (used by 12 students). This information can aid the system in intervening positively to improve student performance.

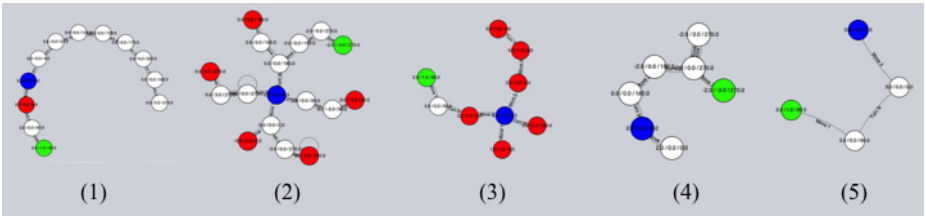


Fig. 2. Visualization of the metacognitive strategies taken by students while solving problems in the TAG system. (1) Wandering (2) Checking and resetting (3) Constant checking (4) Intelligent novice (5) Expert.

Bug Taxonomy. We also used the behavior graph to identify the nodes where students submitted an incorrect response, and classified their misconceptions. We identified several common misconceptions across students. Some examples are: 1) *Sum coordinates*, student summed the two numbers in the coordinate and move that amount in one arbitrary axis. 2) *Switch x and y*: student switched the x-axis with the y-axis. *3) Move only in one dimension* student moved the correct distance in either x or y, but remained in zero for the other dimension. The system could use this information to address misconceptions individually.

Multiple Paths to a Solution. The behavior graph enables a visualization of the various paths taken by students to get to the answer (both correct and incorrect), with thicker edges indicating more students took a given path. Therefore, we could identify that most students preferred to move positive distances instead of negative distances and generally turned using cardinal points instead of angles. The system could use this information to prompt students to consider alternative paths.

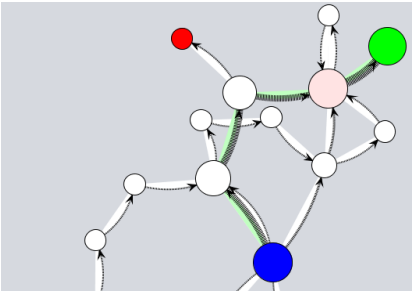


Fig. 3. Behavior graph with video information. A green edge indicates more embodiment

Influence of Embodiment. We also used the tool to encode video information into the log files. For this analysis, we encoded information from a single problem that produced a variety of correct and incorrect solution strategies from participants who interacted with the system. Within this problem, we annotated log data with participants' levels of embodied movement. Subject movement was coded using a binary schema: At each opportunity for physical interaction, a score of either 0 (little/no movement) or 1 (purposeful movement) was added as an annotation to the subject's log data. These annotations added a green highlighting to the edges of the behavior graph that denoted a higher level of movement. As illustrated in Fig. 3, by looking at the subject's behavior at each step in the problem solving process, we can identify a higher average level of embodied movement and behaviors occurring on transitions that are part of correct solution paths. This exploratory visualization may indicate an interesting relationship between levels of embodiment and problem-solving success. We see this analysis as a jumping-off point for quantitative analysis of the relationship between embodiment in our system and problem-solving success.

4 Conclusion and Discussion

In this paper, we presented a tool that facilitates analysis by integrating data from logs and video into a behavior graph. The features of this tool were demonstrated using data from a study that used the TAG System. Using the graph generated by the tool, we identified strategies, misconceptions and multiple solution paths. Furthermore, the encoded video information provided visual insight on aspects such as the relationship between a student's movements and their efficiency in solving the posed problems.

Future research could focus on making this tool generalizable, enabling it to work on systems that use different log structures. Different forms of data visualization could also improve its usefulness, such as making use of the temporal aspect of the data, allowing users to see the evolution of the graphs. Lastly, clustering algorithms would be a natural step towards automating the analysis of this data, thus improving the speed through which conclusions could be drawn from the graph.

Acknowledgments. This research was funded by NSF 1249406: EAGER: A Teachable Robot for Mathematics Learning in Middle School Classrooms and by the CAPES Foundation, Ministry of Education of Brazil, Brasília - DF 70040-020, Brazil.

References

1. Romero, C., Ventura, S., García, E.: Data mining in course management systems: Moodle case study and tutorial. Comput. Educ. 51, 368–384 (2008)
2. Romero, C., Ventura, S.: Educational data mining: A survey from 1995 to 2005. Expert Syst. Appl. 33, 135–146 (2007)
3. Arroyo, I., Woolf, B.: Inferring learning and attitudes from a Bayesian Network of log file data. In: AIED, pp. 33–40 (2005)
4. Lau, T., Horvitz, E.: Patterns of Search: Analyzing and Modeling Web Query Refinement. Courses Lect. Cent. Mech. Sci., 119–128 (1999)
5. Mostow, J., Beck, J.: Some useful tactics to modify, map and mine data from intelligent tutors. Nat. Lang. Eng. 12, 195–208 (2006)
6. Murray, T.: Authoring knowledge-based tutors: Tools for content, instructional strategy, student model, and interface design. J. Learn. Sci. 7, 5–64 (1998)
7. Harrison, B., Baecker, R.: Designing video annotation and analysis systems. Graph. Interface. 92, 157–166 (1992)
8. Costa, M., Correia, N., Guimarães, N.: Annotations as multiple perspectives of video content. In: Proc. Tenth ACM Int. Conf. Multimed., pp. 283–286 (2002)
9. Kipp, M.: ANVIL: A Generic Annotation Tool for Multimodal Dialogue. In: Proceedings of the 7th European Conference on Speech Communication and Technology (Eurospeech), Aalborg, Denmark, pp. 1367–1370 (2001)
10. Newell, A.: On the analysis of human problem solving protocols (1966)
11. McLaren, B., Koedinger, K., Schneider, M.: Bootstrapping Novice Data: Semi-automated tutor authoring using student log files (2004)
12. Muldner, K., Lozano, C., Girotto, V., Burleson, W., Walker, E.: Designing a Tangible Learning Environment with a Teachable Agent. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 299–308. Springer, Heidelberg (2013)
13. Muldner, K., Girotto, V., Lozano, C., Burleson, W., Walker, E.: The Impact of a Social Robot's Attributions for Success and Failure in a Teachable Agent Framework Tangible Activities For Geometry (TAG). In: International Conference of the Learning Sciences (2014)

Virtual Environment for Monitoring Emotional Behaviour in Driving

Claude Frasson, Pierre Olivier Brosseau, and Thi Hong Dung Tran

Université de Montréal
Département d'informatique et de recherche opérationnelle
2920 Chemin de la Tour, Montréal, H3T-1J4, Canada
{frasson,pierre-olivier.brosseau,mylife.tran}@iro.umontreal.ca

Abstract. Emotions are an important behaviour of humans and may arise in driving situations. Uncontrolled emotions can lead to harmful effects. To control and reduce the negative impact of emotions, we have built a virtual driving environment in which we can capture and analyse emotions felt by the driver using EEG systems. By simulating specific emotional situations we can provoke these emotions and detect their types and intensity according to the driver. Then, in the environment, we generate corrective actions that are able to reduce the emotions. After a training period, the driver is able to correct the emotions by himself.

Keywords: Emotions, Simulation, EEG, Driving, Emotional state.

1 Introduction

Emotions play an important role in decision. Emotions can last from a few minutes to several days (in this case they are called moods). What is more important is that they place the driver into a cerebral state that will allow or disallow him/her to react adequately to a cognitive or a decisive situation. *Mental engagement* is related to the level of mental vigilance and alertness during the task. Sometimes engagement is considered as the level of attention and motivation. The loss or diminution of engagement is considered as a *distraction* [16]. *Mental workload* can be seen as the mental effort and energy invested in terms of human information processing during a particular task. If a driver is in a high mental workload he can ignore possible dangers. While driving, these emotions can have very harmful effects on the road, or even cause death. For instance, anger can lead to sudden driving reactions, often involving collisions. Sadness or an excess of joy can lead to a loss of attention. Generally, emotions that increase the reaction time in driving situations are the most dangerous. Several questions arise. How do we measure or estimate the emotion of the driver in certain situations? How can the driver act on his emotions to reduce their intensity? How can we train the driver to react differently and control his emotions?

Different technologies can be used to assess emotions. We can use physiological sensors that are able to evaluate seat position, facial recognition, voice recognition, heart rate, blood pressure, sweating and the amount of pressure applied on the computer

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 75–83, 2014.

mouse. The galvanic skin conductivity is a good indication of emotional change but its evaluation is not precise. The use of Electroencephalograms (EEG) sensors is more precise and more recently used [17]. In fact, EEG signals are able to detect emotions and cerebral states which, synchronized with the driving scene, can highlight what happens in the brain and when. To reduce emotions, most of the systems use a voice to interact with the driver. In the present paper we aim to assess emotions felt by a driver in specific driving situations. For that, we have built a virtual environment that is able to generate these emotions. Then, a virtual agent intervenes to reduce the emotional impact so that the driver can return to a neutral emotion. Following this introduction, we first comment first on previous works realized in this domain. Then, we present the main components of our simulator, a virtual environment that is able to generate emotions and an agent in charge of reducing emotions. We describe the experiments realized, show the resulting emotions and the measures obtained. Finally, we show how the system can reduce emotional reactions and create an impact on reducing road accidents.

2 Previous Work

Intuitively emotions play a role in driving, but even if they are not listed as a direct factor in road accidents [4, 12], it is reported that 16 million drivers in the United States have disabilities road rage [10]. What is the effect on driving when emotions such as anger and excitement arise, since they increase the driver reaction time?

Nass, Jonsson et al. [1, 2] realized a study to determine whether a car equipped with the ability to speak may influence the performance of its user. Participants of the simulation were invited to converse with the voice of car. Results showed that when the voice of the car met the voice of the participant (happy / sad / moderate) he had less accidents, paid more attention to the road and was more involved in the conversation with the voice of car. Jones and Jonsson [14] have presented a method to identify five emotional states of the driver during simulations. They used neural networks as classifiers, but they have not studied the impact of ambient noise. Schuller et al. [3] also based their experiences on driving simulators recognizing four emotions using support vector machines. However, these studies have shown that the performance of emotion recognition depends largely on the ratio of noise that they have also ignored.

Results obtained by Cai et al. [13] show that anger and excitement, in a scenario involving several drivers, cause an increase in heart rate, breathing and skin conductivity. More specifically, drivers who are not in the neutral state, cross more the lines on the road, turn more on the wheel, and are changing lanes much faster when they are angry or excited. We can conclude that the emotions of anger and excitement negatively affect the control of the vehicle when driving as compared to driving in a neutral state. And this control is directly connected to road safety.

Works undertaken by a team at The Institute Human-Machine Communication in Munchen [18] confirm the influence of the affective state on driver performances. Again, the study emphasized the importance of developing an intelligent system inside the car. To achieve this, emotion plays a significant role in the comfort and safety

of driver's performance. Facial expressions, voice, physical measurements, driving parameters and contextual knowledge of the driver are important and reliable methods for recognizing the emotions and state of the driver. A distraction detection system is also under way and will assist the driver with a Lane Keeping System and a Head Tracking System. Research on emotions detection is being funded by Toyota. Their system, which is still in the prototype stage, can identify the emotional state of the driver with a camera that stands 238 points on their face. The car can then make suggestions to the driver, or simply adjust the music for relaxation. Everything is still in the prototype stage, but Toyota says that their system could be available in their next car generation.

3 The Emotional Car Simulator

3.1 The Environment

To generate and assess emotions in a driving situation we have built a Virtual Environment able to simulate specific driving situations that could be a source of emotions. The virtual environment takes the form of a game in which the player is a driver who is submitted to a variety of realistic situations that everybody could experience every day in traffic. Our environment is divided into 6 parts: the profile of the user, the quiz (before and after simulation), the simulation, the emotion corrector (an agent able to calm the user and reduce his emotions), and the result part. On the right side of the interface we have integrated the measures which come in real time from an EEG headset: Excitement, Engagement, Boredom, Meditation, and Frustration. First of all, the user has to register and provide personal information (profile) in the Virtual Environment, then he is submitted to the first quiz in which he has to determine the perception of his own emotions. This quiz is invoked before and after each scenario in the simulator. The simulator is the part intended to cause emotional reactions. It is based on a video game where a user can experiment nine different driving scenarios designed to generate emotions by using stimulating sounds and mobile cars or trucks that suddenly arise in the traffic to disrupt the driving behaviour of the user. The emotion corrector module is intended to reduce player's emotions. It is represented by a virtual emotional agent which is aware of user's emotion and will try to talk to him according to various scenarios, explaining the good behavior to adopt in order to reduce his emotions.

3.2 Collecting the Data

To collect the data we used the EPOC headset built by Emotiv. EPOC is a high resolution, multi-channel, wireless neuroheadset which uses a set of 14 sensors plus 2 references to tune into electric signals produced by the brain to detect the user's thoughts, feelings and expressions in real time (Figure 1). Using the Affectiv Suite we can monitor the player's emotional states in real-time. This method was used to measure the emotions throughout the whole simulation process. Emotions are rated between 0 and 100%, where 100% is the value that represents the highest power/probability for this emotion.

Fig. 1. The experimental environment (EEG and simulation system)

3.3 The Correcting Agent

To correct the negative emotions generated in the scenarios we created an agent in charge of neutralizing player's emotions (Figure 2).

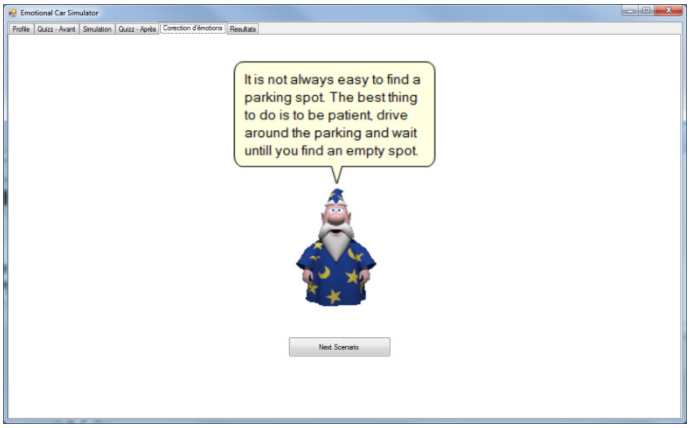


Fig. 2. The correcting agent

Its soothing voice combined with its funny appearance are there to reassure the player and tell him the proper behaviour to handle the scenario correctly and to reduce his emotions.

3.4 The Results

The last part of the virtual environment is an interface which shows the results. For each user it is possible to select a given scenario and retrieve the emotions captured by the EPOC headset during the simulation. Each pair of emotions can be hidden or shown in the graphic using this interface. For instance, in Figure 3, on the left side of the screen, we select the user (which is blurred out for privacy purposes), a scenario and the emotions to display.

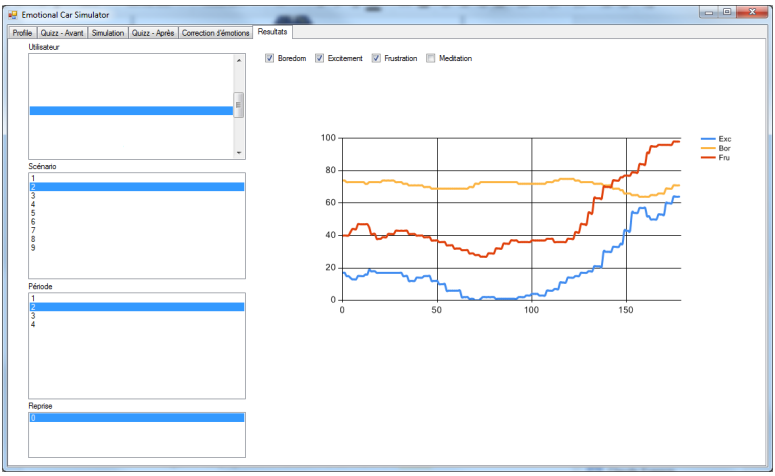


Fig. 3. The variation of emotions while in traffic, x-axis is the time and y-axis is the percentage of emotion

The graph appears on the right and displays the progression of the selected emotions over the course of scenario. It represents scenario #2, in which the player finds himself stuck in the traffic. We can clearly see the slow increase in both frustration and excitement.

4 Experimentation

From the EPOC we collect 3 main emotions: boredom, excitement and frustration. *Frustration* is an emotion that occurs in situations where a person is blocked from reaching a desired outcome. In general, whenever we reach one of our goals, we feel pleased and whenever we are prevented from reaching our goals, we may succumb to frustration and feel irritable, annoyed and angry. Typically, if the goal is important, frustration and anger or loss of confidence will increase. *Boredom* creeps up on us silently, we are lifeless, bored and have no interest in anything, due perhaps to a build-up of disappointments, or just the opposite, due to an excess of stimuli that leads to boredom, taking away our ability to be amazed or startled anymore when things happen. *Excitement* is a state of having a great enthusiasm while calm is a state of tranquility, free from excitement or passion.

The simulation module contains nine scenarios; each scenario is a different situation likely to cause emotions. The first scenario is simply intended to help the participant to become familiar with the car's controls and the simulator. In the second scenario, the participant is stuck in a traffic jam with a lot of noise. In the third scenario, the participant drives near a school with a school bus waiting on the other side of the road. The participant has to stop five meters before the bus and wait until the stop signal is gone. In the fourth scenario the participant is driving straight up to an intersection with a stop sign on his side, he has to wait for all other cars to pass.

The fifth scenario is similar to the fourth, this time with a pedestrian crossing the street. In the sixth scenario, the participant has to find a place in a public parking. There is only one place left and before the participant can reach it, another car takes it. The participant has to look around to find another place (Figure 4). In the seventh scenario, the car is already on a highway at high speed and the brakes are no more working. The participant has to stay calm, verifies if the brakes are working properly and tries to stop the car. In the eighth scenario, a fire truck comes from behind and starts its siren.

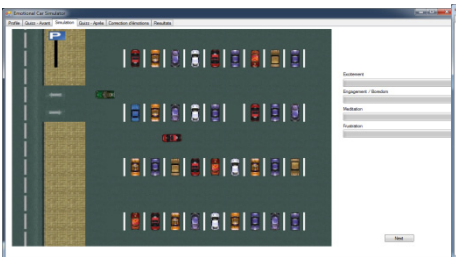


Fig. 4. The parking lot (Scenario #6)

Fig.5. The Fire Truck (Scenario #8)

The participant has to move his car to the right and stay immobilised until the fire truck is gone (Figure 5). In the last scenario, the participant is late for work. If the participant takes the first turn right he will arrive at work on time but there is an interdiction to turn right, he has to take the second right turn. The participant has to respect the signalisation and turn where it is permitted, even if he is in a hurry.

5 Results

The subjects of this study were 30 college students from Quebec, 6 females and 24 males aged between 17 and 33. Amongst them, 24 had their driving license. In this section, we present the common emotions generated during the scenarios. Participants are excited when an event (pedestrian, siren, stop sign, parking, etc.) occurs. They become very frustrated and excited when they realized that their brakes did not work (41.2% frustrated and 70.6% excited). Participants got bored when they had to wait for the pedestrian (23.5%) or when nothing happened (29.4% during the first scenario). Participants became very frustrated when they caused a collision (70.6% in scenario 6) or when they failed a scenario. The following figures show the influence of a Fire Truck'siren and a brakes failure. Figure 6 shows the generation of excitement when the Fire Truck started its siren. Figure 7 shows the generation of frustration when the user noticed that the brakes failed. These data are for a single user.

We consider a *generated* emotion by observing the emotions that have increased their value by at least 20% in the course of the scenario (Figure 6). A *corrected* emotion is also defined by observing a decrease of at least 20% between the end of the scenario and the end of the correcting agent phase (Figure 7).

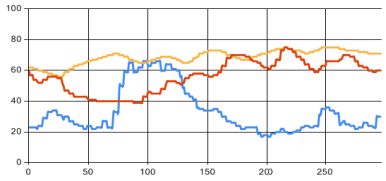


Fig. 6. Excitement generated

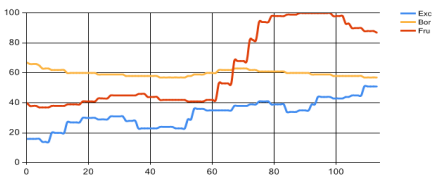


Fig.7. Frustration (in red) generated

Table 1. Average emotions generated in the simulator for all participants

Scena-rio/emotions	Excitement	Boredom	Frustration
1	35.3%	29.4%	11.7%
2	70.6%	11.8%	64.7%
3	70.6%	5.9%	47.1%
4	41.2%	11.8%	52.9%
5	52.9%	23.5%	41.2%
6	76.5%	11.8%	70.6%
7	41.2%	0.0%	70,6%
8	52.9%	0.0%	58.8%
9	52.9%	5.8%	64.7%

Strong emotions generated during the scenarios are corrected with the correcting agent. The efficiency is the percentage of the amount of reduction. The correcting agent worked with the best efficiency of 70.0% for the frustrated participants in scenario 8 (Table 2), and 66.7% of excited participants in scenario 3.

Table 2. Efficiency of the correcting agent for all participants

Scena-rio/emotions	Excitement	Boredom	Frustration
1	66.7%	21.1%	50.0%
2	50.0%	8.6%	45.5%
3	66.7%	4.8%	62.5%
4	42.9%	10.3%	66.7%
5	66.7%	19.7%	57.1%
6	46.2%	7.5%	41.6%
7	57.1%	0.0%	58.3%
8	55.6%	0.0%	70.0%
9	44.4%	4.3%	36.4%

Figure 8 shows the influence of the correcting agent on excitation in this case.

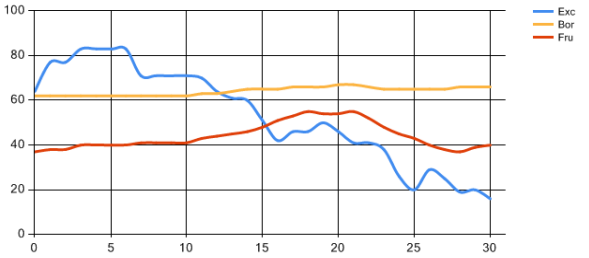


Fig. 8. The effect of the correcting agent on the high excitement of the user (while the agent is talking)

6 Conclusion

Emotions affect the drivers' behaviour. Strong emotions or negative emotions can lead to aggressive reactions. Correcting these strong or negative emotions can have a positive impact on the drivers' safety. In our simulator the correcting agent has a considerable impact to improve drivers' actions. Excitement and frustration have decreased after the driver followed the advices of the agent with an efficiency varying from 36.4% to 70.0%. Driving is an everyday activity in which people rarely prevent or stop these strong or negative emotions. The correcting agent improve the drivers' emotional behaviour. Our emotional correcting agent has only corrected the emotions of excitement and frustration, but it is also possible to create agents to correct other negative or strong emotions such as boredom, which can cause distractions and accidents on the road. Correcting this emotion is also an important step to improve driving quality and is a good subject for further research. Instead of correcting these emotions, preventing them is also a very interesting subject for further investigations. Preventing strong or negative emotion will also prevent bad driving behaviours. By training the emotional behaviour of the driver with successive use of our system it would reduce the impact on the driver's emotional state. Experiments in virtual environments have shown the improvement for the user in terms of emotional reactions (flight simulations, phobia reduction) and we think that applying our system repeatedly should have the same effect on drivers. A useful application of our system could be installed in driving schools, combining training driving codes and adequate emotional behaviour. In order to integrate portable technology, we could insert into future cars a face reading system which is able to detect not only driver's emotions but also his weariness. This approach will contribute to the birth of intelligent cars that will detect the capabilities and emotional conditions of the driver for a safer environment.

Acknowledgments. We acknowledge the National Science and Engineering Research Council (NSERC) for funding this work.

References

1. Jonsson, I.M., Nass, C., Harris, H., Takayama, L.: Matching In-Car Voice with Driver State: Impact on Attitude and Driving Performance. In: Proceedings of the Third International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design, pp. 173–181 (2005)

2. Nass, C., Jonsson, I.M., Harris, H., Reaves, B., Endo, J., Brave, S., Takayama, L.: Improving Automotive Safety by Pairing Driver Emotion and Car Voice Emotion. In: Proc. CHI (2005)
3. Schuller, B., Lang, M., Rigoll, G.: Recognition of Spontaneous Emotions by speech within Automotive Environment. In: Proc. 32. Deutsche Jahrestagung für Akustik (DAGA), Braunschweig, Germany, pp. 57–58 (2006)
4. Setiawan, P., Suhadi, S., Fingscheidt, T., Stan, S.: Robust Speech Recognition for Mobil Devices in Car Noise. In: Proc. Interspeech, Lisbon, Portugal (2005)
5. Grimm, M., Kroschel, K., Narayanan, S.: Support vector regression for automatic recognition of spontaneous emotions in speech. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Proccessing (ICASSP) (2007) (accepted for publication)
6. Grimm, M., Kroschel, K., Harris, H., Nass, C., Schuller, B., Rigoll, G., Moosmayr, T.: On the Necessity and Feasibility of Detecting a Driver's Emotional State While Driving. In: Paiva, A.C.R., Prada, R., Picard, R.W. (eds.) ACII 2007. LNCS, vol. 4738, pp. 126–138. Springer, Heidelberg (2007)
7. Al-Shihabi, T., Mourant, R.R.: Toward more realistic driving behavior models for autonomous vehicles in driving simulators. Transportation Research Record 1843, 41–49 (2003)
8. Lisetti, C.L., Nasoz, F.: Using noninvasive wearable computers to recognize Human Emotions from Physiological Signals. EURASIP Journal on Applied Signal Processing 11, 1672–1687 (2004)
9. Balling, O., Knight, M.R., Walters, B., Sannier, A.: Collaborative Driving Simulation. In: SAE 2002 World Congress & Exhibition, Session: Vehicle Dynamics & Simulation (Part A), Detroit, MI, USA, Document Number: 2002-01-1222 (March 2002)
10. CNN news, CNN News Health Study: 16 million might have road rage disorder (June 5, 2006), http://www.cnn.com/2006/HEALTH/06/05/road.rage.disease.ap/
11. Cowie, R., Douglas-Cowie, E., Cox, C.: Beyond emotion archetypes: Databases for emotion modeling using neural networks. Neural Networks 18(4) (May 2005); Emotion and Brain, pp. 371-388, DSC 2007 North America – Iowa City (September 2007)
12. NHTSA, Traffic safety facts, DOT HS 809 848, NHTSA annual report. Washington, USA (2005)
13. Cai, H., Lin, Y., Mourant, R.R.: Study on Driver Emotion in Driver-Vehicle-Environment Systems Using Multiple Networked Driving Simulators. DSC 2007 North America – Iowa City (September 2007)
14. Jones, C., Jonsson, I.M.: Automatic recognition of affective cues in the speech of car drivers to allow appropriate responses. In: Proc. OZCHI (2005)
15. Schuller, B., Arsic, D., Wallhoff, F., Rigoll, G.: Emotion Recognition in the Noise Applying Large Acoustic Feature Sets. In: Proc. Speech Prosody, Dresden, Germany (2006)
16. Stevens, R.H., Galloway, T., Berka, C.: EEG-Related Changes in Cognitive Workload, Engagement and Distraction as Students Acquire Problem Solving Skills. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 187–196. Springer, Heidelberg (2007)
17. Chaouachi, M., Jraidi, I., Frasson, C.: Modeling Mental Workload Using EEG Features for Intelligent Systems. In: Konstan, J.A., Conejo, R., Marzo, J.L., Oliver, N. (eds.) UMAP 2011. LNCS, vol. 6787, pp. 50–61. Springer, Heidelberg (2011)
18. Eyben, F., Wöllmer, M., Poitschke, T., Schuller, B., Blaschke, C., Färber, B., Nguyen-Thien, N.: Emotion on the Road—Necessity, Acceptance, and Feasibility of Affective Computing in the Car. In: Advances in Human-Computer Interaction, vol. 2010, 263593, 17 p. (2010)

The Affective Meta-Tutoring Project: Lessons Learned

Kurt VanLehn[1], Winslow Burleson[1], Sylvie Girard[2],
Maria Elena Chavez-Echeagaray[1], Javier Gonzalez-Sanchez[1],
Yoalli Hidalgo-Pontet[1], and Lishan Zhang[1]

[1] Arizona State University, Tempe, AZ, USA
{kurt.vanlehn,winslow.burleson,mchaveze,javiergs,
yoalli.hidalgopontet,Lishan.zhang}@asu.edu
[2] University of Birmingham, Birmingham, UK
s.a.girard@bham.ac.uk

Abstract. The Affective Meta-Tutoring system is comprised of (1) a tutor that teaches system dynamics modeling, (2) a meta-tutor that teaches good strategies for learning how to model from the tutor, and (3) an affective learning companion that encourages students to use the learning strategy that the meta-tutor teaches. The affective learning companion's messages are selected by using physiological sensors and log data to determine the student's affective state. Evaluations compared the learning gains of three conditions: the tutor alone, the tutor plus meta-tutor and the tutor, meta-tutor and affective learning companion.

Keywords: Tutoring, meta-tutoring, learning strategies, affective learning companion, and affective physiological sensors.

1 Introduction

A *learning strategy* is a method used by a student for studying a task domain and doing exercises; a good learning strategy tends to increase the learning of students who follow it, whereas a poor learning strategy tends to decrease learning. A learning strategy is a kind of meta-strategy or meta-cognition. That is, it is knowledge about knowledge acquisition. For example, when studying a worked example, a good learning strategy is to self-explain every line of the example [1]. When working on a tutoring system that gives hints, a good learning strategy is to ask for hints when and only when one is unsure about what to do [2].

A perennial problem is that after students have mastered a learning strategy, they may still choose not to use it [3]. The AMT (Affective Meta-Tutoring) project tested whether an affective learning companion (ALC) could persuade students who were taught a learning strategy to continue using it after instruction in the learning strategy had ceased. The project built a system composed of four modules:

- An *editor*, which was used by students to take the steps needed to solve problems.
- A *tutor*, which taught students a problem-solving skill by giving hints and feedback on each step as the problem is being solved.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 84–93, 2014.

- A *meta-tutor*, which taught a learning strategy by giving hints and feedback about it as the students' used the tutor.
- An *affective learning companion*, having the goal of persuading students to use the learning strategy even after the *meta-tutor* is turned off.

The evaluation of the system focused on students' learning gains. We hypothesized that when ranked by learning gains, the three conditions we studied would exhibit this pattern:

$$tutor < meta\text{-}tutor + tutor < ALC + meta\text{-}tutor + tutor$$

We also tested whether students instructed with the affective pedagogical agent persisted in using the learning strategy when the meta-tutoring ceased.

This paper summarizes the AMT system and its evaluation, and concludes by discussing similar work. Many details are suppressed in order to keep the paper short, but can be found in the project publication referenced herein.

2 The Task Domain: System Dynamics Modeling

Recent standards for K-12 science and math education have emphasized the importance of teaching students to engage in modeling [4, 5]. Although "modeling" can mean many different things [6], we are interested in teaching students to construct models of systems that change over time (dynamic systems) where the model is expressed in a graphical language that is equivalent to sets of ordinary temporal differential equations. Stella (www.iseesystems.com), Vensim (vensim.com), Powersim (hwww.powersim.com) and similar graphical model editors are now widely used in education as well as industry and science. Much is known about students' difficulties with "systems thinking" and how it improves when students learn how to construct models [6]. The practical importance and strong research base motivated our choice of task domain.

However, even with kid-friendly editors [7], students still require a long time (tens of hours) to acquire even minimal competence in the task. Most science and math classes cannot afford to dedicate this amount of time to learning a modeling tool, so this path to deeper understanding of systems, too often, remains closed. One of the long-term practical goals of this work is to reduce the time-to-mastery from tens of hours to just an hour or two.

3 The AMT System

This section introduces the main parts of the AMT system: the editor, tutor, meta-tutor and ALC.

3.1 The Model Editor

The model editor had two tabs. One presented the problem to be solved, such as:

A bottle initially holds 100 bacteria. They grow quickly. On average, 40% of the bacteria reproduce each hour, each creating one new bacterium. Graph the number of bacteria in the bottle each hour.

The second tab was for constructing the model, which was done by creating nodes (see Figure 1). Each node represented a quantity. A node was defined in 3 steps, each achieved by filling out a form. The first step in defining a node was selecting a quantity from a large menu that included both relevant and irrelevant quantities. The second step (which was actually split into two forms) required qualitative planning and consisted of deciding whether the quantity was a numerical constant (e.g., the bacteria birthrate is 0.4), a quantity that was a function of other quantities (e.g., the number of bacteria births per hour is a function of the bacteria birthrate and the current bacteria population), or a quantity that changed by a specified amount per unit time (e.g., bacteria population increased each hour by the number of bacteria births per hour). The third step was stating a specific formula for calculating the quantity represented by the node.

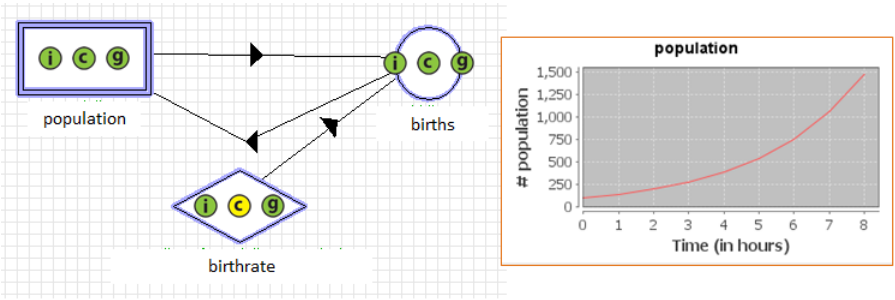


Fig. 1. Model for the bacteria problem, and graph of node "population"

When students had finished constructing a model, they could click on a button to execute it. This added a graph to each node, which students could see by clicking on the node (See Figure 1). Even if the tutoring system was turned off, students were given feedback on the correctness of their graphs. They could see both their graph and the correct graph. The little "g" circle inside the node icons (see Figure 1) was green if the students' graph matched the correct graph and red otherwise. Students could go on to the next problem only when all the nodes' graphs were correct.

3.2 The Tutor

When the tutor was turned on, students could get minimal feedback and bottom out hints. When they were filling out forms, the student could click on a button labelled "Check." This would color the student's entries either green for correct or red for incorrect. The color coding comprised minimal feedback on correctness. Students could also click on a button labelled "Solve it for me." It would finish filling in the form, but would also color the entries yellow. This comprised a bottom-out hint. In order to discourage overuse of the Solve-it-for-me button, when the student finished

editing a node, the status of their work was visible as colors on the little circles inside the node icons ("i" means input; "c" means calculation).

When the tutor was turned off, its feedback and hints were disabled. In particular, the Solve-it-for-me button was always disabled, and the Check button was disabled on all forms except the first one. The Check button was enabled during the first step because the system needed to know which node the student was trying to define so that it could associate a correct graph with the node.

3.3 The Meta-Tutor

When the meta-tutor was turned off, students tended to first define nodes for all the numbers in the problem statement, even if the numbers were irrelevant. Next, they tried to guess definitions of more nodes using keywords such as "initially," "increase" or "altogether." Sometimes they used methodical guessing. Indeed, some students seldom looked at the tab containing the text of the problem. These represent a few of the practices called "shallow modeling" in the literature [6]. The purpose of the meta-tutor is to prevent shallow modelling and encourage deep, thoughtful modeling.

Inspired by the success of the Pyrenees meta-tutor [8], our meta-tutor explicitly taught students a general goal decomposition method. For the students' benefit, we called it the Target Node Strategy and described it as follows:

1. Pick the quantity that the problem asks you to graph, create a node for it, and call it the *target node*.
2. Define the target node *completely*. If the node needs some inputs that haven't been defined yet, create those nodes but don't bother filling them in yet. Return to working on the target node, and don't stop until it's finished.
3. When the target node is finished, if there are nodes that have been created but not defined, then pick *any of them* as the new target node, and go to step 2. If every node has been defined, then the model is complete and you can execute it.

When the meta-tutor was on, it required the student to follow the Target Node Strategy. It also complained if the students overused the Solve-it-for-me or Check buttons, just as other meta-tutors do [9]. The meta-tutor also advised students on how to debug models (e.g., if several nodes have incorrect graphs, examine first those whose input nodes have correct graphs). We use the term "meta-strategy" to refer to this whole collection of strategic advice about how to use the tutor and the editor.

3.4 The Affective Learning Companion (ALC)

The main job of the ALC was to persuade students that the meta-strategy was worth their time and effort, and thus they should use it frequently not only when the meta-tutor was nagging them, but also when the meta-tutor and the ALC were turned off. To achieve this persuasion, we used both affect-based and motivation-based designs for the agent and its behavior. These designs are discussed in the order in which they were encountered by the student.

Following Dweck and others [10], all students began their training by reading a brief text introducing the "mind is a muscle" paradigm: the more you exercise your mind, the more competent you become. The ALC often referred to this concept, whereas the non-ALC interventions never mentioned it again.

After reading the "mind is a muscle" passage, students in the ALC condition first encountered the agent. The agent's appearance and initial behavior were designed to help establish rapport with the student. Following Gulz [11], the agent was a cartoon of a human. Following Arroyo et al. [12], its gender matched the gender of the student. Given the mixed results of D'Mello and Graesser [13], the agent display a fixed neutral expression. Following Gulz et al. [14], the agent introduced him/herself, and engaged the student in light conversation about the student's interests. The agent's dialogue turns were text, and the student's turns were selected from menus.

The student's next activity was to study a series of PowerPoint slides interwoven with simple exercises. These taught the basics of modeling and the user interface. This activity was the same for both the ALC intervention and the non-ALC intervention, and the agent was absent during it.

When the student had finished the introduction and was about to begin problem solving with the tutor, the ALC appeared and expressed enthusiasm about the upcoming challenges. It also reminded the student that the "mind is a muscle."

Once the student began solving a problem, the ALC "spoke" via a pop-up text approximately once a minute. If the student was practicing deep modeling frequently, then the agent remained silent.

When the agent "spoke," its message was selected based on log data and physiological sensor data that were interpreted by machine-learned models. The sensors were a facial expression camera and a posture-sensing chair. The sensor data were cleaned, synched and input to a regression model that predicted the student's emotional state. The emotional state and the output of the log data detectors drove a decision tree that selected one of the following 7 categories, whose message was then presented to the student:

1. *Good Modeling*: Students exhibit frequent deep modeling behaviors and low variation among affective states. ALC: "You really nailed it efficiently! It seems like you are using the strategy and that all your efforts are helping you to make strong connections in your brain. Nice work!"
2. *Engaged*: Students make few errors and show high level of excitement and confidence. ALC: "That's it! By spending time and effort verifying your answers and planning ahead as you use the strategy, your brain is creating more connections that will help you in your future modeling."
3. *New Understanding*: Students show some shallow behaviors without making too many errors, and some may show some frustration. ALC: "You're getting good at this. Planning ahead is the way to go. I can almost see the connections forming in your brain."
4. *Inconsistent*: Students make many shallow behaviors and show high level of frustration. ALC: "Remember to stay focused and use the strategy and your plan. Your actions seem to be inconsistent with the plan you picked earlier. If you

planned on having a <fixed value> node, then why are you trying to create a <function>? It's OK; sometimes it can be confusing; just remember to always try to do your best…"

5. *Guessing*: Students enter several answers before getting one correct, perform many shallow behaviors, and show low level of excitement: "Sometimes one must guess. But even if you've been guessing recently, try to figure out why the response that got green was correct. That way you can get there faster next time without guessing."
6. *Fluttering, Confused, Lost*: Students make many errors. While the student sometimes refers to instructions and the problem, the student only uses these features when stuck, not when planning the modeling activity. ALC: "You seem a little lost. Sometimes these activities can be confusing. Do you think you can go back to the strategy and use it to make a plan about the best way to spend your effort? This will probably help you make progress."
7. *Boredom*: Students make some errors and show consistently low level of interest. ALC: "If this activity seems boring, why not turn it into a game to make it more fun? For instance, do you think you can finish a node while getting green on your first try at every tab?"

The ALC messages quoted above were the ones presented initially. If the same message needed to be presented later, one of 10 short versions was presented instead.

When students finished a problem, a rectangular bar appeared alongside the agent in order to reify the student's meta-cognitive performance, following [15, 16]. The bar was divided into three segments that displayed the proportion of student actions that were deep (green), shallow (red) or neutral (yellow). The modeling depth bar was intended to shift students' motivational focus from correctness (the red/green coding of the tutor) to effort (the red/green coding of the bar). After the ALC explained what the bar meant, it presented a message based on a 6-way categorization that took into account the student's behavior throughout the solving of the problem [17]. The student was then prompted to begin the next problem.

When the training phase was completed, the ALC appeared for the last time and encouraged the student to continue to use deep modeling practice in the forthcoming transfer phase.

The ALC's messages turned out to be mostly motivational and meta-cognitive. The messages were designed "bottom up" by experienced human coders who were familiar with the affect and motivation literature. The messages were tailored to fit the student's state as the coders interpreted it rather than to cleave precisely to one affect/motivation theory or another.

However, the ALC did choose which message to present on the basis of the student's affective state, as detected by the sensors. As advocated by [18], some messages probably work best if they were delivered only in some affective states. For instance, criticizing the students' effort when they are frustrated may cause disengagement, but the same message delivered to a bored student might have a better chance at re-engaging them.

4 Evaluation

This section reports the outcomes (main results) of our experiments evaluating the meta-tutor (studies 3, 4 and 5) and the ALC (studies 6 and 7). Studies 1 and 2 were pilot studies that involved only the editor and the tutor, and will not be discussed here.

4.1 Methods

Procedure: All five experiments used the same procedure. There were two phases: A 75 minute training phase and a 30 minute transfer phase. During the training phase, all students studied PowerPoint slides which introduced them to system dynamics modeling, the model editor and the Target Node Strategy. They also engaged in a series of training problems of increasing complexity. The Check and Solve-it-for-me buttons were available to give them feedback and demonstrations, respectively, on each step in constructing a model. During the transfer phase, the tutor, meta-tutor and ALC were all turned off. Thus, the transfer phase allowed students to display both competence at system dynamics modeling and the Target Node Strategy.

Design: Students were randomly assigned to treatment groups. The treatment manipulation occurred only during the training phase and only while the students were solving problems. There were three treatment conditions: tutor alone; tutor + meta-tutor and tutor + meta-tutor + ALC.

Measures: The studies used basically the same measures, although there were improvements as the studies progressed. There were three types of measures, which were all calculated from log data:

- *Efficiency*: How much modeling were students able to complete in a fixed period of time?
- *Error rate*: How many mistakes did students make when defining a node? How often did they get green (correct) the first time they clicked the Check button?
- *Modeling depth*: Did students use deep or shallow modeling practices?
  - How frequently did students guess or otherwise "game the system?"
  - How frequently were their actions consistent with the Target Node Strategy?
  - How frequently did students refer to the problem statement?
  - How frequently did students refer back to the introductory PowerPoint slides?
  - How many irrelevant nodes did students create?
  - How many episodes were classified as deep by the log data detectors?

Participants: Because we aimed at evaluating affective interventions, we conducted the studies (except 6) in a classroom context, namely ASU summer schools for high school students. ASU summer school classes always had between 40 and 50 students each. Background questionnaires indicated that students varied in their mathematical preparation from Algebra I to Calculus. We attempted to deal with the high incoming variance using co-variants (studies 3, 4 and 5) and stratified sampling (studies 6 and 7).

Nonetheless, the high variance in incoming attributes and the limited number of participants resulted in our studies being underpowered, which partly explains why several tests presented below turned out to be statistically unreliable.

4.2 Results of Comparing Meta-Tutor + Tutor to Tutor Alone

Studies 3, 4 and 5, which are fully described in [19], evaluate the impact of meta-tutoring using two treatment groups. The experimental group had both the meta-tutor and tutor turned on, whereas the control group had the meta-tutor turned off leaving only the tutor active. Our three main hypotheses and their evaluations follow.

During the training phase, meta-tutoring should improve students' efficiency, error rate and depth of modeling. In all three studies, on almost all measures, the results were in the expected direction, but the differences were statistically reliable only about half the time. The results for efficiency were weakest, probably because guessing often took less time than thinking hard. On the whole, we conclude that meta-tutoring probably did improve training phase performance.

During the transfer phase, efficiency and error rate should be better for the meta-tutored group because they should have acquired more skill in modeling during the training phase. Although there were weak trends in the expected direction, only one of the depth measures showed a statistically reliable difference. We conclude that meta-tutoring did not improve transfer phase performance enough to be detectable.

During the transfer phase, the meta-tutored group should not use deep model practices more frequently than the control group because the meta-tutor merely nags; it is the job of the ALC to persuade students to keep using deep modeling practices. This hypothesis predicts a null result, which was observed with all measures in all experiments, but the low power prevents drawing any conclusion from the null results.

4.3 Results of Adding the ALC to the Meta-Tutor + Tutor

Study 6 evaluated a preliminary version of the ALC that only intervened between modeling tasks and was not driven by the physiological sensors. None of the Study 6 measures showed benefits for this preliminary ALC compared to using the system without the ALC. Unlike the other studies, this was a lab study with university students intended mostly to collect data for calibrating the physiological sensors.

Study 7 compared the complete system to the same system with the ALC turned off. Our findings were:

- During the training phase, the ALC group was better than the non-ALC group on all measures, although the differences were reliable on only half the measures.
- During the transfer phase, the two groups tied on all error rate and efficiency measures, suggesting that they both learned the same amount during training.
- Also during the transfer phase, the ALC group was not different from the non-ALC group in this use of deep modeling practices.

Our interpretation of the results of Study 7 is that the ALC probably acted like an improved meta-tutor. That is, during training, it caused students to use deeper

modeling strategies, which increased their efficiency and decreased their error rates, but did not apparently affect their learning very much, because their advantage over the comparison group did not persist into the transfer phase. Although the AMT project has made many contributions, this finding is perhaps the main result of the project.

5 Discussion

While our studies were being conducted, other related studies were being done. There are now 12 studies in the literature besides our own where an ALC acted somewhat like ours [20], and only 4 had reliable main effects. Of them, 3 studies used memorization tasks, and the fourth study confounded instructional information with the affective intervention. On the other hand, all 8 studies with null effects used complex tasks, as did our studies. It is tempting to hypothesize that ALCs work best with simple, short tasks perhaps because there are more frequent opportunities for interacting with the ALC between tasks.

Overall, the good news is that we have discovered improvements to meta-tutoring that increase the frequency of deep modeling practices when the meta-tutoring is operating. This is important because modeling is becoming a more central part of the math and science standards, and students have strong tendencies to use shallow modeling practices. Unfortunately, we have not yet found a way to get this improved performance to persist when the meta-tutoring is turned off.

Another piece of good news is that students were able to achieve adequate competence in constructing system dynamics models with only 75 minutes of training. This is nearly an order of magnitude faster than earlier work with high school students [6].

In one key respect, the ALC's intervention could be improved. Our hypothesis was that using the affect sensors and detectors would allow the ALC's messages to be presented at emotionally optimal times. However, we did not actually vary the time of the messages enough. This would be a good topic for future work.

Acknowledgements. This material is based upon work supported by the National Science Foundation under Grant No. 0910221.

References

1. Fonseca, B., Chi, M.T.H.: The self-explanation effect: A constructive learning activity. In: Mayer, R.E., Alexander, P. (eds.) The Handbook of Research on Learning and Instruction, pp. 296–321. Routledge, New York (2011)
2. Aleven, V., et al.: Help seeking and help design in interactive learning environments. Review of Educational Research 73(2), 277–320 (2003)
3. Hattie, J., Biggs, J., Purdie, N.: Effects of learning skills interventions on student learning: A meta-analysis of findings. Review of Educational Research 66, 99–136 (1996)
4. National, R.C.: A Framework for K-12 Science Education: Practices, Crosscutting concepts, and Core Ideas. National Academies Press, Washington (2012)

5. CCSSO, The Common Core State Standards for Mathematics (2011), http://www.corestandards.org (October 31, 2011)
6. VanLehn, K.: Model construction as a learning activity: A design space and review. Interactive Learning Environments 21(4), 371–413 (2013)
7. Metcalf, S.J., Krajcik, J., Soloway, E.: Model-It: A design retrospective. In: Jacobson, M.J., Kozma, R.B. (eds.) Innovations in Science and Mathematics Education: Advanced Designs for Technologies of Learning, pp. 77–115 (2000)
8. Chi, M., VanLehn, K.: Meta-cognitive strategy instruction in intelligent tutoring systems: How, when and why. Journal of Educational Technology and Society 13(1), 25–39 (2010)
9. Roll, I., et al.: Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. Learning and Instruction, 267–280 (2011)
10. Dweck, C.S., Leggett, E.L.: A social-cognitive approach to motivation and personality. Psychological Review 95(2), 256–273 (1988)
11. Gulz, A.: Benefits of virtual characters in computer-based learning environments: Claims and evidence. International Journal of Artificial Intelligence and Education 14(3), 313–334 (2004)
12. Arroyo, I., et al.: The impact of animated pedagogical agents on girls' and boys' emotions, attitudes, behaviors and learning. In: International Conference on Advanced Learning Technologies (ICALT 2011), Athens, Georgia (2011)
13. D'Mello, S., Lehman, B., Sullins, J., Daigle, R., Combs, R., Vogt, K., Perkins, L., Graesser, A.: A time for emoting: When affect-sensitivity is and isn't effective at promoting deep learning. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 245–254. Springer, Heidelberg (2010)
14. Gulz, A., Haake, M., Silvervarg, A.: Extending a teachable agent with a social conversation module – Effects on student experiences and learning. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 106–114. Springer, Heidelberg (2011)
15. Arroyo, I., et al.: Repairing disengagement with non-invasive interventions. In: Luckin, R., Koedinger, K.R., Greer, J. (eds.) Artificial Intelligence in Education, pp. 195–202. IOS Press, Amsterdam (2007)
16. Walonoski, J.A., Heffernan, N.T.: Prevention of off-task gaming behavior in intelligent tutoring systems. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 722–724. Springer, Heidelberg (2006)
17. Girard, S., Chavez-Echeagaray, M.E., Gonzalez-Sanchez, J., Hidalgo-Pontet, Y., Zhang, L., Burleson, W., VanLehn, K.: Defining the behavior of an affective learning companion in the affective meta-tutor project. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 21–30. Springer, Heidelberg (2013)
18. D'Mello, S.K., Graesser, A.C.: Dynamics of affective states during complex learning. Learning and Instruction 22, 145–157 (2012)
19. Zhang, L., et al.: Evaluation of a meta-tutor for constructing models of dynamic systems. Computers & Education (in press)
20. Girard, S., et al.: How can Affect be used to improve the Learning outcomes of Interactive Instructional Systems? (in prep.)

Identifying Learning Conditions that Minimize Mind Wandering by Modeling Individual Attributes

Kristopher Kopp[1], Robert Bixler[2], and Sidney D'Mello[1,2]

[1] Department of Psychology
[2] Computer Science, University of Notre Dame, South Bend, IN 46556, USA
{kkopp,rbixler,sdmello}@nd.edu

Abstract. The propensity to involuntarily disengage by zoning out or mind wandering (MW) is a common phenomenon that has negative effects on learning. The ability to stay focused while learning from instructional texts involves factors related to the text, to the task, and to the individual. This study explored the possibility that learners could be placed in optimal conditions (task and text) to reduce MW based on an analysis of individual attributes. Students studied four texts which varied along dimensions of value and difficulty while reporting instances of MW. Supervised machine learning techniques based on a small set of individual difference attributes determined the optimal condition for each participant with some success when considering value and difficulty separately (kappas of .16 and .24; accuracy of 59% and 64% respectively). Results are discussed in terms of creating a learning system that prospectively places learners in the optimal condition to increase learning by minimizing MW.

Keywords: engagement, mind wandering, affect, machine learning.

1 Introduction

Advances in research on intelligent tutoring systems (ITSs) have recently intertwined aspects of the cognitive sciences with the affect sciences [1,2,3,4]. ITSs have evolved from systems that emphasize modeling student cognition [5,6] to systems that detect and respond to student affect as well [7,8,9]. One related area of interest is learner engagement. Engagement has been defined as a state of involvement in some activity or task with focused attention and intense concentration [7]. Engagement is a necessary condition to learning since learners have to attend to information in order to learn. It is not uncommon, however, for students to experience involuntary lapses in attention and suddenly realize that they were thinking about things totally unrelated to the learning content. Such *mind wandering* (MW) activities can be detrimental to learning [10,11], so it is important to develop systems that can sustain engagement by reducing the propensity of MW behaviors. The goal of this paper is to take steps towards developing a preventative system with the ability to place students in an optimal learning condition that would result in the least amount of MW based on measures of individual difference attributes.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 94–103, 2014.

1.1 Related Works

Recently, researchers have been interested in the relationship between affect and learning. D'Mello [2] conducted a meta-analysis of 24 studies that investigated the influence of student affective states on learning. Basic affective states, such as anger, fear, happiness, etc. [12], are considered to have specific and culturally unanimous qualities to them that make them rather distinguishable and easy to detect. However, it is the non-basic affective states (e.g., confusion, boredom and engagement) that were more frequent during learning with ITSs. For example, Craig and colleagues [13] identified significant and positive relationship between confusion and learning when interacting with an ITS.

Similarly, Baker and colleagues [7] observed the presence of non-basic affective states of students while they interacted with various ITSs. One of their main findings was that when boredom occurred, it was difficult to get the students to re-engage in the learning task. Instead, students experiencing boredom exhibited a propensity to engage in behaviors such as "gaming the system." They also found that confusion and engagement were the most prevalent states and better precursors to learning than boredom since those who chose to game the system do not learn.

The studies mentioned above are just a few examples of research identifying affective states during interactions with ITSs and the different types of repercussions they can have. Research along these lines has led to the development of **Reactive** affect-sensitive ITSs that attempt to sense affective states that could have an effect on learning and respond accordingly [1], [14,15]. One of the early examples of this type of system is Affective AutoTutor [16] which detects specific emotions (i.e., boredom, confusion) based on conversational modeling, facial cues, and body language and alters the dynamics of the tutoring session to react to the learner through dialog moves designed to address specific affective states.

With respect to mind wandering, Drummond and Litman [17] attempted to identify episodes of "zoning out" while students were engaged in a spoken dialog with an ITS. Students were periodically interrupted to complete a short survey to indicate the extent to which they were focusing on the task (low zoning out) or on other thoughts (high zoning out). J48 decision trees trained on acoustic-prosodic features extracted from the students' utterances yielded 64% accuracy in discriminating high vs. low zone-outs. The next step in this line of research would be for the ITS to respond when zone-outs are detected. A system called GazeTutor [8] attempted this by using eye tracking to assess a lack of attention and responded with interventions to re-engage learners. Thus, based on affect detection methodologies, systems are able to identify and respond to affective states to increase learning.

1.2 The Current Project

An alternative to reacting to affective states as they arise is to implement **Proactive** strategies that attempt to create or foster affective states that would be beneficial for learning. Here, we focus on engagement since it is a necessary condition for learning. Engagement is considered to have three components: a cognitive, an affective, and a behavioral component [18]. The affective and behavioral components have been extensively studied in previous ITS research (e.g., [19,20]); hence, our present

emphasis is on the cognitive component, specifically momentary lapses of attention or MW which has been shown to have a detrimental influence on learning under various conditions [10].

Our approach is motivated by the assumption that engagement emerges from an intersection of factors related to the learning task itself (e.g., task difficulty), factors related to the perceptions of the learning activity (e.g., task value), and factors related to the individual performing the task (e.g., abilities and traits) [21]. The unique interaction will differ among individuals depending on their own unique traits. The purpose of our overall project is to investigate whether or not we can capitalize on this interaction and place students into an ideal learning condition (i.e., influenced by text and task factors) based on the factors related to the learner (i.e., abilities and traits) that would lead to the least amount of MW.

As an initial step in this direction, we first considered the possibility of using machine learning techniques to predict the learning condition that was optimal in terms of minimizing MW for a specific learner based on his or her attributes. To do this, we collected a large data set where students were asked to study about scientific research methods from instructional texts. During learning, students were asked to report incidents of MW using standard probe-based methods [10]. Each student was exposed to four conditions that varied in combinations of difficulty (easy or difficult) and value (high or low value) of the text. Students also completed multiple measures of individual attributes. Ideal conditions were identified for each student as defined by the least proportion of MW reports. Supervised machine learning was used to predict the ideal condition for each student using their individual attributes as features.

2 Data Collection

2.1 Participants

Undergraduate students (N = 187) from two U.S. universities participated for course credit. 105 students were recruited from a medium-sized private mid-western university while 82 were from a large public university in the mid-south. The average age was 19.7 years (SD = 2.65).

2.2 Texts and Task Context

Students learned from four different texts, on a computer screen, on research methods topics (i.e., experimenter bias, replication, causality, and dependent variables). The texts contained 1500 words on average (SD = 10) and were split into 30-36 pages. The difficulty manipulation consisted of presenting either an easy or a difficult version of each text. Texts were made more difficult by replacing words and sentences with more complex versions while retaining content, length, and semantics. The value manipulation was modeled after a common strategy used by instructors during review sessions before exams. Specifically, value was manipulated based on the weight assigned to each text on a subsequent posttest. Questions corresponding to the "high-value" texts counted three times more toward the test score than questions

for the "low-value" texts. Students were made aware of this before reading each text. Thus students saw all four texts with 1 text in each one of the 4 conditions: 2 (difficulty: easy vs. difficult) × 2 (value: high vs. low). The success of the manipulations was confirmed with self-reports of the perceived difficulty and perceived value of the texts (see [22]).

2.3 Measures

Mind Wandering was measured through auditory probes, a standard and validated method for collecting online MW report [10]. Nine pseudorandom pages in each text were identified as "probe pages." When a student encountered a probe page, an auditory probe (i.e., a beep) was triggered at a randomly chosen time interval 4 to 12 seconds from the time the page appeared. Students were instructed to indicate if they were MW or not by pressing keys marked "Yes" or "No," respectively. The MW rate for each text was then obtained by computing the proportion of "Yes" responses to probes.

Individual Attribute measures were collected for use as features in our models. The following measures were collected: (a) performance scores of the Nelson Denny self-paced *reading comprehension test* [23], (b) median sentence reading time of the Nelson Deny test as a measure of *reading fluency*, (c) performance on the *reading span* test as a measure of *working memory ability* [24], (d) *interest in research methods*, measured using a Topic Interest Scale adapted from Linnenbrink-Garcia et al. [25], (e) the Boredom Proneness scale measured the participant's *trait behavior of general boredom* [26], (f, g) the Academic Boredom Survey [27] measured traits specific to *boredom in academic situations* when overwhelmed and underwhelmed (considered separately), (h) self-reported ACT/SAT scores from each participant as a measure of *scholastic aptitude*, and (i) pretest performance on an assessment of the target concepts as *prior knowledge*. Scores of all measures were standardized by school to alleviate any large discrepancies due to population differences between schools.

2.4 Procedure

First, students filled out a brief demographic survey and completed the Nelson Denny test. Second, students completed one of two multiple choice pretests (counterbalanced between pre and posttest across all students) comprised of 24 deep-reasoning questions. Students were then given the topic interest measure. Students next received instruction on the learning task and how to respond to the MW probes based on instructions taken from previous studies [28]. All students studied four texts (one at a time) for an average of 32.4 mins (SD = 9.09) on a page-by-page basis, using the space bar to navigate forward. The name of the topic and the corresponding weight of the test questions (value manipulation) were explicitly presented before each text. After students studied all four texts, they were presented with the remaining 24 item posttest. They then completed several additional measures: the boredom proneness scale; the academic boredom survey; and a reading span test.

3 Supervised Classification

Our principal goal was to assess our ability to place a student in a learning condition that would result in the least amount of MW reports. Each data point corresponded to one participant and was labeled with the conditions (difficulty and value) of the text with lowest rate of MW resulting in 187 data points. We then attempted to predict this optimal condition using nine measures of individual attributes as features using supervised machine learning.

3.1 Model Building

The WEKA machine learning software tool's [29] implementations of 34 machine learning algorithms were used to build models predicting which text condition (difficulty and value) led to optimal values of MW reports. There were two additional parameters for the classification task. The first parameter was a *threshold* for the difference between the standardized MW rate for the best and worst condition. For each data point, if this difference was above the threshold the data point was included in the data set. This allowed us to consider only those who reported a meaningful difference of MW between conditions. Values used for this threshold included 0, 0.25, and 0.5 standard deviations. The second parameter was the *classification task*. In addition to classifying across all four conditions, we collapsed difficulty across value and vice versa, resulting in two additional classification tasks: classifying difficult texts from easy texts, and high value texts from low value texts. This resulted in 408 models (4 classification task × 3 difference threshold × 34 classifiers) and the classifier that yielded the best model for each parameter combination was retained for analysis.

3.2 Model Validation

Models were evaluated using leave-one-student-out cross validation. The model was trained on all but one student, which was then used to predict the best text condition for the remaining student. This process was repeated until each student had been classified in this way. This method ensures generalizability across students because each of the training and testing sets are student-independent. The Kappa statistic was taken as the measure of classifier accuracy since it is less sensitive to variations in data distribution.

4 Results

We first assessed any differences to assigned conditions across all three classification tasks for the threshold value of 0. When considering all four conditions of difficulty × value, 26% of the students reported the least amount of MW in the easy and low value condition, 28% reported in the easy and high value condition, 28% reported in the difficult and low value, and 18% reported in the difficult and high value condition. When considering value and difficulty separately, 53% of the students reported the

least amount of MW in the low value condition and 57% in the easy condition when considering text difficulty. Thus, these differences indicate that there is not one single, optimal condition for all students.

4.1 Classification Accuracy

We first analyzed models that were built in an attempt to place individuals into one of the four ideal conditions (i.e., easy and low value, easy and high value, difficult and low value, difficult and high value) based on the nine individual attribute measures (i.e., features). As can be seen in Table 1, the best classification (i.e., highest kappa) occurred when we discriminated .25 standard deviations between the highest and least amount of MW reports between conditions with a Decision Stump classifier.

In addition to attempting to classify according to the four conditions, we collapsed MW reports across value and then difficulty and assessed each separately. As can be seen in Table 1, when collapsing across value conditions, the best classification occurred when we discriminated .25 standard deviations between the highest and least amount of MW reports between conditions with a Simple Logistic Classifier. Similarly, when collapsing across difficulty conditions, the best classification occurred when we discriminated .5 standard deviations between the highest and least amount of MW reports between conditions with a Decision Stump Classifier.

Table 1. Classification results

Classification Task	Classifaction Threshold	Kappa	Observed Accuracy	Expected Accuracy	*N*
Difficulty × *Value*	0	.03	.27	.25	187
	.25	**.11**	**.34**	**.26**	**141**
	.5	.06	.31	.26	98
Value	0	.01	.51	.50	187
	.25	**.16**	**.59**	**.51**	**141**
	.5	.13	.56	.50	98
Difficulty	0	.05	.54	.51	187
	.25	.05	.55	.52	141
	.5	**.24**	**.64**	**.53**	**98**

Note: The kappa value is calculated using the formula (Observed Accuracy - Expected Accuracy) / (1 - Expected Accuracy), where Observed Accuracy is equivalent to recognition rate and Expected Accuracy is estimated from the marginal probabilities in the confusion matrix.

4.2 Features

We next considered the correlations between the performance on individual attribute measure (i.e., features) and placement in the optimal conditions of the value and difficulty classification tasks. For value, the conditions were dummy coded as low = 0 and high = 1. For difficulty, easy = 0 and difficult = 1. As can be seen in Table 2, there are some similarities and some differences with respect to the features that correlate with optimal classification for each classification task. With regard to the

highest correlations, for value, for students who have a higher propensity to experience boredom during academic situations that are underwhelming, the low value condition would be the optimal condition for the least amount of mind wandering. On the other hand, for difficulty, students with the propensity to experience boredom during overwhelming situations would benefit from having more difficult text. Additionally, the topic interest measure shows that a student may benefit from a more difficult text if they have a high amount of interest in the topic.

Table 2. Correlations (pearson *r's*) between performance on the individual attribute measures (i.e., features) optimal conditions of classification task dummy coded for value (low = 0 and high = 1) and difficulty (easy = 0 and difficult = 1)

Individual Attribute	Value (n = 141)	Difficulty (n = 98)
Working Memory	.10	-.03
Academic Boredom (Overwhelmed)	-.03	.20
Academic Boredom (Underwhelmed)	-.16	-.04
General Boredom	-.05	.09
Prior Knowledge	-.03	-.09
Reading Fluency	.10	.09
Reading Comprehension	.01	.01
Topic Interest (Research Methods)	-.05	.18
Scholastic Aptitude	-.04	.01

5 Discussion

The negative influence of mind wandering (MW) on learning coupled with the frequency of MW suggest that educational technologies could benefit by prospectively selecting learning conditions to reduce the incidence of MW. As an initial step in this direction, our hypothesis was that it was possible to determine an optimal learning condition that would lead to a lowered rate of MW based on a relatively modest set of nine individual attributes.

There was not a single condition that was optimal for all students, which suggests that even though *on average* one condition might yield lower MW rates than others, assigning every student to the same condition is not an optimal strategy since individual differences matter. We attempted to capitalize on those differences and our results show that it is possible to determine the condition that leads to the lowest rate of MW for an individual by considering that individual's trait attributes. Removing students with stable MW rates across all conditions improved our kappas from .03, .01 and .05 to .11, .16 and .24, for difficulty × value, value, and difficulty, respectively. This method of participant removal is justified because a participant whose MW rate does not change across condition does not add any meaningful variability to model. Furthermore, individuals who do not have different rates of MW

across conditions could not possibly have their MW rate lowered by altering condition no matter their individual attributes.

We acknowledge that classification rates were modest, even for the best models. However, one needs to consider the difficulty of the task in that we are attempting to prospectively predict a task condition that yielded the lowest rates of MW from a set of sparse individual difference measures alone, despite the fact that MW is an extremely complex and elusive state that is likely influenced by numerous additional factors. Furthermore, we have some confidence in the generalizability of our results because we employed a leave-one-subject-out validation method and our data included students from a medium-sized private mid-western university and a large public mid-south university with very different characteristics.

The usefulness of this research depends on how it can act to influence future designs of ITSs that intend to increase learner engagement by minimizing off-task thought. It may be of interest for designers of these systems to be able to predict mind wandering behaviors from attributes of the learner in order to advance preventative technologies. From our results, it was difficult to accurately predict conditions when including students who did not deviate in their MW behaviors across conditions in a meaningful way. This work does show, however, that it is possible to predict optimal conditions for those who show some contrast of mind wandering behaviors between different learning conditions. It may be that ITSs would benefit from initially targeting those whose mind wandering behaviors are somewhat different under different learning conditions.

There were some limitations of this work. First, the method of tracking MW through auditory probes is subject to students providing an incorrect rate of MW. An incorrectly reported rate of MW would result in our models being trained on data which was not completely correct. This would ultimately make classification more difficult. However, many studies have used this method of measuring MW as there are no alternatives to tracking this highly internal phenomenon (see [10] for a review), so we are confident that we are adhering to state of the art methods. Second, these findings are based on a task that requires studying texts on research methods. Future studies may consider incorporating other topics and other modes of information delivery to ensure generalizability. Furthermore, the present study was conducted in a laboratory context, so replication in more ecological learning situations is warranted.

This paper reports a first step towards a proactive learning system to reduce the rates of MW. The present work demonstrated the ability to select the best condition of easy and difficult text or high and low value for a learner to have the lowest rate of MW based on the learner's individual attributes. Our approach generalizes to individuals due to the method of validation and the diversity of the students. The next step is to use the best models in a personalized learning environment that optimizes the potential for the least amount of mind wandering during a learning session by personalizing the experience based on the measures of individual differences. For example, for each learner, the environment can prescribe conditions that minimize MW. MW and learning associated with this personalized environment can then be compared to control conditions (e.g., randomly assigning learners to condition or assigning all learners to the condition that resulted in the lowest MW overall). Whether, the proposed approach outperforms these alternatives awaits further research.

Acknowledgment. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. Calvo, R.A., D'Mello, S.K.: Frontiers of affect-aware learning technologies. IEEE Intelligent Systems 27(6), 86–89 (2012)
2. D'Mello, S.K.: A Selective Meta-analysis on the Relative Incidence of Discrete Affective States during Learning with Technology. Journal of Educational Psychology (2013)
3. D'Mello, S.K., Graesser, A.C.: Feeling, Thinking, and Computing with Affect-Aware Learning Technologies. In: Calvo, R.A., D'Mello, S.K., Gratch, J., Kappas, A. (eds.) Handbook of Affective Computing. Oxford University Press (in press)
4. Picard, R.W.: Affective Computing. MIT Press, Cambridge (1997)
5. Graesser, A.C., Olney, A., Haynes, B.C., Chipman, P.: AutoTutor: A cognitive system that simulates a tutor that facilitates learning through mixed-initiative dialogue. In: Forsythe, C., Bernard, M.L., Goldsmith, T.E. (eds.) Cognitive Systems: Human Cognitive Models in Systems Design. Erlbaum, Mahwah (2005)
6. VanLehn, K., Graesser, A.C., Jackson, G.T., Jordan, P., Olney, A., Rosé, C.P.: When are tutorial dialogues more effective than reading? Cognitive Science 31(1), 3–62 (2007)
7. Baker, R., D'Mello, S., Rodrigo, M., Graesser, A.: Better to be frustrated than bored: The incidence, persistence, and impact of learners' cognitive–affective states during interactions with three different computer-based learning environments. International Journal of Human-Computer Studies 68(4), 223–241 (2010)
8. D'Mello, S., Olney, A., Williams, C., Hays, P.: Gaze tutor: A gaze-reactive intelligent tutoring system. International Journal of human-computer studies 70(5), 377–398 (2012)
9. Woolf, B., Burleson, W., Arroyo, I., Dragon, T., Cooper, D., Picard, R.: Affect-aware tutors: Recognizing and responding to student affect. International Journal of Learning Technology 4(3/4), 129–163 (2009)
10. Mooneyham, B.W., Schooler, J.W.: The costs and benefits of mind-wandering: A review. Canadian Journal of Experimental Psychology/Revue Canadienne de Psychologie Expérimentale 67(1), 11 (2013)
11. Szpunar, K.K., Moulton, S.T., Schacter, D.L.: Mind Wandering and education: From the classroom to online learning. Frontiers in Psychology, 4 (2013)
12. Ekman, P.: An argument for basic emotions. Cognition & Emotion 6(3-4), 169–200 (1992)
13. Craig, S., Graesser, A., Sullins, J., Gholson, B.: Affect and learning: An exploratory look into the role of affect in learning with AutoTutor. Journal of Educational Media 29(3), 241–250 (2004)
14. Baker, R.S.J.d., Gowda, S.M., Wixon, M., Kalka, J., Wagner, A.Z., Salvi, A., Aleven, V., Kusbit, G., Ocumpaugh, J., Rossi, L.: Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. In: Proceedings of the 5th International Conference on Educational Data Mining, pp. 126–133 (2012)
15. Conati, C., Maclaren, H.: Empirically building and evaluating a probabilistic model of user affect. User Modeling and User-Adapted Interaction 19(3), 267–303 (2009)
16. D'Mello, S.K., Jackson, G.T., Craig, S.D., Morgan, B., Chipman, P., White, H., Person, N., Kort, B., el Kaliouby, R., Picard, R., Graesser, A.C.: AutoTutor detects and responds to learners affective and cognitive states. In: Workshop on Emotional and Cognitive Issues at the International Conference on Intelligent Tutoring Systems (2008)

17. Drummond, J., Litman, D.: In the Zone: Towards Detecting Student Zoning Out Using Supervised Machine Learning. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part II. LNCS, vol. 6095, pp. 306–308. Springer, Heidelberg (2010)
18. Fredricks, J.A., Blumenfeld, P.C., Paris, A.H.: School engagement: Potential of the concept, state of the evidence. Review of Educational Research 74(1), 59–109 (2004)
19. Pekrun, R., Linnenbrink-Garcia, L.: Academic emotions and student engagement. In: Handbook of Research on Student Engagement, pp. 259–282. Springer, US (2012)
20. Gregory, A., Allen, J.P., Mikami, A.Y., Hafen, C.A., Pianta, R.C.: Effects of a professional development program on behavioral engagement of students in middle and high school. Psychology in the Schools 51(2), 143–163 (2014)
21. Snow, C.: Reading for understanding: Toward an R&D program in reading comprehension. RAND Corporation, Santa Monica (2002)
22. Mills, C., D'Mello, S.K.: How Do Extrinsic Value and Difficulty Impact Engagement: An Experimental Approach (in prep.)
23. Brown, J.I.: The Nelson-Denny Reading Test (1960)
24. Daneman, M., Carpenter, P.A.: Individual differences in working memory and reading. Journal of Verbal Learning and Verbal Behavior 19(4), 450–466 (1980)
25. Linnenbrink-Garcia, L., Durik, A.M., Conley, A.M., Barron, K.E., Tauer, J.M., Karabenick, S.A., Harackiewicz, J.M.: Measuring Situational Interest in Academic Domains. Educational and Psychological Measurement 70(4), 647–671 (2010)
26. Farmer, R., Sundberg, N.D.: Boredom proneness–the development and correlates of a new scale. Journal of Personality Assessment 50(1), 4–17 (1986)
27. Acee, T.W., Kim, H., Kim, H.J., Kim, J.I., Chu, H.N.R., Kim, M., Cho, Y., Wicker, F.W.: Academic boredom in under-and over-challenging situations. Contemporary Educational Psychology 35(1), 17–27 (2010)
28. Feng, S., D'Mello, S., Graesser, A.C.: Mind Wandering while reading easy and difficult texts. Psychonomic Bulletin & Review, 1–7 (2013)
29. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter 11(1), 10–18 (2009)
30. Kononenko, I.: Estimating attributes: Analysis and extensions of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) ECML 1994. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994)

Investigating the Effect of Meta-cognitive Scaffolding for Learning by Teaching

Noboru Matsuda[1], Cassondra L. Griger[1], Nikolaos Barbalios[1], Gabriel J. Stylianides[3], William W. Cohen[2], and Kenneth R. Koedinger[1]

[1] Human-Computer Interaction Institute, Carnegie Mellon University, USA
{mazda,grigercl,nbarba,krk}@cs.cmu.edu
[2] Machine Learning Department, Carnegie Mellon University, USA
wcohen@cs.cmu.edu
[3] Department of Education, University of Oxford, Oxford, UK
gabriel.stylianides@education.ox.ac.uk

Abstract. This paper investigates the effect of meta-cognitive help in the context of learning by teaching. Students learned to solve algebraic equations by tutoring a teachable agent, called SimStudent, using an online learning environment, called APLUS. A version of APLUS was developed to provide meta-cognitive help on what problems students should teach, as well as when to quiz SimStudent. A classroom study comparing APLUS with and without the meta-cognitive help was conducted with 173 seventh to ninth grade students. The data showed that students with the meta-cognitive help showed better problem selection and scored higher on the post-test than those who tutored SimStudent without the meta-cognitive help. These results suggest that, when carefully designed, learning by teaching can support students to not only learn cognitive skills but also employ meta-cognitive skills for effective tutoring.

Keywords: Learning by teaching, teachable agent, SimStudent, Algebra equation solving, meta-cognitive help.

1 Introduction

The effect of learning by teaching has been well known [1, 2] in many disciplines for diverse student populations and skill levels. Many empirical studies observe that when students tutor each other, not only tutees but also tutors learn—often called the *tutor-learning effect*. Yet it is only recently that researchers have started to investigate why and how students learn by teaching. This scholarly development is largely due to the growing maturity of advanced learning technologies that allow students to interactively tutor a synthetic peer, commonly called a *teachable agent* [3]. The teachable agent technology allows researchers to collect detailed interaction data to understand the relationship between tutoring activities and the tutor-learning outcome [4, 5].

Learning by teaching is a complicated phenomenon that includes many factors to be considered, which are often hard to control [2, 6]. Therefore, researchers conduct exploratory studies that focus on particular aspects of tutor learning and the functionalities of the learning by teaching environment. The current paper focuses on the effect of the *meta-cognitive help* for learning by teaching.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 104–113, 2014.

Biswas et al. examined the effect of the meta-cognitive assistance for tutor learning [7]. Students taught Betty's Brain, the teachable agent, about river ecosystems. There was a mentor agent who provided both cognitive help (e.g., corrective feedback on the errors that Betty's Brain made on the quiz) and meta-cognitive help (e.g., how to gauge what Betty's Brain knows about the river ecosystems). In the classroom study, they found no effect of the mentor agent on tutor learning. In the current study, however, since students need to learn both procedural skills and conceptual knowledge, we might see different effect of the meta-cognitive help.

Walker et al. [8] compared "adaptive" and "fixed" meta-cognitive help for tutor learning in Algebra equations where pairs of students teach each other. The "adaptive" help was contextualized, whereas the "fixed" help was provided randomly. The results from a classroom study showed that the "adaptive" meta-cognitive help is more effective for tutor learning than the "fixed" meta-cognitive help. The current study will build on these findings to further investigate the effect of the meta-cognitive help for tutor learning.

Our previous studies showed that students often failed to select appropriate problems to tutor [4]. Therefore, we hypothesized that providing students with scaffolding on how to select problems to tutor would facilitate tutor learning. On the other hand, to select appropriate problems to tutor, students need to gauge their tutees' proficiency. Therefore, we further hypothesized that providing students with scaffolding on how to gauge tutee's proficiency would amplify the effect of the meta-cognitive help on problem selection, which would result in better tutor learning. To test these hypotheses, we used the online learning environment (called APLUS) where students learn to solve algebra equations by teaching a teachable agent called SimStudent.

2 SimStudent and APLUS

2.1 SimStudent

SimStudent is a computational model of learning, realized as a machine-learning agent, which can be interactively tutored. It is implemented with various AI techniques including programming by demonstration in the form of inductive logic programming, version space, and iterative-deepening search [4].

SimStudent learns cognitive skills in the form of production rules by generalizing *positive examples* (showing when to apply a particular skill, e.g., adding a constant to both sides) and *negative examples* (showing when not to apply a particular skill).

When SimStudent is used as a teachable agent, the affirmative feedback from the student for steps performed by SimStudent and the steps demonstrated by the student as a hint become positive examples, whereas the negative feedback becomes negative examples. A hint from the student on how to perform the next step also becomes a positive example. The next section provides details about the interaction between the student and SimStudent. See [4] for more technical details.

2.2 APLUS

Figure 1 shows an example screenshot of APLUS. To teach SimStudent, shown as an avatar (g), a student enters an equation in the first row, e.g., 2x+4 = 2 (c). When a

problem is entered, SimStudent attempts to solve it step-by-step (d) by applying already learned productions. If SimStudent was able to perform a step, it asks the student about the correctness of the step performed. The student then provides *yes/no feedback*. When the student's feedback is negative (i.e., "no," which means the student thinks that the step performed by SimStudent is incorrect), then SimStudent makes another attempt, if able.

If there is no production applicable to perform a step, SimStudent asks the student what to do for the next step. The student provides help by actually performing the next step in the tutoring interface (i.e., entering the text in the next empty cell) (e).

Resources are available at the top left corner of the APLUS interface for students to review in order to prepare for tutoring (b). The [Introduction Video] tab shows a 10-minute video clip explaining how to use APLUS. The [Unit Overview] tab summarizes how to solve equations with worked-out examples. It also shows suggestions of problems to be used for tutoring. The [Examples] tab shows worked-out examples in the tutoring interface with detailed explanations about how to perform each step.

The quiz has four sections ordered by difficulty: 1 one-step equation, 3 two-step Equations, 4 equations with variables on both sides, and Final Challenge that has 8 equations with variables on both sides. Quiz sections are "locked" until the previous section is passed (i.e., all equations in the section are solved correctly). When the

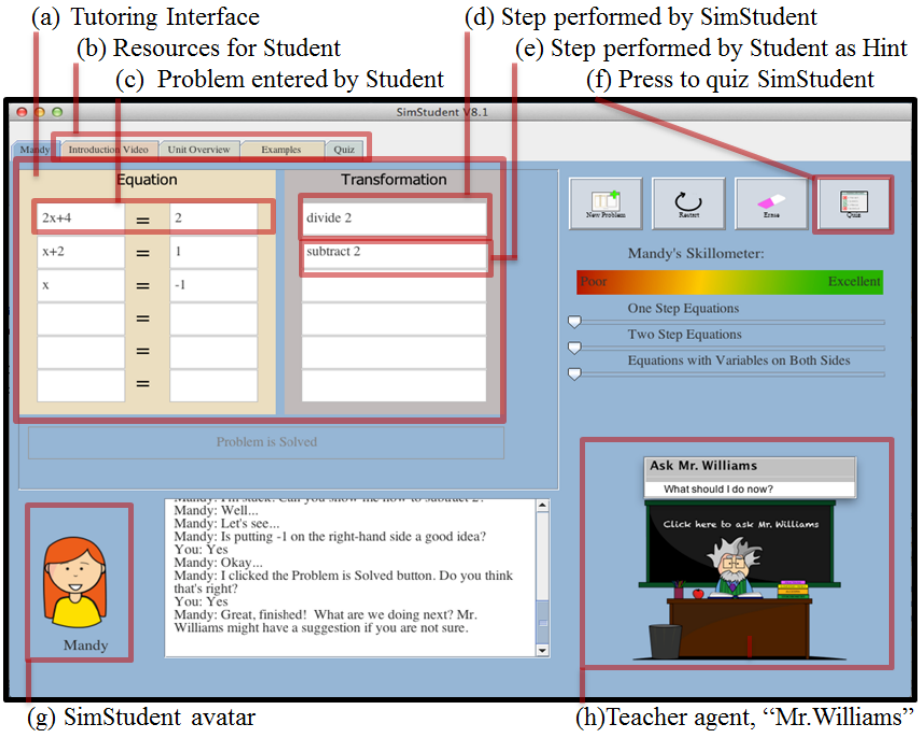


Fig. 1. Annotated sample screenshot of APLUS

student quizzes SimStudent (f), SimStudent attempts to solve quiz problems by applying learned productions. Mr. Williams, the teacher agent shown in the lower right corner (h), then summarizes SimStudent's performance on the quiz. The student can review the exact solutions made by SimStudent one by one in the tutoring interface.

In the meta-tutor version of APLUS, students can click on Mr. Williams to ask him for help. The next section explains details about the meta-cognitive help.

2.3 Meta-tutor with Meta-cognitive Help

In a version of APLUS, Mr. Williams performs as a *meta-tutor* who provides meta-cognitive help when asked. For the current version, two types of meta-cognitive help are available: (1) the *quiz help* suggests to students when to quiz their SimStudent and explains why (e.g., "It's a good strategy to quiz Mandy, because it would help you to understand what Mandy already knows. Click the Quiz button."), and (2) the *problem help* suggests to students what problem should be tutored next and explains why (e.g., "Since Mandy was wrong on the quiz, you may want to give 4y-8=10 to Mandy.").

Meta-tutor's help is thus available only when a problem is completed or a quiz is done. When the student asks for help by clicking on Mr. Williams, Mr. Williams shows only one menu item saying, "What am I supposed to do now?" (Figure 1-h); otherwise, Mr. Williams says, "You should complete the problem."

We use a model-tracing technique [9] to control the meta-tutor. That is, we have a (meta)cognitive model of how to tutor SimStudent, written as a set of production rules. Each production has associated hint messages. A student's tutoring activities are model-traced using the (meta)cognitive model so that when the student asks for help, the meta-tutor can provide just-in-time suggestions. Currently, there are six production rules: three for quiz and three for problem help.

3 Evaluation Study

3.1 Research Questions and Hypotheses

The goal of the evaluation study was to understand the effect of the meta-cognitive help provided by the meta-tutor. In particular we address the following two research questions: (1) Does the meta-tutor providing the quiz and problem help facilitate tutor learning? (2) If so, how does each type of help affect tutor learning?

We hypothesized that selecting problems based on the quiz results is an effective strategy, because it allows students to address specific weaknesses of their SimStudent's learning. Therefore, providing a meta-cognitive hint on problem selection based on quiz results should facilitate tutor learning—the *problem hint* hypothesis. To make the quiz-based problem selection work, students need to quiz SimStudent with appropriate timing. Thus, we also hypothesized that a meta-cognitive hint on when to quiz, in combination with the problem hint, should further facilitate tutor learning—the *quiz hint* hypothesis.

3.2 Methods

A classroom (in-vivo) study in the normal Algebra I classes at an urban public middle school in Pittsburgh, Pennsylvania was conducted with assistance of Pittsburgh

Science of Learning Center. The study was a randomized controlled trial with two conditions. The Meta-Tutor condition used the version of APLUS with the meta-tutor described in section 2.3. The baseline condition used the basic version of APLUS, which also had Mr. Williams but it did not provide the meta-cognitive help.

The study was five 42 minutes classroom periods over five consecutive days. On the first day, all students took an online pre-test (section 3.4) and then watched the introduction video available in APLUS. Students were then randomly assigned to a study condition. On the second through the fourth day, students used the assigned version of APLUS. On the fifth day, students took an online post-test.

Students were told that their goal was to have SimStudent pass the quiz, and SimStudent must learn how to solve equations with variables on both sides to pass the quiz (which is also mentioned in the Unit Overview). We will therefore call equations with variables on both sides as the *target equation* hereafter.

3.3 Participants

One hundred seventy-three (173) 7th through 9th grade students in nine Algebra-I classes participated in the study. A classroom-level randomization was applied to eight classes, and a within-class randomization for the remaining class. Out of those 173 students, 151 were present in the class on the first day and took the pre-test, 127 participated all three days for tutoring SimStudent, and 121 took the post-test.

As the result, 112 out of 173 students took both pre- and post-tests and participated in all three days of tutoring sessions. Those 112 students (53 in the Meta-Tutor condition and 59 in the Baseline condition) are included in the following data analysis. No other criteria for inclusion were used.

3.4 Measure

The online test consisted of two parts—Procedural Skill Test and Conceptual Knowledge Test. The *Procedural Skill Test* consisted of three sections: (1) The equation section had 10 equation problems with four one-step equations, two two-step equations, and four target equations. (2) The effective next step section had two problems each showing an equation with four options for a next step: add or subtract a term from both sides, or multiply or divide both sides by a constant. Students were asked to indicate whether each option was correct or not. (3) The error detection section had three problems each showing an incorrect solution for a given equation with multiple intermediate steps that contained one (and only one) incorrect step. Students were asked to identify the incorrect step. 53% (8 out of 15) of Procedural Skill Test items were about the target type of equation (with variables on both sides).

The *Conceptual Knowledge Test* consisted of 24 true/false items with seven items asking about variable terms, six items asking about constant terms, six items asking about like terms, and five items asking about equivalent terms.

After the study, the reliability of the test items was evaluated using Cronbach's alpha. For the Procedural Skill Test, the equation section showed $\alpha = .87$, the effective next step section had $\alpha = .76$, and the error detection section had $\alpha = .57$. Due to the low reliability index, we decided to exclude the error detection section from the analysis (and refer the average of other two sections as the score for the Procedural Skill Test). For the Conceptual Knowledge Test, $\alpha = .89$.

Table 1. Means (and standard deviations) for the Conceptual Knowledge Test (CKT) and the Procedural Skill Test (PST) by condition

	CKT		PST	
	Pre-test	Post-test	Pre-test	Post-test
Baseline	.43(.25)	.54(.21)	.69(.24)	.71(.25)
Meta-tutor	.43(.30)	.49(.22)	.71(.23)	.78(.19)
Total	.43(.27)	.52(.21)	.70(.24)	.74(.23)

In the analysis below, we also used the process data in addition to the learning outcome data (i.e., test scores). APLUS automatically logged detailed interaction between the student and the system included the problems used for tutoring, frequency of quiz, status of the resource and meta-tutor usage, and suggestions from the meta-tutor, etc. The correctness of steps suggested by SimStudent, and the accuracy of feedback and hints that students provided to SimStudent were also logged. Cognitive Tutor Algebra-1 [10] was embedded into the system to compute accuracy of feedback and hints for the purposes of logging.

4 Results

4.1 Test Scores

Table 1 shows the test scores. For the Procedural Skill Test, there was a reliable condition difference on the post-test scores—a one-way ANCOVA with the pre-test score as a covariate revealed a statistically significant difference on the post-test; $F(1,110) = 3.99$, $p < 0.05$. The effect size, Cohen's d, was 0.30. A post-hoc analysis revealed that only the Meta-Tutor condition showed a significant increase from pre- to post-test; *paired-t*$(52) = -2.96$, $p < 0.01$. No pre- and post-test difference was observed for the Baseline condition; *paired-t*$(58) = -0.68$, $p = 0.45$.

For the Conceptual Knowledge Test, there was no reliable condition difference observed, but the difference between pre- and post-test scores (when aggregated across all students in the two conditions) was statistically significant; $M_{pre} = .43$ ($SD = 0.27$) vs. $M_{post} = .52$ ($SD = 0.22$). A two-way repeated measures ANOVA with test-time (pre vs. post) as a within-subject variable and condition as a between-subject variable revealed a main effect of test-time; $F(1,110) = 18.32$, $p < 0.001$.

4.2 Meta-tutor Help

On average, students in the Meta-Tutor condition (N=53) asked Mr. Williams for help 5.5 times ($SD = 7.1$). The distribution was very skewed—11 (21%) students did not ask Mr. Williams at all, while 24 (45%) of students asked up to three times. Data also showed that different students apparently had different biases on the timing of hint requests—45% of students did not receive meta-tutor's message for the problem help at all, whereas 49% did not receive the message for the quiz help at all.

Despite the surprisingly low frequency of meta-tutor use, there was a reliable difference on the Procedural Skill post-test between conditions. Since the meta-tutor only provided quiz help and problem help, we predicted a difference in the way students quizzed SimStudent and selected problems to tutor that affected tutor learning.

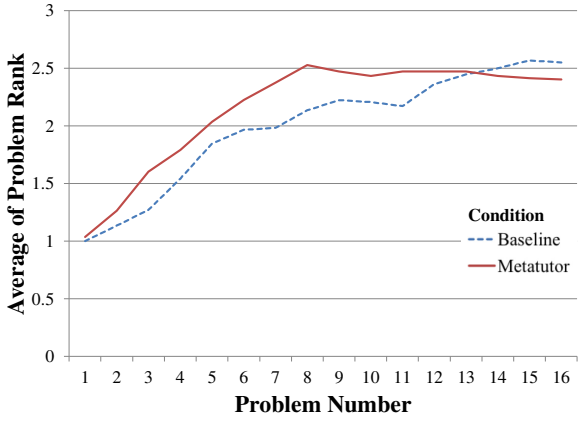


Fig. 2. The transition of problem types. The y-axis shows the problem "rank." The x-axis shows the number of problems tutored. On the 8th problem, the majority of the Meta-Tutor students had tutored the rank 3 problems, i.e. the target equations.

A regression analysis showed the number of help asked was not a reliable predictor of the Procedural Skill post-test score; $F(1,50) = 0.05$, $p = 0.82$. The probability of following meta-tutor's advice had no correlation with the post-test score either.

On average, students in each condition tutored 32.9±9.7 (Baseline) and 29.8±7.3 (Meta-tutor) problems. The number of problems tutored per se was not a reliable predictor of the Procedural Skill post-test. However, the type of problems (i.e., one- step, two-step, and target equations, which are equations with variables on both sides) tutored was a reliable predictor for the Procedural Skill post-test. The percent ratios of each problem type to all problems tutored were used as independent variables to predict the Procedural Skill post-test score. All three independent variables turned out to be statistically reliable predictors: $PST_{post} = -0.48 \times P_{ONE} + 0.99 \times P_{TWO} + P_{TGT} \times 0.63$ ($r^2 = 0.93$) where PST_{post} means the Procedural Skill post-test score; P_{ONE}, P_{TWO}, and P_{TGT} show the percent of one-step, two-step, and target equations tutored, respectively; for P_{ONE}, $F(1,109) = 902.97$, $p < 0.001$; for P_{TWO}, $F(1, 109) = 580.20$, $p < 0.001$; and for P_{TGT}, $F(1,109) = 117.98$, $p < 0.001$.

We then hypothesized that the meta-tutor's advice affected the way students selected problems, and in particular, students in the Meta-Tutor condition made quicker transitions from one-step equations to more advanced types of equations than the Baseline students. The data in Figure 2 support our hypothesis. In the figure, we "ranked" the types of problems that students used for tutoring: the rank is "1" for one-step equations, "2" for two-step equations, and "3" for target equations. The x-axis shows the chronological number of problems tutored. The y-axis shows the average "rank" of the problem tutored aggregated across all students in each condition. As we hypothesized, the Meta-Tutor condition showed a steeper slope that reached to 2.5 on the 8th problem, meaning that the majority of the students started to tutor target problems on and then after the 8th problem. On the other hand, it was around the 14th problem before the Baseline students started tutoring the target problems.

A regression analysis confirmed that our hypothesis was supported. Since two conditions did not reach the 2.5 rank-level in the same way, we computed the regression slope for the first 8 problems for the Meta-Tutor (MT) condition and the first 15 problems for the Baseline (BL) condition. The regression analysis revealed a significant difference between the slopes for the two conditions: $\beta_{MT} = 0.22$ vs. $\beta_{BL} = 0.11$,

F(1, 1298) = 33.60, $p < 0.001$. $r^2 = 0.25$. The Meta-Tutor students made a quicker transition from the entry-level problems to the target problems than the Baseline students.

We also examined the effect of quiz help. Since the meta-tutor (if asked) suggested quizzing SimStudent before tutoring, we hypothesized that the Meta-Tutor (MT) students showed a higher probability of starting the tutoring session with quiz than the Baseline (BL) students. This hypothesis was not supported. There was no difference in the probability of starting with quiz; M_{MT} = .08 (SD = .07) vs. M_{BL} = .07 (SD = .06), $t(107) = 1.98$, p = 0.88. We also computed the probability of "appropriate" tutoring actions, which, by definition, is the ratio of selecting problems based on the quiz results and quizzing after tutoring to all tutoring activities (which the meta-tutor also suggested upon a request). Again, there was no condition difference in their averages: M_{MT} = 0.33 (SD = .16) vs. M_{BL} = 0.33 (SD = .13), $t(104) = 0.22$, p = 0.83.

4.3 Accuracy of Tutoring

On average, 70% (SD = 22%) of Hints and 73% (SD = 10%) of Feedback that students provided to SimStudent were correct. To measure the overall accuracy of tutoring, we computed the Response Accuracy as 2×HA×FA/(HA+FA), where HA means the accuracy of Hints and FA means the accuracy of Feedback. The overall mean Response Accuracy was .70 (SD = .17).

It turned out that the Response Accuracy (RA) was a reliable predictor of the Procedural Skill post-test score (PST); F(1,109) = 10.7, $p = 0.001$; even when the PST pre-test score was controlled; F(1,109) = 56.9, $p < 0.001$; the model equation $PST_{post} = 0.55 \times PST_{pre} + 0.34 \times RA + 0.12$ ($r^2 = 0.56$). The Response Accuracy was also a reliable predictor of the Conceptual Knowledge post-test score (CKT); F(1,109) = 30.42, $p < 0.001$; even when the CKT pre-test score was controlled; F(1, 109) = 5.55, $p < 0.05$; the model equation $CKT_{post} = 0.40 \times CKT_{pre} + 0.26 \times RA + 0.16$ ($r^2 = 0.41$).

4.4 Resource Usage

There was no notable condition difference in the way students used the resources—in general, students did not use resources as often as we expected. **Table 2** shows average frequency and duration. Example problems were reviewed 29 times on average per student, but the average total duration on examples was only about 10 seconds per student. Regression analyses revealed that both frequency and duration of resource usage were not reliable predictors of the post-test score for the Procedural Skill and Conceptual Knowledge Tests.

Table 2. Average frequency (top) and duration (bottom) of resource usage

	Video	Unit Overview	Examples
Frequency	2.2 (3.4)	4.1 (8.3)	29.3 (33.6)
Duration	7.3s (38.2s)	6.6 (11.5s)	10.6s (13.1s)

5 Discussion

The data showed that the ability to tutor the *target* problems *correctly* (operationalized as the ratio of target problems tutored and the response accuracy as shown in sections 4.2 and 4.3) had a strong predictive power for the Procedural Skill post-test score, regardless of the availability of the meta-tutor. This finding is a replication of our previous study [4] that used the same version of APLUS that was used in the control condition of the current study.

The data also showed that the meta-cognitive help provided by the meta-tutor positively affected tutor learning. In particular, suggestions provided by the meta-tutor allowed students to make appropriate transition in tutoring from entry-level equations to the target equations. This finding supports the previous observation that learning by teaching is not an automated process, but rather requires careful scaffolding [4].

Despite the meta-tutor's assistance, many Meta-Tutor students failed to tutor a sufficient number of target equations. Ironically, it might be the case that the lack of teaching a sufficient number of target equations was due to the advice of the meta-tutor—since students were not able to manage tutoring the entry-level problems correctly, their SimStudents did not pass the entry-level quiz sections (i.e., one- and two-step equations), hence why the meta-tutor kept suggesting to students to continue teaching those entry level equations.

The challenge for the meta-tutor, therefore, is how to encourage students to teach a sufficient number of target equations with appropriate accuracy. For those students who have trouble teaching entry-level equations, the meta-tutor should provide assistance on skills to solve those equations (which are a prerequisite for learning the target equations). We have recently started to extend the meta-tutor (for our future studies) to provide *cognitive help* on feedback and hints that students provide to their SimStudents. With this extension, when students are not sure about the correctness of the steps performed by SimStudent, they will be able to ask Mr. Williams if their judgments are correct (before providing feedback to SimStudent). Additionally, when students do not know how to perform a next step for which SimStudent asks for help, students will be able to ask the meta-tutor what they should do next.

The meta-tutor should also encourage students to use resources more often as needed. For example, when students continue to ask for help on what to do next, then the meta-tutor might suggest that student should review the unit overview. If students repeatedly fail to have their SimStudents pass the quiz, then the meta-tutor might suggest that students should review example problems.

6 Conclusion

We found that the availability of the meta-tutor facilitated tutor learning on procedural skills for solving algebra equations. The data suggested that the meta-cognitive help given by the meta-tutor positively allowed students to select appropriate problems that affected both SimStudents' and hence students' learning.

Our data suggest that learning by teaching with meta-cognitive tutoring supports students in employing meta-cognitive skills on how to better tutor their peers that may not be available in traditional classroom instructions. At the same time, the data also

suggest that to make learning by teaching more effective, the learning environment must be carefully designed so that students can tutor their tutees appropriately, which involves scaffolding both on how to teach (meta-cognitive help) and what to teach (cognitive help).

Acknowledgements. The research reported here was supported by National Science Foundation Awards No. DRL-0910176 and DRL-1252440; and the Institute of Education Sciences, U.S. Department of Education, through Grant R305A090519 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education. This work is also supported in part by the Pittsburgh Science of Learning Center, which is funded by the National Science Foundation Award No. SBE-0836012.

References

1. Gartner, A., Kohler, M., Riessman, F.: Children teach children: Learning by teaching. Harper & Row, New York (1971)
2. Roscoe, R.D., Chi, M.T.H.: Understanding tutor learning: Knowledge-building and knowledge-telling in peer tutors' explanations and questions. Review of Educational Research 77(4), 534–574 (2007)
3. Brophy, S., et al.: Teachable agents: Combining insights from learning theory and computer science. In: Lajoie, S.P., Vivet, M. (eds.) Proceedings of the International Conference on Artificial Intelligence in Education, pp. 21–28. IOS Press, Amsterdam (1999)
4. Matsuda, N., et al.: Cognitive anatomy of tutor learning: Lessons learned with SimStudent. Journal of Educational Psychology 105(4), 1152–1163 (2013)
5. Leelawong, K., Biswas, G.: Designing Learning by Teaching Agents: The Betty's Brain System. International Journal of Artificial Intelligence in Education, 18–3 (2008)
6. Foot, H., et al.: Theoretical issues in peer tutoring. In: Foot, H., Morgan, M., Shute, R. (eds.) Children Helping Children, pp. 65–92. Wiley, New York (1990)
7. Biswas, G., et al.: Measuring Self-Regulated Learning Skills through Social Interactions in a teachable Agent Environment. Research and Practice in Technology Enhanced Learning, 123–152 (2010)
8. Walker, E., Rummel, N., Koedinger, K.R.: Adaptive support for CSCL: Is it feedback relevance or increased student accountability that matters? In: Proceedings of the International Conference on CSCL (2011)
9. Corbett, A.T., Koedinger, K.R., Hadley, W.S.: Cognitive tutors: From the research classroom to all classrooms. In: Goodman, P.S. (ed.) Technology Enhanced Learning: Opportunities for Change, pp. 235–263. Erlbaum, Mahwah (2001)
10. Ritter, S., et al.: Cognitive tutor: Applied research in mathematics education. Psychonomic Bulletin & Review 14(2), 249–255 (2007)

Togetherness: Multiple Pedagogical Conversational Agents as Companions in Collaborative Learning

Yugo Hayashi

Department of Psychology, Ritsumeikan University
56-1 Kitamachi, Toji-in, Kita-ku, Kyoto, 603-8577, Japan
y-hayashi@acm.org

Abstract. This study investigates the design of effective interaction using pedagogical conversational agents (PCAs) as companions in collaborative learning activities. Specifically, we focus on the use of embodied PCAs that evoke social awareness and engagement from human learners. In controlled experiments, paired collaborative learners were selectively accompanied by "peer-advisor" PCAs in a set of learning activities. Results show that learners who engaged with multiple PCAs gained a better understanding of target concepts than those using a single PCA. Furthermore, learners who engaged PCAs playing different collaborative roles (e.g., "mentor" and "expert") outperformed those who engaged PCAs without distinct roles. The implications of these results are explored and directions for future study are discussed.

Keywords: Pedagogical Conversational Agents; Collaborative Learning; Explanation Activities; Social Facilitation.

1 Introduction

As a result of Vygotsky's sociocultural learning theories and Lave's Situated learning theories, it is now widely accepted that group-based learning is an effective strategy for facilitating learning [1, 2]. Recent studies in CSCL have implemented artificial intelligence technologies in tutoring systems and show the benefits of pedagogical conversational agents (PCAs) [3, 4, 5, 6, 7, 8, 9]. One of the challenges is to design and develop PCAs that can effectively facilitate a learner's cognitive state. To accomplish such a goal, it is necessary to use interaction models and theories from cognitive and learning sciences [10, 11]. Studies show how effectively collaborative learning facilitates the understanding of new concepts depends on how the explanations are provided [12]. Based on this theory, the present study focuses on a collaborative learning where students attempt to explain a classroom-taught concept.

1.1 Supporting Learner-Learner Collaborative Learning with PCAs

Recently, studies have shown that conversational agents acting as educational companions or tutors can facilitate learning [5, 13]. Many computer-based tutoring systems

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 114–123, 2014.

use conversational agents [4], but it is not fully understood what kinds of support from these agents improve learner-learner collaborative learning. There are several issues that need to be solved when designing PCAs for this purpose, for instance, (1) interface and media design [7], (2) responses and feedback [14, 15], and (3) agents roles [6], and the design of the interaction [9].

Working in groups in a classroom provides an opportunity for learners to re-construct their knowledge and organize their ideas by themselves [16]. During such activities, it is important for learners to adopt a conversational manner known as "constructive interaction" [17]. When pair of learners is working on a problem together, constructive interaction is where one learner works on the problem by externalizing explanations and the other simply observes and questions his/her partner to facilitate meta-cognitive perspectives [18]. Despite the idealistic interaction model, collaborative activities are somewhat difficult, especially for new learners who are not used to expressing their thoughts or understanding other viewpoints. Assuming that learners experience high cognitive loads during explanation activities, paying attention to both their partners and third parties (e.g., computer agents) could be too difficult. It is difficult to make learners continually pay attention to a PCA in a human-human based collaborative task [19]. Holmes (2007) indicated that learning pairs ignored the presence of an agent and conducted the learning activities on their own [9]. Hayashi (2012) showed that some students who did not achieve high learning scores on a pair explanation activity did not consider the PCA's suggestions that were needed to construct an effective explanation [20].

There are several methods to make learners pay attention to a PCA's suggestions. For example, Kumar and Rose (2000) designed methodologies such as requiring the students to ask the PCA to initiate the learning session or move it forward (ask when ready strategy) and/or having the PCA interrupt their conversation (attention grabbing strategy) [3]. However, in human-human collaborative learning, it is important not to forcibly interrupt or disturb the learners' natural interaction and compromise their self-reliant learning activities. It is important to design the interface such that it naturally attracts the learners' attentions in a way that is psychologically consistent with their internal processes. In the next section, we present our methods for bringing attention to the PCA's suggestions in a psychologically consistent way and thus maintaining the learners' natural conversation.

1.2 Using Multiple Agents to Enhance a Tutor's Social Presence and Role

The present study uses the notion of "social facilitation" effects, taken from social psychology and dynamics research [21]. Studies in this field have shown that when one feels that he/she is engaging in an intellectual task with several members, it motivates him/her to work harder to satisfy other group members [22]. It is also well known that a person often feels social pressure from others when he/she is persuaded or informed of something by several group members during intellectual tasks. It is assumed that if a learner is collaborating with other learners and advised by several tutors, he/she may feel more pressure to include their comments into the learning activities. This study proposes a new methodology for creating a virtual group-based

learning platform that enhances the co-presence of the tutoring agents and uses multi-agent techniques to facilitate such social presence.

The first question to answer is whether agents can generate social pressure to make learners pay attention to them. A few studies in human-computer interaction have investigated the impact of social pressure from embodied agents. For example, Lee and Nass [23] examined the impact of visual representations of multiple agents on performance in a social dilemma task. Beck, Wintermantel, and Borg [24] investigated how social relationships with multiple agents affect persuasion. These studies imply that under some conditions, the use of multiple-agents can motivate and facilitate a change in human opinions. Therefore, the use of multiple PCAs may have the potential to exaggerate their presence and facilitate social pressures such as the need to work harder by causing the learners to consider the PCA's comments and suggestions. Based on the discussion above, the following hypothesis is presented:

H1: Multiple PCAs are more effective than a single PCA at facilitating their presence and motivating learners to engage in explanation activities and thus facilitate learning performance.

The next question that arises is what kind of roles the multiple agents should take during those interactions. It may be sufficient to increase the number of PCAs, however, it may also be necessary to design the character types and roles for each agent to provide more social presence. Many studies in collaborative problem solving and learning have pointed out the importance of member diversity and the beneficial effects of members taking different roles during those activities [18]. The diversity of tutors with different roles in group-based learning activities may also play an important role. If learners engage with multiple tutors that have different roles, it helps them to distinguish between the different tutoring content. If learners perceive agents as individual actors, this implies to them that there are different ways and viewpoints to consider when solving a problem. We may also find synergetic effects with regards to social pressure, as multiple members with diverse perspectives may create more impact and direct attention back to the learners than tutors with the same perspectives would. Past studies have shown that human learners can correctly understand the different roles that an agent may take. For example, Baylor and Kim [5] found that learners apply the same social rules and expectations to human-agent interactions as they do to human-human interactions. They pointed out that if agents are designed to have particular roles, learners could understand those roles as intended. Their results showed that when using agents with motivational characteristics and roles (motivator and mentor), the agents were more human-like and self-sufficiency was improved. They also found that using expertise characteristics (expert and mentor) facilitated learning outcomes along with positive feelings towards the agents such as credibility and had the best impact on learning and motivation.

Although this study showed that people can distinguish between an agent's roles and this led to different types of impressions during learning, they did not investigate different of combinations of the multi-party situation nor directly compare the effects of divisive PCA roles. This study focuses on the use of multiple PCAs with different roles versus no roles and investigates whether learners can perceive the variety of members in the group. It also looks at the effects of divisive PCA members.

H2: By splitting the roles of multiple agents, learners can more sensitively distinguish between the types of facilitations provided by the agents and thus can perform better interactions.

1.3 Aim of the Study

This study investigates the most effective way for PCAs to attract adequate attention in learner-leaner explanation activities and thus help them gain a deeper understanding of the problem. Based on the notions of learning science those stress the importance of learner-centered activities, the study focuses on a situation where a pair of learners' main activity is to collaboratively explain a key conceptual term to each other. During such activities, we investigated the use of a PCA that facilitates activities from a third-person point of view; for instance, providing (1) encouragement and (2) meta-cognitive suggestions. In this study, we investigate in particular the use of multiple PCAs that produce a social presence that could avoid the misuse of the agents and leads to more awareness of and attention to its suggestions and instructions. In addition, based on the studies of human-human collaborative problem solving, we investigate whether dividing the types of PCA facilitation can create a diversity of the group members and facilitate more aggressive behaviors to assist the explanation activities.

2 Method

2.1 Experimental Setting

To investigate our hypotheses, the present study set up an activity in which a pair of participants (called learners) participated. The learners consisted of one 118 students taking a psychology course who participated as a part of their coursework and were randomly assigned to three conditions that varied according to the PCAs' types of suggestions, number, and roles (see the section below for details). Learners were required to form explanations for a key term that was introduced in one of their course lectures, "figure ground reversal," and participated in groups of the same gender.

During the task, they used a desktop computer and a text-based chat application developed for this study (see Figure 1). All messages were sent and processed through the server. On the server side, all their text messages were analyzed by the PCA (details of this system are described in the next section). On the screen, there was a text area to input messages and a history of the conversation. In addition, a fundamental description of the key term was presented on their screen for basic guidance. Learners were instructed to explain the key term to each other by inputting text-based messages. As they proceeded with the task, a companion agent appeared on their screen and gave them suggestions as how to form a sufficient explanation (e.g., use examples or try to take turns), applauded them (e.g., for using important keywords), and/or gave back-channel feedback. They were also told that the agents would only participate as mentors to guide them and that their main activity was to discuss the key term and reach a mutual understanding of the key concept with their partner.

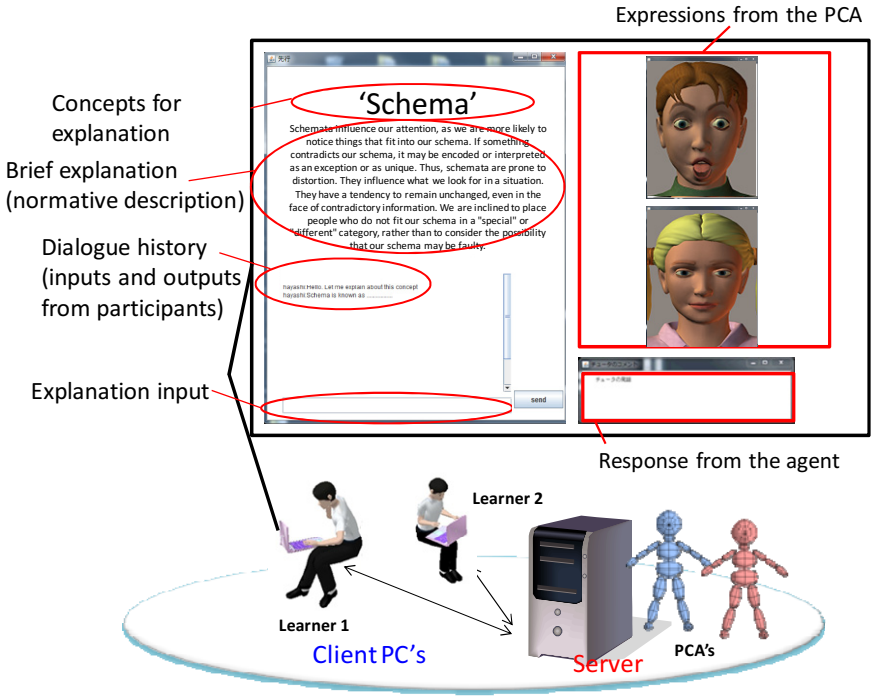


Fig. 1. The chat application (top) and experimental situation (bottom)

To analyze the learners' performance, they were required to take a pre- and post-test. In these tests, learners were asked to describe the meaning of the same technical words. As in Hayashi [19], the results were then compared to find out how the different conditions facilitated learners' learning of the concepts. In the comparison, descriptions were scored in the following way: one point was awarded for a wrong description or no description, two points for a nearly correct description, three points for a fairly correct description, four points for an excellent description, and five points for an excellent description with concrete examples. Two graders (with a correlation of 0.74) graded the answers and discussed their results before making any final decisions. The pre- and post-test scores were used to assess the degree of learning performance.

2.2 Structure of the PCA

The application was programmed in Java and designed as a server-client based network application using multi-cast processing methods. The system consisted of three sub-systems: (1) a chat interface, (2) server, and (3) agents. For the agent, three components comprised the system: (a) the input analyzer, (b) generator, and (c) output handler.

Input Analyzer. Important messages related to the explanation activities are stored in the keyword database. These keywords (phrases) were extracted from dialogues in Hayashi (2012) and each of their values was weighted by importance [19]. The system detects keywords from an inputted sentence and lists the patterns of those keywords. Next, the detected keywords were sent in an array to the generator along with information about the order of turn-taking. If no keywords were detected, a null result was returned.

Generator. The array list of keywords (phrases) transferred from the input analyzer is processed in the generator. The generator contains a rule-based system in the IF THEN format typically used in artificial intelligence. The system was originally developed in Java and uses forward chaining methods to constrain the keyword list patterns [15, 19]. When the rule-based matching is complete, one sentence is randomly chosen from the database to be the output sentence. The agent was designed to respond based on the related keywords. For example, if the system detects a constant rate of some keywords (phrases) related to 'explanations' (e.g., "for example", "this means", etc), then the system would generate (1) encouragement suggestions like "Yes!! Keep on like that and keep up with explaining. Try to use some original ideas too. Good job!!.". If a constant rate of keywords (phrases) related to 'trouble' (e.g., "don't know", "help", etc) were detected, then the system would generate (2) metacognitive suggestions from the database such as "I know this is a tough one. Why not explain it using examples from a daily situation."

Output Handler. Based on Hayashi (2012), the learners were given positive suggestions that were synchronized with facial expressions of the embodied agent [19]. Output text messages generated by the generator were next sent to the output handler. In this module, the system counted the number of words of the output messages and calculated the length of time needed to move the agent. Then the agent sent the text message along with the required motion time to each chat client system. The messages were given through chat dialogue while the virtual character moved its hands and lips. The agent graphics were designed by Poser8 (www.e-fronteir.com) and presented in frame-by-frame playback. A male or female agent was randomly used. Furthermore, a corresponding a male or female voice was generated using the Microsoft speech platform while the agents produced facial expressions.

2.3 Experimental Conditions

As explained in the previous section, the PCA used in this experiment produced prompts such as encouragement and meta suggestions. In various sessions, these two types of prompts were either both presented by one agent or presented separately by two PCAs. In the single condition (n = 38), learners engaged in the task using one PCA as a mentor. In the double condition (n = 42), learners engaged in the task using two PCAs. The PCAs in this condition did not have any distinct roles and both generated (1) encouragement and (2) meta suggestion prompts. To adjust for the amount of

information quantity given the single condition, only one PCA generated a message per turn. In the split double condition (n = 38), learners used two PCAs as in the double condition, however, each agent had a distinct role. One PCA only generated prompts based on (1) encouragement and the other PCA generated messages based on (2) meta suggestions. The PCA expressing encouragement was labeled as the "mentor" and learners were told that this PCA would give them comments based on their conversation. The PCA that gave meta suggestions was labeled as the "expert" and learners were told that this PCA would sometimes give directions and comments of a more sophisticated nature.

3 Results

In this section, we present results from three different dependent variables: (1) length of descriptions, (2) pre- and post-test scores, and (3) number of turn-takings. The first variable, description length, was measured by the length of the rows of the post-test (written on a sheet where one row consists of 20 words). The second variable, pre- and post-test scores consist of the graded results of those descriptions. The analysis of variables (1) and (2) indicate the performance of the task. The third variable, number of turn-taking, is the number of transaction between the learners and focuses on the process during the explanation task.

3.1 Length of Descriptions

A statistical analysis was performed using a 2 (evaluation test: pre-test vs. post-test) × 3 (PCA condition: single condition vs. double condition vs. split double condition) mixed-factor analysis of variance (ANOVA). There was no significant interaction between the two factors ($F(2, 115) = 0.18$, $p = .83$) and there were no main effects between conditions ($F(2, 115) = 0.22$, $p = .97$). However, there were differences between the pre- and post- test, where learners tended to write longer answers on the post-test ($F(2, 115) = 101.38$, $p < .01$). However, these performance results only show the increase in quantitative outputs of the learners. In the next analysis we see how these results change qualitatively (i.e., analysis done by grading).

3.2 Pre- and Post-Tests

The gain scores were calculated by subtracting the pre test scores from the posttest scores. An analysis was performed using a one-way between-factor analysis of variance (ANOVA). There was a significant interaction ($F(2, 115) = 3.254$, $p < .05$). Next, analysis from multiple comparisons indicates that the average of test scores of the split double condition and double condition was higher than that of the single condition ($p < .05$ for both). There were no differences between the split double condition and double condition ($p = .55$). These results show that the use of multiple PCAs is more effective than using only a single PCA, supports hypothesis H1.

3.3 Turn-Taking

Statistical analysis was performed using a one-way between-factor analysis of variance (ANOVA). There was a significant interaction ($F(2, 56) = 6.571$, $p < .01$). Next, analysis from multiple comparisons indicates that the average number of turns of the split double condition was higher than that of the double condition and the single condition ($p < .01$ for both). Results also show that the number of turns of the split double condition was higher than that of the single condition ($p < .01$). This result indicates that using multiple PCAs with different roles may facilitate the turn-taking process. This may be due to the effects of the divisiveness of the roles of PCAs, which brings better impact on its presence. The results show that using multiple PCAs significantly influences turn taking when suggestions are made from various roles/viewpoints. This result supports hypothesis H2.

4 Discussion

The analysis shows that the use of multiple agents outperforms learning performance when using single agents in a learner-leaner centered collaboration task. This shows that the methodology of using multiple agents can produce a stronger PCA social presence and thus reduce the learners' tendency to ignore them. Avoiding such a lack of attention to or misuse of the PCA has been a big problem when designing these systems [3, 14, 15]. It is also difficult not to interrupt the learners' natural interactions and scaffolding should be made in an implicit way. Using multiple agents can afford such implicit psychological impact and thus provide more social presence compared to the ordinary use of a single agent. Since the number of prompts from the PCA was controlled to be the same in all conditions, the only effects on the learner's experience were the presence of the PCAs. However, there are some issues that need to be studied in the future, such as the amount of time learners spent actually paying attention to the PCAs. We are now conducting more experiments and collecting eye movement data to find how frequently learners look at the PCAs under various conditions.

The results in the analysis also show that when using multiple PCAs, it is better to split their roles rather than mix the roles together. Splitting the roles of the PCA brings more variety to the group members and thus provides more PCA social presence. In addition, it may help the learners distinguish the types of content provided by the PCA. In this study, one agent (the mentor) was assigned to generate prompts based on keywords to provide learners with reflective thoughts about the keywords they were using. On the other hand, one agent (the expert) generated meta-suggestions and gave directions about how to think or make explanations. Such kinds of suggestions are useful when the learner is thinking what to put in a message or how to form explanations. Results from the conversational analysis show that learners using PCAs with different roles took more turns than PCAs with no distinct roles. This indicates that learners may have found it easier to capture the information provided from the PCA (expert) that gave directions on what to speak. On the other hand, where the learners interacted with PCAs with mixed roles, they may have been unable

or found it difficult to capture the messages that included directions and meta-suggestions. This point could also be investigated by further detailed analysis about when the learners looked at or responded to each PCA.

5 Conclusion

The present study investigated the most effective interaction design to evoke the presence of an embodied PCA on a multi-agent platform while creating social awareness and engagement with the learners. A controlled experiment was conducted to investigate the effects of using such PCAs and their roles during pedagogical activities. In the experiment, pairs of students collaboratively formed explanations about a key concept taught in the classroom and PCAs joined their activities as peer-advisors. Results of the experiment show that learners who engaged with the multiple PCA gained a higher understanding of the concept than learners using a single PCA. In addition, learners using PCAs with distinct of roles such as the meta-cognitive advisor (expert) and the emotional supporter (mentor) enhanced better interactions. The results lead to implications such as the possibility of using the multi-agent platform to facilitate social awareness and help learners gain a better understanding of target concepts. Furthermore, using different PCA roles (e.g., "mentor" and "expert") outperformed those who engaged PCAs having same roles in terms of amount of turn-taking activities thus facilitating explanation activities. The present study contributes to the knowledge about the design of PCAs that are effective at facilitating human-human explanation activities in learning. Future work includes the implementation of these findings to tutoring systems for use in classrooms and other learning situations.

Acknowledgements. This work was supported (in part) by 2012 KDDI Foundation Research Grant Program and the Grant-in-Aid for Scientific Research (KAKENHI), The Ministry of Education, Culture, Sports, Science, and Technology, Japan (MEXTGrant), Grant No. 25870910.

Reference

1. Vygotsky, L.S.: Mind in Society: The Development of Higher Psychological Processes. Harvard University (1978)
2. Lave, J., Wenger, E.: Situated Learning - Legitimate Peripheral Participation. Cambridge University Press (1991)
3. Kumar, R., Rose, C.: Architecture for building conversational agents that support collaborative learning. IEEE Transactions on Learning Technologies 4(1), 21–34 (2011)
4. Graesser, A., McNamara, D.: Self-regulated learning in learning environments with pedagogical agents that interact in natural language. Educational Psychologist 45(4), 234–244 (2010)
5. Baylor, A.L., Kim, Y.: Simulating instructional roles through pedagogical agents. International Journal of Artificial Intelligence in Education 15(1), 95–115 (2005)

6. Baylor, A.L., Ryu, J.: The API (Agent Persona Instrument) for Assessing Pedagogical Agent Persona. In: Lassner, D., McNaught, C. (eds.) Proc. World Conference on Educational Multimedia, Hypermedia and Telecommunications, pp. 448–451 (2003)
7. Gulz, A., Haake, M.: Design of animated pedagogical agents: A look at their look. International Journal of Human-Computer Studies 63(4), 322–339 (2006)
8. Kim, Y., Baylor, A.L., Shen, E.: Pedagogical agents as learning companions: The impact of agent emotion and gender. Journal of Computer Assisted Learning 23(3), 220–234 (2007)
9. Heidig, S., Clarebout, G.: Do pedagogical agents make a difference to student motivation and learning? Educational Research Review 6(1), 27–54 (2011)
10. Chi, M., Leeuw, N., Chiu, M., Lavancher, C.: Eliciting self-explanations improves understanding. Cognitive Science 18(3), 439–477 (1994)
11. Okada, T., Simon, H.: Collaborative discovery in a scientific domain. Cognitive Science 21(2), 109–146 (1997)
12. Shirouzu, H., Miyake, N., Masukawa, H.: Cognitively active externalization for situated reflection. Cognitive Science 26(4), 469–501 (2002)
13. Holmes, J.: Designing agents to support learning by explaining. Computers and Education 48(4), 523–547 (2007)
14. Moreno, R., Mayer, E.: Role of guidance, reaction, and interactivity in an agent-based multi-media game. Journal of Educational Psychology 97(1), 117–128 (2005)
15. Hayashi, Y.: Learner-support agents for collaborative interaction: A study on affect and communication channels. In: Proc. 10th International Conference on Computer Supported Collaborative Learning, pp. 232–239 (2013)
16. Salomon, G.: Distributed Cognition: Psychological and Educational Considerations. Cambridge University Press, New York (2001)
17. Miyake, N.: Constructive interaction and the interactive process of understanding. Cognitive Science 10, 151–177 (1986)
18. Hayashi, Y., Miwa, K., Morita, J.: A laboratory study on distributed problem solving by taking different perspectives. In: Proc. 28th Annual Conference of the Cognitive Science Society, pp. 333–338. Lawrence Erlbaum Associates (2006)
19. Hayashi, Y.: On pedagogical effects of learner support agents in collaborative interaction. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 22–32. Springer, Heidelberg (2012)
20. Hayashi, Y.: Pedagogical conversational agents for supporting collaborative learning: Effects of communication channels. In: Proc. CHI EA 2013, pp. 655–660. ACM Press (2013)
21. Levine, D., Resnick, L.B., Higgins, E.T.: Social foundations of cognition. Annual Review of Psychology 44, 585–612 (1993)
22. Alport, F.H.: The influence of the group upon association and thought. Journal of Experimental Psychology 3, 159–182 (1920)
23. Lee, E.J., Nass, C.: Experimental tests of normative group influence and representation effects in computer-mediated communication when interacting via computers differs from interacting with computers. Human Communication Research 28(3), 349–381 (2002)
24. Beck, B.U., Wintermantel, M., Borg, A.: Principles of regulating interaction in teams practicing face-to-face communication versus teams practicing computer-mediated communication. Small Group Research 36, 499–536 (2005)

What Works: Creating Adaptive and Intelligent Systems for Collaborative Learning Support

Nia M. Dowell[1], Whitney L. Cade[1], Yla Tausczik[2], James Pennebaker[3], and Arthur C. Graesser[1]

[1] Institute for Intelligent Systems, The University of Memphis, Memphis TN 38152 USA
{ndowell,wlcade,graesser}@memphis.edu
[2] Department of Social Computing, Carnegie Mellon University, Pittsburg PA 13289 USA
ylataus@cs.cmu.edu
[3] Department of Psychology, University of Texas at Austin, Austin TX 78705 USA
pennebaker@mail.utexas.edu

Abstract. An emerging trend in classrooms is the use of collaborative learning environments that promote lively exchanges between learners in order to facilitate learning. This paper explored the possibility of using discourse features to predict student and group performance during collaborative learning interactions. We investigated the linguistic patterns of group chats, within an online collaborative learning exercise, on five discourse dimensions using an automated linguistic facility, Coh-Metrix. The results indicated that students who engaged in deeper cohesive integration and generated more complicated syntactic structures performed significantly better. The overall group level results indicated collaborative groups who engaged in deeper cohesive and expository style interactions performed significantly better on posttests. Although students do not directly express knowledge construction and cognitive processes, our results indicate that these states can be monitored by analyzing language and discourse. Implications are discussed regarding computer supported collaborative learning and ITS's to facilitate productive communication in collaborative learning environments.

Keywords: collaborative interactions, learning, computational linguistics, Coh-Metrix.

1 Introduction

Current educational practices suggest an emerging trend toward collaborative problem solving or group learning [1,2]. This is reflected in the more recent upsurge of computer-mediated collaborative learning or groupware tools, such as email, chat, threaded discussion, massive open online courses (MOOCs), and trialog-based intelligent tutoring systems (ITSs). The growing adoption of collaborative learning environments is supported by research that shows that, in general, collaboration can increase group performance and individual learning outcomes (see [3] for a review). The interest of educational researchers in this topic has motivated a substantial area of

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 124–133, 2014.

research aimed at identifying and improving collaborative knowledge building processes using both ITSs and computer-supported collaborative learning (CSCL) systems [4]. Previous research in the area of collaborative learning has shown that information in the interaction itself can be useful in predicting the cognitive benefits that students take away [5,6]. For instance, cognitive elaboration, quality argumentation, common ground, task difficulty, and cognitive load have been shown to influence knowledge acquisition of the individual learner and performance of the overall group [7,8,9,10]. One factor that sets collaborative learning apart from individual learning is the use of collaborative language [11,12,13]. Being the root of all computer-mediated collaboration, language, discourse, and communication are critical for organizing a team, establishing a common ground and vision, assigning tasks, tracking progress, building consensus, managing conflict, and a host of other activities [1].

However, previous research in this area has predominantly focused on asynchronous communication, such as email or discussion boards, that require no real-time interaction between the users. In contrast, synchronous communication, such as text-based IM tools and videoconferencing, involves interactions that are dynamic and constantly updated [14]. Additionally, scholars typically rely on human coding, and have only recently applied automatic or semi-automatic natural language evaluation methods [2], [5], [15,16]. Consequentially, we know little about the actual process of knowledge construction in synchronous collaborative learning interactions.

There are several advantages to utilizing textual features as an independent channel for assessing collaborative communication processes. First, in the past, it has been an arduous task to assess communication during collaborative learning due to the complex nature of transcribing spoken conversations. However, advances in technology have increased the use of computer-mediated collaborative learning (CMCL), which allows researchers to track and analyze the language and discourse characteristics in group learning environments. Second, linguistic features derived from CMCL are contextually constrained in a fashion that provides cues regarding the social dynamics and an in-depth understanding of different qualities of interaction [2], [5], [17,18]. Third, recent advances in computational linguistics have convincingly demonstrated that language and discourse features can predict complex phenomenon such as personality, deception, emotions, successful group interaction, and even physical and mental health outcomes [19,20,21,22,23,24]. Thus, it is plausible to expect a textual analysis of symmetrical collaborative learning interactions to provide valuable insights into collaborative learning processes and performance.

A number of psychological models of discourse comprehension and learning, such as the construction-integration, constructionist, and indexical-embodiment models, lend themselves nicely to the exploration of how knowledge is constructed in collaborative learning interactions. These psychological frameworks of comprehension have identified the representations, structures, strategies, and processes at multiple levels of discourse [7], [25,26]. Computational linguistic tools that analyze discourse patterns at these multiple levels, such as Coh-Metrix (described later), can be applied in collaborative learning interactions to gain a deeper understanding of the discourse patterns useful for individual and group performance [7], [27,28]. This endeavor also holds the potential for enabling substantially improved collaborative learning environments both by providing real-time detection of students and group performance

and by using this information to develop the student model and trigger collaborative learning support as needed.

In the current study, we employ computational linguistic techniques to systematically explore chat communication during collaborative learning interactions in a large undergraduate psychology course. Specifically, we identify the discourse levels and linguistic properties of collaborative learning interactions that are predictive of learning. Further, we examine how these relations may differ for individual students and overall group level discourse. A more general overarching goal of this paper is to illustrate some of the advantages of automated linguistics tools to identify pedagogically valuable discourse features that can be applied in collaborative learning ITS and CSCL environments.

1.1 Brief Overview of Coh-Metrix

Coh-Metrix is a computer program that provides over 100 measures of various types of cohesion, including co-reference, referential, causal, spatial, temporal, and structural cohesion [27,28,29]. Coh-Metrix also has measures of linguistic complexity, characteristics of words, and readability scores. Currently, Coh-Metrix is being used to analyze texts in K-12 for the Common Core standards and states throughout the U.S. More than 50 published studies have demonstrated that Coh-Metrix indices can be used to detect subtle differences in text and discourse [28], [30].

There is a need to reduce the large number of measures provided by Coh-Metrix into a more manageable number of measures. This was achieved in a study that examined 53 Coh-Metrix measures for 37,520 texts in the TASA (Touchstone Applied Science Association) corpus, which represents what typical high school students have read throughout their lifetime [29]. A principal components analysis was conducted on the corpus, yielding eight components that explained an impressive 67.3% of the variability among texts; the top five components explained over 50% of the variance. Importantly, the components aligned with the language-discourse levels previously proposed in multilevel theoretical frameworks of cognition and comprehension [7], [25,26]. These theoretical frameworks identify the representations, structures, strategies, and processes at different levels of language and discourse, and thus are ideal for investigating trends in learning-oriented conversations. Below are the five major dimensions, or latent components:

- **Narrativity.** The extent to which the text is in the narrative genre, which conveys a story, a procedure, or a sequence of episodes of actions and events with animate beings. Informational texts on unfamiliar topics are at the opposite end of the continuum.
- **Deep Cohesion.** The extent to which the ideas in the text are cohesively connected at a deeper conceptual level that signifies causality or intentionality.
- **Referential Cohesion**. The extent to which explicit words and ideas in the text are connected with each other as the text unfolds.

- **Syntactic Simplicity**. Sentences with few words and simple, familiar syntactic structures. At the opposite pole are structurally embedded sentences that require the reader to hold many words and ideas in working memory.
- **Word Concreteness**. The extent to which content words that are concrete, meaningful, and evoke mental images as opposed to abstract words.

2 Methods

2.1 Participants, Materials, and Procedure

The participants were 851 undergraduates (62.4% female) in two introductory-level psychology courses at a large Midwestern university. Caucasians accounted for 49.6% of participants while Hispanic/Latino accounted for 22.4%, Asian American for 16.1%, African American 4.2% and less than 1% identified as either Native American or Pacific Islander. Twelve participants were discarded as outliers or due to computer failure, resulting in $N = 839$.

Students logged into an education platform managed within the University at specified times to complete the group interaction task. The education platform was an online course center where students filled out surveys, took quizzes, completed writing assignments, and participated in group chat. Prior to logging into the system, students were instructed that, in order to complete the assignment, they would need to read supplementary material on a few psychological theories (e.g. 10 pages of the text-book).

Once students logged into the educational platform, they were directed to the first quiz. The quiz was 10 multiple-choice questions and tested students' knowledge of the reading material. After completing the quiz, they were randomly matched with other students currently waiting to engage in the chatroom portion of the task. When there were at least 2 students and no more than 5 students ($M = 4.59$), individuals were directed to an instant messaging platform that was built into the educational platform. The group chat began as soon as someone typed the first message and lasted for 20 minutes. The chat window closed automatically after 20 minutes, at which time students took a second 10 multiple-choice question quiz. Each student contributed 154 words on average ($SD = 104.94$) in 19.49 sentences ($SD = 12.46$). As a group, discussions were about 714.8 words long ($SD = 235.68$) and 90.62 sentences long ($SD = 33.47$).

2.2 Performance

On average, students scored better on the posttest after the group discussion than on the pretest. Pretest and posttest scores, for both the individual and group, were converted to proportions based the number of correct answers. Group performance was then operationalized as the average group members' score on the pretest and posttest.

2.3 Data Treatment and Computational Evaluation

The educational platform logged all of the students' contributions. Prior to analysis, the logs were cleaned and parsed to facilitate two levels of evaluation. First, for the individual-level analyses, texts files were created that included all contributions from a single student, resulting in 839 text files. Second, we combined all group members' contributions into a text file for group-level analyses. All files were then analyzed using Coh-Metrix. Following the Coh-Metrix analysis, the scores were normalized by removing any outliers. Specifically, the normalization procedure involved Winsorising the data based on each variable's upper and lower percentile.

3 Results and Discussion

A mixed-effects modeling approach was adopted for all analyses due to the nested structure of the data (e.g., learners embedded within groups). Mixed-effects modeling is the recommended analysis method for this type of data [31]. Mixed-effects models include a combination of fixed and random effects and can be used to assess the influence of the fixed effects on dependent variables after accounting for any extraneous random effects. The lme4 package in R [32] was used to perform the requisite computation.

The primary analyses focused on identifying discourse features (namely, the five dimension used to generally describe texts in Coh-Metrix: Narrativity, Deep Cohesion, Referential Cohesion, Syntax Simplicity, and Word Concreteness) of the chat data that are predictive of learning. We also tested whether prior knowledge moderated the effect of discourse on learning performance. Separate models were constructed to analyze discourse at the individual learner and group levels in order to isolate their independent contributions on learning performance. Therefore, there were two sets of dependent measures in the present analyses: (1) individual learners' performance on the multiple-choice posttest and (2) overall groups' performance on the multiple-choice posttest. The independent variables in all models were the 5 discourse features of interest, as well as proportional pretest performance scores, which were included to control for the effect of prior knowledge. The random effects for the individual learner models were participant (839 levels), while the group model used participant (839 levels) within group (183 levels) as the random effect.

Table 1 shows the discourse features that were predictive of learning performance for both the individual and group level models. As can be seen from this table, learners' deep cohesion and syntax are predictive of individual learning performance. Specifically, we see that learners who engaged in deeper cohesive integration and generated more complicated syntactic structures were significantly more likely to score higher on the posttest than learners who used simpler syntax and less deep cohesion. Discourse cohesion, defined as the extent to which the ideas in the text are cohesively connected at a deeper conceptual level that signifies causality or intentionality, is a central component in a number of processes that facilitate individual learning and comprehension [7]. With regard to the findings for deep cohesion, this suggests that students who are learning are engaging in deeper integration of topics with

their background knowledge, generating more inferences to address any conceptual and structural gaps, and consequentially increasing the probability of knowledge retention. The finding for syntactic structure might provide evidence for the cognitive explanation hypothesis [17]. In general, this suggests that students who are producing denser sentence compositions are high verbal and/or are engaging in increased effort, inferences, and elaboration.

The analysis of collaborative group interaction discourse revealed that narrativity and deep cohesion were predictive of learning performance. In particular, the group-level results indicated that collaborative groups who engaged in more expository, or informational, style interactions significantly outperformed those with more narrative discourse. Initially, these findings seem counterintuitive based on previous research which found that narrative text is substantially easier to read, comprehend, and recall than informational text [7], even when the familiarity of the topics and vocabulary are controlled. However, students were instructed to talk about what they read in their textbook, which could suggest that groups that learned more were mirroring their textbook's more expository nature. Additionally, [29] noted that informational texts tend to have higher cohesion, as compared with narratives, and thus cohesion plays an important role in in compensating for the greater difficulty of expository style discourse. Deep cohesion was also predictive of learning performance in the group-level interaction analysis.

In addition to the previously mentioned benefits of deep cohesion for learning, cohesion also aids processes important for collaboration, including establishing and maintaining common ground [33], negotiating references [7], and coordinating group members' mental models [34]. High cohesion dialogue may indicate more thorough collaboration and learning in building a shared mental model. This is similar to the way high cohesion text can aid learners in building a solid mental model (relative to low cohesion text). In the context of group interactions, our findings support research showing that collaborative learners may create and preserve shared conceptions of a topic, and this social co-construction facilitates optimal collaboration for knowledge building [35]. We also tested whether prior knowledge moderated the effect of discourse on learning by assessing whether the prior knowledge x discourse feature interaction term significantly predicted posttest scores. However, the interaction term was not significant ($p > .05$) for any of the models.

Table 1. Descriptive Statistics and Mixed-Effects Model Coefficients

Measure	**Learner Model**				**Group Model**			
	M	*SD*	*B*	*SE*	*M*	*SD*	*B*	*SE*
Narrativity	.15	.79	.01	.01	.53	.34	**-.04***	.02
Deep Cohesion	.87	1.68	**.01****	.003	1.29	.75	**.03****	.01
Referential Cohesion	-.52	1.52	-.003	.005	-1.64	.42	.01	.02
Syntax Simplicity	.69	.81	**-.01***	.01	1.30	.37	-.001	.02
Word Concreteness	-2.07	1.07	-.01	.001	-2.67	.41	-.03	.01

Note: * $p < .05$; ** $p < .001$. Standard error (*SE*).

4 General Discussion

This paper explored the possibility of using discourse features to predict student and group performance during collaborative learning interactions. Although students do not directly express knowledge construction and cognitive processes, our results indicate that these states can be monitored by analyzing language and discourse. This suggests that it takes a more systematic and deeper analysis of dialogues to uncover diagnostic cues of the knowledge construction. Overall, the findings suggest that automated analyses of linguistic characteristics can provide valid representations of individual and group processes that are beneficial for knowledge construction during collaborative learning. In particular, students and collaborative groups can achieve new levels of understanding during collaborative learning interactions where more complex cognitive activities occur, such as analytical thinking, elaboration and integration of ideas and reasoning.

It is also interesting to note that it takes an analysis of both the student and collaborative group interaction to obtain a comprehensive understanding of the linguistic properties that influence knowledge acquisition during collaborative group interactions. These findings stimulate an interesting discussion because, until recently, most research on groups has concentrated on the individual people in the group as the cognitive agents [36]. This traditional granularity uses the individual as the unit of analysis both to understand behavioral characteristics of individuals working within groups and to measure performance or knowledge-building outcomes of the individuals in group contexts. However, the present findings support the claims of many in the CSCL community to also consider group levels of granularity in discourse tracking.

The present research has important implications for CSCL and collaborative learning-focused ITSs. In order to tailor interaction feedback to student needs, a system has to be able to automatically evaluate student interactions and to provide adaptive support. The support should be sensitive to these evaluations and also follow models of ideal collaboration. While the field has started to recognize the benefits of automated language evaluation, thus far, this technology has only been used effectively in limited ways (e.g. classifying the topic of conversation or speech acts) [37]. Some research has attempted to address the issue of evaluating dialogue by relying on more shallow measures like participation to trigger feedback. Unfortunately, these approaches make it difficult to give students feedback on *how* to contribute, which may ultimately be more valuable. Computational linguistics facilities, like Coh-Metrix and the Linguistic Inquiry and Word Count (LIWC) tool, could be used to alleviate some of the burdens of capturing these important processes. Additionally, systems that are based on underlying cognitive frameworks of knowledge construction have the advantage of being applicable in diverse contexts.

The present findings suggest that these systems have the capability of identifying linguistic features beneficial for knowledge construction on multiple levels, including individual learners and overall collaborative group interaction. Information gleaned from such analyses could be useful for those in pursuing CSCL and collaborative learning-focused ITSs. For instance, a system could provide accurate real time support for learners using an interface that delivered suggestions via a simple pop up window or a more sophisticated intelligent agent. However, the value of such enhancements awaits future work and empirical testing.

Acknowledgments. This research was supported by the National Science Foundation (BCS 0904909, DRK-12-0918409), the Institute of Education Sciences (R305G020018, R305A080589), The Gates Foundation, U.S. Department of Homeland Security (Z934002/UTAA08-063), and the Army Research Institute (W5J9CQ12C0043). Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. Graesser, A.C., Foltz, P., Rosen, Y., Forsyth, C., Germany, M.: Challenges of Assessing Collaborative Problem-Solving. In: Csapo, B., Funke, J., Schleicher, A. (eds.) The Nature of Problem Solving. OECD Series (in press)
2. De Wever, B., Schellens, T., Valcke, M., Van Keer, H.: Content Analysis Schemes to Analyze Transcripts of Online Asynchronous Discussion Groups: A Review. Comput Educ. 46, 6–28 (2006)
3. Lou, Y., Abrami, P.C., d' Apollonia, S.: Small Group and Individual Learning with Technology: A Meta-Analysis. Rev. Educ. Res. 71, 449–521 (2001)
4. Gress, C.L.Z., Fior, M., Hadwin, A.F., Winne, P.H.: Measurement and Assessment in Computer-Supported Collaborative Learning. Comput. Hum. Behav. 26, 806–814 (2010)
5. Rosé, C., Wang, Y.-C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., Fischer, F.: Analyzing Collaborative Learning Processes Automatically: Exploiting the Advances of Computational Linguistics in Computer-Supported Collaborative Learning. Int. J. Comput.-Support. Collab. Learn. 3, 237–271 (2008)
6. King, A.: Scripting Collaborative Learning Processes: A Cognitive Perspective. In: Fischer, F., Kollar, I., Mandl, H., Haake, J.M. (eds.) Scripting Computer-Supported Collaborative Learning, pp. 13–37. Springer, Heidelberg (2007)
7. Graesser, A.C., McNamara, D.S.: Computational Analyses of Multilevel Discourse Comprehension. Top. Cogn. Sci. 3, 371–398 (2011)
8. Kirschner, P.A., Ayres, P., Chandler, P.: Contemporary Cognitive Load Theory Research: The Good, the Bad and the Ugly. Comput. Hum. Behav. 27, 99–105 (2011)
9. Noroozi, O., Weinberger, A., Biemans, H.J.A., Mulder, M., Chizari, M.: Argumentation-Based Computer Supported Collaborative Learning (ABCSCL): A synthesis of 15 years of research. Educ. Res. Rev. 7, 79–106 (2012)
10. Baker, M., Hansen, T., Joiner, R., Traum, D.: The Role Of Grounding In Collaborative Learning Tasks. In: Dillenbourg, P. (ed.) Collaborative Learning: Cognitive and Computational Approaches, pp. 31–36. Emerald Group Publishing Limited, Bingley (1999)
11. Fiore, S., Schooler, J.: Process Mapping and Shared Cognition: Teamwork and the Development of Shared Problem Models. In: Salas, E., Fiore, S. (eds.) Team Cognition: Understanding the Factors that Drive Process and Performance, pp. 133–152. American Psychological Association, Washington, D.C (2004)
12. Fiore, S.M., Rosen, M.A., Smith-Jentsch, K.A., Salas, E., Letsky, M., Warner, N.: Toward an Understanding of Macrocognition in Teams: Predicting Processes in Complex Collaborative Contexts. Hum. Factors. 52, 203–224 (2010)
13. Dillenbourg, P., Traum, D.: Sharing Solutions: Persistence and Grounding in Multimodal Collaborative Problem Solving. J. Learn. Sci. 15, 121–151 (2006)

14. Hou, H.-T., Wu, S.-Y.: Analyzing the Social Knowledge Construction Behavioral Patterns of an Online Synchronous Collaborative Discussion Instructional Activity Using an Instant Messaging Tool: A Case Study. Comput. Educ. 57, 1459–1468 (2011)
15. Yoo, J., Kim, J.: Can Online Discussion Participation Predict Group Project Performance? Investigating the Roles of Linguistic Features and Participation Patterns. Int. J. Artif. Intell. Educ., 1–25 (2014)
16. Murray, T., Woolf, B.P., Xu, X., Shipe, S., Howard, S., Wing, L.: Supporting Social Deliberative Skills in Online Classroom Dialogues: Preliminary Results Using Automated Text Analysis. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 666–668. Springer, Heidelberg (2012)
17. Webb, N.M.: Peer Interaction and Learning in Small Groups. Int. J. Educ. Res. 13, 21–39 (1989)
18. Van der Pol, J., Admiraal, W., Simons, P.R.J.: The Affordance of Anchored Discussion for the Collaborative Processing of Academic Texts. Int. J. Comput.-Support. Collab. Learn. 1, 339–357 (2006)
19. Scholand, A.J., Tausczik, Y.R., Pennebaker, J.W.: Assessing Group Interaction with Social Language Network Analysis. In: Chai, S.-K., Salerno, J.J., Mabry, P.L. (eds.) SBP 2010. LNCS, vol. 6007, pp. 248–255. Springer, Heidelberg (2010)
20. D'Mello, S., Dowell, N., Graesser, A.C.: Cohesion Relationships in Tutorial Dialogue as Predictors of Affective States. In: Dimitrova, V., Mizoguchi, R., du Boulay, B., Graesser, A. (eds.) AIED 2009, pp. 9–16. IOS Press, Amsterstam (2009)
21. Mairesse, F., Walker, M.A.: Towards Personality-Based User Adaptation: Psychologically Informed Stylistic Language Generation. User Model. User-Adapt. Interact. 20, 227–278 (2010)
22. Newman, M.L., Pennebaker, J.W., Berry, D.S., Richards, J.M.: Lying Words: Predicting Deception from Linguistic Styles. Pers. Soc. Psychol. Bull. 29, 665–675 (2003)
23. Tausczik, Y.R., Pennebaker, J.W.: The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. J. Lang. Soc. Psychol. 29, 24–54 (2010)
24. Hancock, J.T., Woodworth, M.T., Porter, S.: Hungry Like the Wolf: A Word-Pattern Analysis of the Language of Psychopaths. Leg. Criminol. Psychol. 18, 102–114 (2013)
25. Kintsch, W.: Comprehension: A Paradigm for Cognition. Cambridge University Press, Cambridge (1998)
26. Snow, C.E.: Reading for Understanding: Toward a Research and Development Program in Reading Comprehension. Rand Corporation, Santa Monica (2002)
27. Graesser, A.C., McNamara, D.S., Louwerse, M.M., Cai, Z.: Coh-Metrix: Analysis of Text on Cohesion and Language. Behav. Res. Methods Instrum. Comput. J. Psychon. Soc. Inc. 36, 193–202 (2004)
28. McNamara, D.S., Graesser, A.C., McCarthy, P.M., Cai, Z.: Automated Evaluation of Text and Discourse with Coh-Metrix. Cambridge University Press, Cambridge (2014)
29. Graesser, A.C., McNamara, D.S., Kulikowich, J.M.: Coh-Metrix: Providing Multilevel Analyses of Text Characteristics. Educ. Res. 40, 223–234 (2011)
30. McNamara, D.S., Crossley, S.A., McCarthy, P.M.: Linguistic Features of Writing Quality. Writ. Commun. 27, 57–86 (2010)
31. Pinheiro, J.C., Bates, D.M.: Mixed-effects models in S and S-Plus. Springer, Heidelberg (2000)
32. Bates, D., Maechler, M., Bolker, B., Walker, S.: lme4: Linear mixed-effects models using Eigen and S4 (2013)

33. Clark, H.H., Brennan, S.E.: Grounding in Communication. In: Resnick, L.B., Levine, J.M., Teasley, S.D. (eds.) Perspectives on Socially Shared Cognition, pp. 127–149. American Psychological Association, Washington, DC (1991)
34. Pickering, M.J., Garrod, S.: Toward a Mechanistic Psychology of Dialogue. Behav. Brain Sci. 27, 169–190; discussion 190–226 (2004)
35. Fischer, F., Bruhn, J., Gräsel, C., Mandl, H.: Fostering Collaborative Knowledge Construction with Visualization Tools. Learn. Instr. 12, 213–232 (2002)
36. Stahl, G.: From Individual Representations to Group Cognition. In: Stahl, G. (ed.) Studying Virtual Math Teams, pp. 57–73. Springer, US (2009)
37. Diziol, D., Walker, E., Rummel, N., Koedinger, K.R.: Using Intelligent Tutor Technology to Implement Adaptive Support for Student Collaboration. Educ. Psychol. Rev. 22, 89–102 (2010)

Using an Intelligent Tutoring System to Support Collaborative as well as Individual Learning

Jennifer K. Olsen[1], Daniel M. Belenky[1], Vincent Aleven[1], and Nikol Rummel[1,2]

[1] Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
{Jkolsen,aleven}@cs.cmu.edu, dbelenky@andrew.cmu.edu
[2] Institute of Educational Research, Ruhr-Universität Bochum, Germany
nikol.rummel@rub.de

Abstract. Collaborative learning has been shown to be beneficial for older students, but there has not been much research to show if these results transfer to elementary school students. In addition, collaborative and individual modes of instruction may be better for acquiring different types of knowledge. Collaborative Intelligent Tutoring Systems (ITS) provide a platform that may be able to provide both the cognitive and collaborative support that students need. This paper presents a study comparing collaborative and individual methods while receiving instruction on either procedural or conceptual knowledge. The collaborative groups had the same learning gains as the individual groups in both the procedural and conceptual learning conditions but were able to do so with fewer problems. This work indicates that by embedding collaboration scripts in ITSs, collaborative learning can be an effective instructional method even with young children.

Keywords: Problem solving, collaborative learning, intelligent tutoring system.

1 Introduction

While collaborative learning has been shown to be beneficial for both face-to-face and Computer Supported Collaborative Learning (CSCL) [9], [14], collaborative learning often puts challenges on students and teachers that make it hard to implement in the classroom. The challenges teachers face include preparing materials, teaching the students collaboration skills, and learning how to manage small groups [3]. For students, fruitful collaboration does not happen spontaneously, and collaboration scripts are used to support students in their learning [6]. It is important for a script to match the learning goals of the activity and to provide enough support for the students without over-scripting. Collaboration can be supported through different features such as roles, cognitive group awareness, and the distribution of information. The challenges faced by both the students and teachers can make the use of collaboration daunting. Some prior research has indicated that Intelligent Tutoring Systems (ITSs) can be a practical way of addressing the challenges of using collaboration in the classroom. Most CSCL environments are missing the cognitive support that can be beneficial to student learning. An ITS can provide the cognitive support (i.e. step-by-step

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 134–143, 2014.

guidance and hint features) that a student needs for collaboration to be successful [18], but does not provide support for effective collaboration. The current research investigates if embedding a collaboration script into an ITS so it has both the collaborative and cognitive support can help a student to learn successfully.

Even though collaborative learning has been shown to be successful in some instances, few studies have investigated whether CSCL can have a positive impact on learning with young children. The implementation and support of collaboration in the classroom is particularly difficult for students in elementary school and may explain why there is less research with this age group. An important question then is if collaborative learning can be an effective instructional method to use with elementary school students and if it would lead to similar learning gains as students working individually. Some studies have shown successful use of collaboration with elementary school students as well, but have either compared the use of a CSCL setting to face-to-face collaborative learning (i.e., not supported by computers) without comparing it to individual learning or have focused on interventions that mix individual and collaborative learning tasks without looking at each separately [1], [8], [16]. Although this research has shown positive impacts of young children working in small groups and with computers, it is still unknown how the use of a CSCL environment impacts the learning outcomes of young children compared to learning individually. This paper aims to address this question through an ITS designed specifically to support collaborative learning of children in elementary school. ITSs have been shown to have positive impacts on students in this age group when working individually to learn fractions [12]. We now extend this research by testing whether a tutor that supports collaboration can be effective for learning fractions by elementary school students.

Although most prior work on ITSs has focused on individual learning, there has been some work on combining ITSs with collaborative learning that has shown promise for supporting learning with high school students [17]. Walker et al. found that students working with an ITS redesigned to support collaboration (specifically, peer tutoring) had learning gains at least equivalent to those working individually.

In creating a collaborative tutor, it may be important to consider the possibility that individual and collaborative learning activities may be better for acquiring different types of knowledge, such as conceptual and procedural knowledge [10]. Conceptual knowledge is the implicit and explicit understanding of the principles in a domain and how they are interrelated [13]. Procedural knowledge is the ability to be able to perform the steps and actions in sequence to solve a problem [13]. Mullins, Rummel, and Spada found that with 9th graders doing algebra, students who worked collaboratively on conceptual tasks outperformed those who worked individually and students who worked individually on procedural tasks outperformed those who worked collaboratively [10]. Again, this study was implemented with older students and the question still remains if the same difference will be seen with elementary school students.

Why would it be better to acquire different types of knowledge through different instructional methods? Following the Knowledge-Learning Instruction (KLI) framework, instruction should be designed for both the domain and for the type of knowledge component to be learned [5]. Simpler instructional methods tend to be associated with simpler knowledge components, more complex methods with more complex

knowledge components. Thus, collaboration, a more complex instructional method, would be better for more complex knowledge components, where elaboration and a deeper understanding is needed, such as those in conceptual knowledge. More specifically, collaborative learning may be successful because the students give and receive explanations and construct knowledge through their discussions [4]. On the other hand, individual learning would be more geared towards procedural learning where practice and repetition are more important for developing fluency.

In our study, we address the feasibility of using a collaborative ITS with elementary school students learning fractions. We hypothesize that students working collaboratively will show learning gains on both procedural and conceptual fractions tasks. Also, we hypothesize that on conceptual tasks, students working collaboratively will have stronger learning gains than students working individually. By contrast, for students doing procedural tasks, we hypothesize that those working individually will have stronger learning gains than those working collaboratively. These hypotheses are consistent with both the KLI framework and the Mullins et al. findings.

2 Methods

2.1 Tutor Design

Informed by our prior work on the Fractions Tutor [12], we developed a new ITS for a challenging topic in fractions, learning equivalent fractions. Specifically, we built two parallel versions of this tutor for use in our study, one with embedded collaboration scripts and one for individual learning. Both versions had procedural and conceptual problem sets. Both were built with CTAT, which we extended to support collaborative tutors [11]. The collaborative ITS combines the cognitive support normally provided by an ITS (step-level guidance for problem solving) with embedded collaborative scripts for each tutor problem. The collaboration is supported through the use of a shared problem view, roles, cognitive group awareness, and unique information. First, the collaborative tutors support synchronous, networked collaboration, in which collaborating students sit at their own computer and have a shared (though differentiated) view of the problem state. They can discuss the activity through audio chat.

Second, the embedded scripts define roles to distribute the activities between the students. The roles provide guidance to the students about what they should be doing to interact with their partner and help to scaffold this interaction. Students were assigned to either a helper role or a problem solver role for each task in a problem. The students were informed of their role assignment through the use of icons displayed on the interface (see Figure 1). An “ask” icon next to a problem step signaled to the student that they were in the helper role and responsible for asking questions and making sure both they and their partner understood the answer. A “do” icon next to a component meant the student was in the problem solver role and responsible for carrying out the step to move the problem solution forward (Figure 1).

A third collaborative support feature we used in the collaborative problem sets is cognitive group awareness. Cognitive group awareness means that group members

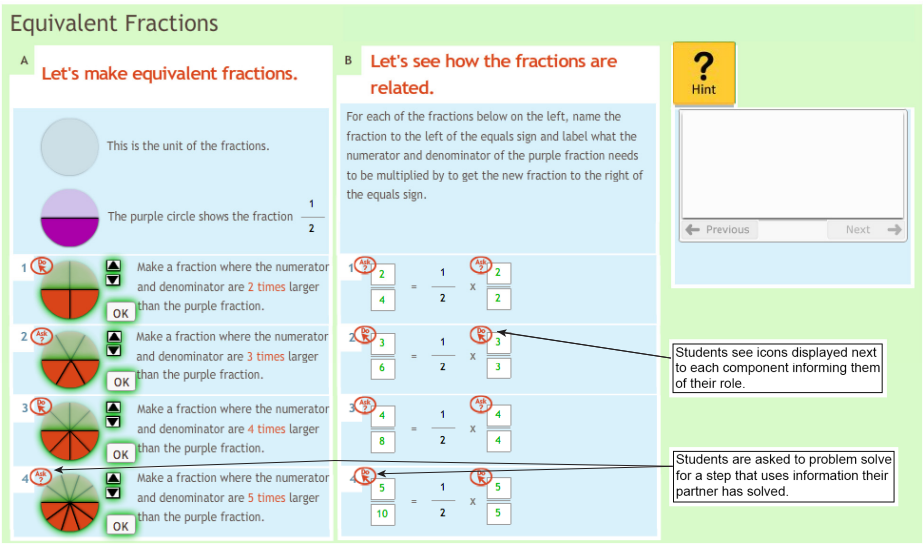


Fig. 1. A collaborative procedural problem: multiplying to make equivalent fractions

have information about other group members' knowledge, information, or opinions and has been shown to be effective for the collaboration process [7]. We implemented cognitive group awareness by a design pattern in which the collaborative tutor poses a question to both students and asks each student to answer independently first without being tutored (bottom of Figure 2). After both students answer the question independently, the tutor shows them each other's answers and gives them the opportunity to answer the question as a group, which is tutored. This activity allowed each student an opportunity to express an opinion and gave each dyad an opportunity to discuss and explain their answer choices, especially important when they disagreed.

The last collaborative support feature is the use of unique information to create a sense of individual accountability, a popular feature in scripts such as the jigsaw [2]. Individual accountability means that each group member takes responsibility for the group reaching its goal [14]. By providing each student with information that their partner does not have and that is needed to complete the problem, both students have a stake in completing the problem. In our problem sets, unique information was implemented by providing one member of the dyad with some information the other student did not see. The student would know they had unique information because there would be a share icon next to the information. The other student would need this information to complete a step of the problem and would see a listen icon to know there was some information they needed to get from their partner.

To test our experimental hypotheses, two problem sets were created for both the collaborative and the individual ITSs. One set focused on procedural knowledge of equivalent fractions while the other set focused on conceptual knowledge of equivalent fractions. The procedural problem set has four problem types, with four problems

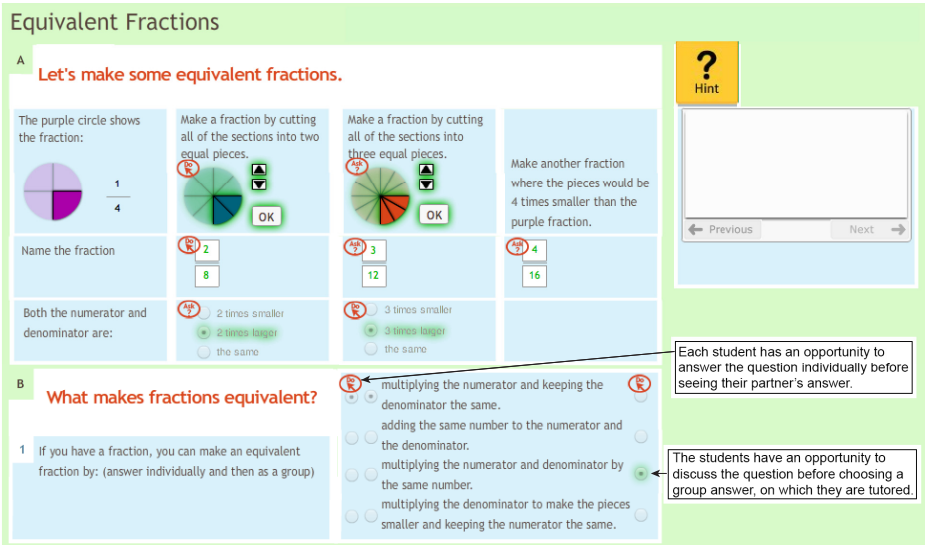


Fig. 2. Example of a collaborative conceptual problem: creating equivalent fractions to find the pattern in the fractions

each, which focus on finding equivalent fractions or determining whether fractions are equivalent, either by finding the common factors and reducing the fraction or by multiplying the numerator and denominator by the same number (see Figure 1). Each of the problem types focused on the steps needed to complete that procedure, without addressing conceptual questions about why the procedure works. The conceptual problem set also has four different problem types and four problems of each type. Two of the problem types provide the students with two stories about whether given fractions are equivalent that they need to compare and contrast (one story is correct and one story focuses on a misconception) or by providing the students with one story that focuses on a misconception that students need to address. The other two problem types focus on the definition of equivalent fractions by either having the students construct equivalent fractions to find a pattern in the fractions or by having students manipulate the denominators and numerators of the fractions independently to see how they relate (see Figure 2). For both problem types, students then induce a definition of what it means for fractions to be equivalent.

2.2 Experimental Design and Procedure

To test the hypotheses stated above, we conducted a study with 84 4th and 5th grade students from two US elementary schools in the same school district. The students came from a total of six classrooms. The experiment was a "pull-out" design, where the student left their normal instruction during the school day to participate in the study. (We did so we could collect eye tracking data, which are not reported here.) All students worked with the fractions ITS designed for this study and described

above. Each teacher paired the students participating in the study based on students who would work well together and had similar math abilities. These pairs were then randomly assigned to one of four conditions: collaborative conceptual, collaborative procedural, individual conceptual, and individual procedural. Twice as many students were assigned to the collaborative conditions as to the individual conditions.

Before participating in the pull-out session, the students had two whole class sessions during which they worked individually with the Fractions Tutor during their normal class period (on fractions topics other than equivalence). This allowed the students to become acclimated with the tutor before the experiment began. During the experiment, the students participated in a 25-minute pretest the morning of their participation. Throughout the day, the pairs of students participated in the pull-out session. Each such session lasted for one hour where during this time, they received 45 minutes of instruction dependent on their condition. The next school day, the students participated in a 25-minute posttest in the morning. The study spanned a total of four weeks. After the end of the study, the students again had two whole class sessions where they again worked independently on the Fractions Tutor.

2.3 Pre and Posttests

We assessed students' knowledge at two different times using two equivalent test forms in counterbalanced fashion. The tests targeted both conceptual and procedural knowledge types. Each test had 11 questions, five procedural and six conceptual. Each question either received a 1 when all parts were correct or a 0 otherwise. The test items were isomorphic to the items used in the practice problems.

3 Results

Table 1. Total correct: means (standard deviation) for conceptual and procedural knowledge at pretest, posttest, Min. score is 0, and max. score is 5 for procedural and 6 for conceptual.

			pretest	posttest
Conceptual Condition	Individual Condition	Conceptual Problems	2.00 (1.63)	2.54 (1.56)
		Procedural Problems	0.46 (0.66)	0.85 (1.21)
	Collaborative Condition	Conceptual Problems	2.04 (1.32)	2.54 (1.20)
		Procedural Problems	0.50 (0.75)	0.82 (0.82)
Procedural Condition	Individual Condition	Conceptual Problems	1.50 (0.76)	1.64 (1.28)
		Procedural Problems	0.50 (0.86)	0.64 (1.08)
	Collaborative Condition	Conceptual Problems	2.08 (1.67)	2.58 (1.42)
		Procedural Problems	0.92 (1.16)	0.92 (1.16)

Because the procedural and conceptual tutor problems were fundamentally different, each of these conditions was treated separately and the collaborative and individual conditions were not compared across problem types. Three students were excluded from the analysis because experimenter error, leaving 81 students. We analyzed the

data by individual so we could evaluate each student's learning gain. To test our hypothesis that, on tutor activities targeting conceptual knowledge, students working collaboratively have higher learning gains than students working individually, we conducted two repeated-measures ANOVAs, one for procedural test items and one for conceptual items, with condition (collaborative or individual) as a between-subjects factor and test-time (pretest and posttest) as repeated measure. For the conceptual test items, there is a significant pre/post difference, F (1, 39) = 4.23, p = .046, no main effect of condition, F (1,39) = .002, p = .966, and no interaction, F (1, 39) = .006, p = .940. For the procedural test items, there is a marginal pre/post difference, F (1, 39) = 4.00, p = .053, no main effect of condition, F (1, 39) = .001, p = .976, and no interaction, F (1, 39) = .032, p = .859. There were significant learning gains for both the collaborative and individual condition and no difference in gains between conditions.

To evaluate our hypothesis that students working individually on tutor problems targeting procedural knowledge have higher learning gains than students working collaboratively, we conducted two repeated-measures ANOVA (for procedural test items and conceptual test items, respectively) with condition (collaborative or individual) as a between-subjects factor and test-time (pretest and posttest) as repeated measure. For the conceptual test items, there is no effect of pre/post, F (1, 38) = 2.10, p = .16, a marginal effect of condition, F (1, 38) = 3.44, p = .071 with the collaborative group higher, and no interaction F (1, 38) = .65, p = .426. For the procedural test items, there is no effect of pre/post, F (1, 38) = .22, p = .64, no main effect of condition, F (1, 38) = 1.12, p = .297, nor an interaction between condition and pre/post, F (1, 38) = .22, p = .64. There was no learning gain difference between the collaborative and individual conditions. The conditional difference reflects the fact that the students in the individual procedural group started lower at pretest and remained lower at posttest. We also analyzed learning curves derived from the tutor logs for evidence of learning during tutor use. Specifically, we looked at the slope coefficient in the AFM regression equation (see Figure 3), a standard way of analyzing tutor log data [15]. Averaged across knowledge components, the slope was 0.27 for the conceptual conditions and 0.15 for the procedural conditions. For the conceptual conditions, 81% of the learning curves has a slope of 0.05 or higher (a rule of thumb threshold value for a slope to represent effective learning) and for the procedural conditions, 60% of the learning curves had a slope above 0.05.

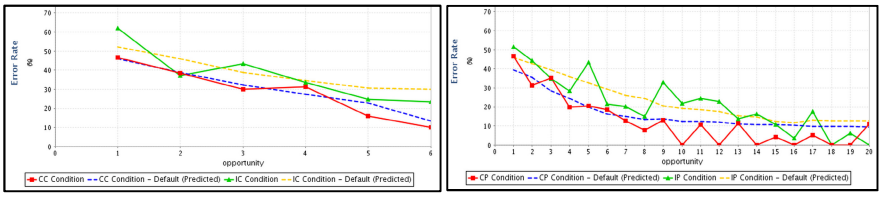


Fig. 1. Learning curves for conditions targeting conceptual (left) and procedural (right) knowledge. The learning curves are averaged across knowledge components encountered in the respective tutor problem sets. The red and blue lines represent the actual and AFM-predicted values for the collaborative conditions; the green and yellow lines for the individual conditions.

We conducted two t-tests (for each procedural/conceptual instructional condition) with collaborative/individual as the condition to see if there was a difference in the number of problems each student completed. For the procedural instructional condition, there is a significant difference, t (38) = 2.65, p = .012, with students working collaboratively doing fewer problems than students working individually by about 2.5 problems. For the conceptual instruction condition, there is a significant difference, t (39) =3.61, p = .001, again with students working collaboratively doing fewer problems than students working individually by about 3.5 problems.

4 Discussion and Conclusion

We hypothesized that elementary school students working collaboratively with a tutor designed to support collaboration would have learning gains from pretest to posttest. The hypothesis was confirmed; the students in the collaborative conceptual condition had learning gains comparable to those in the individual conceptual condition. In the procedural instructional condition, neither the collaborative nor individual conditions saw any learning gains. Thus, collaborative instruction might be as effective for elementary school students as individual instruction, although it appears to be more suitable for activities aimed at acquisition of conceptual knowledge. Collaborative learning activities may have the added benefit that they help students develop social skills and learn to work together.

While students in the collaborative condition saw fewer problems compared to their counterparts in the individual condition, they still had the same learning gains as the students in the individual conditions. This is consistent with other findings in CSCL [17]. This means that when authoring tutors, if collaborative tutors are used, fewer problems need to be developed to facilitate learning. However, we controlled for time and if we had controlled for number of problems, students in the individual condition may have learned as much as the students in the collaborative condition but in less time.

While we had hypothesized that the individual condition would yield greater learning gains than the collaborative condition for activities geared towards acquiring procedural knowledge and that the reverse would hold for activities geared towards acquiring conceptual knowledge, we did not find these differences. We may not have found these differences because the instructional period was relatively short. On average the students in the collaborative conceptual condition completed 7 problems. Because the problem types were interleaved and not all knowledge components were present in each problem type, the students did not always get to practice each knowledge component sufficiently. For the collaborative condition, out of the 16 knowledge components targeted in the conceptual problems, 9 of the knowledge components saw (on average, per student) fewer than 5 opportunities to practice a knowledge component. However, the students in the individual condition completed 12 problems on average and had at least 5 opportunities for all 16 knowledge components. By lengthening the practice time with the tutor, such as using the tutor for consecutive days in the classroom, the students would have more time with the tutor and would get more practice. This would help the students to get more practice with the individual knowledge components.

A second explanation for the fact that the hypothesized differences between the conditions were not confirmed may be that the collaborative learning condition was more novel and perhaps more demanding for students. Put differently, students may need more practice with the instructional method of collaborative learning. Especially given that the number of skill opportunities was low, one might expect to see better performance on the posttest. Other studies have also shown that the introduction of new learning strategies can initially lead to worse learning [19]. These initial performance losses may initially mask the success of a new learning strategy.

The fact that there were no learning gains in the procedural conditions may be due to the fact that the procedural problems may have been too difficult for the students. We also saw that overall for all conditions, the average number of problems solved correctly for the procedural problem types on either the pretest or the posttest was below one out of five (see Table 1). The learning curves for the individual knowledge components do show signs of learning during the instructional session, with an averaged slope across knowledge components of 0.15, well above the 0.05 threshold. Though the learning curves show that students start at an error rate above 50%, they also show clear signs of improvement. Because many of the procedural problems are multistep, the tests may need to be more fine-tuned to the specific knowledge components being learned instead of a cumulative approach of getting the entire problem correct. To be able to differentiate between the procedural and conceptual knowledge, more work will need to be done to develop and test tutors that can target this knowledge.

The study presented in this paper extends ITSs to include support for collaborative learning activities. We have showed that collaborative ITSs are a feasible instructional tool to use with elementary school students, with learning gains equivalent to those of students working independently with ITSs. The students in the collaborative condition also expressed enjoyment in working with a partner to solve problems. To the best of our knowledge, our study is the first showing significant learning gains with elementary school students working with collaborative ITSs. The use of collaborative ITS shows initial promise with elementary school students.

Acknowledgments. We thank the Cognitive Tutor Authoring Tools team, Amos Glenn, and Ryan Carlson for their help. This work was supported by Graduate Training Grant # R305B090023 and by Award # R305A120734 both from the US Department of Education (IES).

References

1. Chen, W., Looi, C.: Group Scribbles-Supported Collaborative Learning in a Primary Grade 5 Science Class. In: Suthers, D.D., Lund, K., Rose, C.P., Teplovs, C., Law, N. (eds.) Productive Multivocality in the Analysis of Group Interactions, pp. 257–263. Springer, New York (2013)
2. Dillenbourg, P., Jermann, P.: Designing integrative scripts. In: Scripting Computer-Supported Communication of Knowledge. Cognitive, computational, and educational perspectives, pp. 275–301. Springer, New York (2007)

3. Gillies, R.M., Boyle, M.: Teachers' reflections on cooperative learning: Issues of implementation. Teaching and Teacher Education 26(4), 933–940 (2010)
4. Hausmann, R.G., Chi, M.T., Roy, M.: Learning from collaborative problem solving: An analysis of three hypothesized mechanisms. In: 26nd Annual Conference of the Cognitive Science Society, pp. 547–552 (2004)
5. Koedinger, K.R., Corbett, A.T., Perfetti, C.: The knowledge-learning-instruction (KLI) framework: Toward bridging the science-practice chasm to enhance robust student learning. Cognitive Science (2010)
6. Kollar, I., Fischer, F., Hesse, F.W.: Collaboration scripts–a conceptual analysis. Educational Psychology Review 18(2), 159–185 (2006)
7. Janssen, J., Bodemer, D.: Coordinated computer-supported collaborative learning: Awareness and awareness tools. Educational Psychologist 48(1), 40–55 (2013)
8. Lazakidou, G., Retalis, S.: Using computer supported collaborative learning strategies for helping students acquire self-regulated problem-solving skills in mathematics. Computers & Education 54(1), 3–13 (2010)
9. Lou, Y., Abrami, P.C., d'Apollonia, S.: Small group and individual learning with technology: A meta-analysis. Review of Educational Research 71(3), 449–521 (2001)
10. Mullins, D., Rummel, N., Spada, H.: Are two heads always better than one? Differential effects of collaboration on students' computer-supported learning in mathematics. Int'l Journal of Computer-Supported Collaborative Learning 6(3), 421–443 (2011)
11. Olsen, J.K., Belenky, D.M., Aleven, V., Rummel, N., Sewall, J., Ringenberg, M.: Authoring collaborative intelligent tutoring systems. In: Kumar, R., Kim, J. (eds.) Proceedings 2nd Workshop on Intelligent Support for Learning in Groups at the 16th International Conference on Artificial Intelligent in Education, pp. 1–10 (2013)
12. Rau, M.A., Aleven, V., Rummel, N., Rohrbach, S.: Sense making alone doesn't do it: Fluency matters too! ITS Support for Robust Learning with Multiple Representations. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 174–184. Springer, Heidelberg (2012)
13. Rittle-Johnson, B., Siegler, R.S., Alibali, M.W.: Developing conceptual understanding and procedural skill in mathematics: An iterative process. Journal of Educational Psychology 93(2), 346 (2001)
14. Slavin, R.E.: Research on cooperative learning and achievement: What we know, what we need to know. Contemporary Educational Psychology 21(1), 43–69 (1996)
15. Stamper, J.C., Koedinger, K.R.: Human-machine student model discovery and improvement using DataShop. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 353–360. Springer, Heidelberg (2011)
16. Tsuei, M.: Development of a peer-assisted learning strategy in computer-supported collaborative learning environments for elementary school students. British Journal of Educational Technology 42(2), 214–232 (2011)
17. Walker, E., Rummel, N., Koedinger, K.: CTRL: A research framework for providing adaptive collaborative learning support. User Modeling and User-Adapted Interaction: The Journal of Personalization Research (UMUAI) 19(5), 387–431 (2009)
18. Walker, E., Rummel, N., McLaren, B.M., Koedinger, K.R.: The student becomes the master: Integrating peer tutoring with cognitive tutoring. In: Proceedings of the 8th International Conference on Computer Supported Collaborative Learning, pp. 751–753. International Society of the Learning Sciences (2007)
19. Westermann, K., Rummel, N.: Delaying Instruction – Evidence from a Study in a University Relearning Setting. Instructional Science 40(4), 673–689 (2012), doi:0.1007/s11251-012-9207-8

Bayesian Student Modeling Improved by Diagnostic Items

Yang Chen, Pierre-Henri Wuillemin, and Jean-Marc Labat

LIP6, University Pierre and Marie Curie
4 Place Jussieu, 75005 Paris, France
{yang.chen,pierre-henri.wuillemin,jean-marc.labat}@lip6.fr

Abstract. Bayesian network (BN) has been successfully applied in hierarchical student models. Some researchers used diagnostic strategies to improve the evidence level of student models. But test items are typically related to a dichotomous response model, namely students' answers are scored as right or wrong. As we know, wrong answers result from lacking one or more relevant concepts in students' knowledge states. This diagnostic information of wrong answers is ignored. To maximize the precision of student model, this paper presents an approach using diagnostic items, which are designed to provide the information about which concepts are probably lacked in students' knowledge states when they give wrong answers. A modified NIDA (Noisy Input, Deterministic AND) model is built to represent the relations between students' answers and their knowledge states. We use simulated students to evaluate our model and the results show that the efficiency and accuracy of student modeling are improved.

Keywords: Student model, Bayesian network, NIDA, Diagnosis.

1 Introduction

Student modeling is a tough task: uncertainty exists when we infer students' knowledge from their performances during problem solving [1]. Students might perform correctly by guessing even though they do not know the relevant concepts. On the contrary, students knowing the relevant concepts might incorrectly perform by slipping. BNs have been successfully applied for hierarchical student models [2, 3, 4, 5]. To maximize the precision, some diagnostic strategies were constructed to improve the evidence level of hierarchical student models. Millán and Pérez-de-la-Cruz [3] combined Item Response Theory (IRT) with Bayesian student modeling on evidence level, which reasonably applied the diagnostic information that the student lacking more relevant concepts has less probability of answering the question correctly. However, their model assessed students' answers as right or wrong, without considering plentiful information from wrong answers. A psychometric model MC-DINA introduced by de la Torre [6] applied the diagnostic information of students' answers by using multiple choice questions (MCQs). The author indicated the relations between wrong options of MCQs and students' knowledge states, and applied a data-driven approach to estimate the parameters. But due to too many parameters in his model,

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 144–149, 2014.

the estimation is complex. Inspired by these researches, we aim to model the relations between students' wrong answers and their knowledge states with a reduced number of parameters. By introducing the diagnostic information of wrong answers, the efficiency and accuracy of student modeling can be improved.

2 Diagnostic NIDA Model

The evidence level of student model deals with the relations between test items (e.g. questions) and knowledge items (e.g. concepts). The relationship between questions and concepts can be represented as Q-matrix in psychometrics. According to the presence or absence of the concepts, students' answers can be characterized. We suppose that the codes 1 and 0 respectively represent concepts correctly and not correctly used. Hence, if answering a question requires using three concepts, the correct answer can be coded as 111, and the wrong answer which is coded as 101 means that only the second concept is not correctly used.

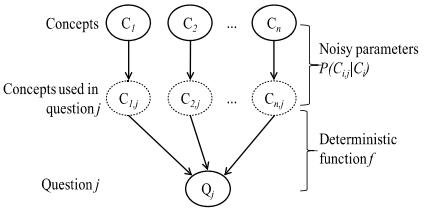


Fig. 1. Diagnostic NIDA model

The NIDA model is a psychometric model which considers the guess and slip parameters for each concept (or skill). It is different from the DINA (Deterministic Input, Noisy AND) model, which associates the guess and slip parameters to test items instead of concepts [7]. As we have to diagnose the presence or absence of each concept, our work is based on the NIDA model. The NIDA model involves a binary response model. When students give a wrong answer, the probabilities of knowing all the relevant concepts are reduced. In fact, wrong answers are usually caused by lacking some but not all the relevant concepts. We introduce the diagnostic information of wrong answers into our model. To reduce knowledge engineering effort, we use MCQs as test items. We propose a modified NIDA model (Fig. 1) to represent the relations between students' answers and their knowledge states. The deterministic function (in Fig. 1) of a binary NIDA model is logical AND. In our diagnostic NIDA model each question involves multiple characterized answers. The parameters of our diagnostic NIDA model are estimated as follows:

- Each concept C_i has two values, 1 (known by students) and 0 (unknown by students). Each used concept $C_{i,j}$ also has two values, 1 (concept C_i is correctly used in question Q_j) and 0 (concept C_i is not correctly used in question Q_j). In a MCQ, the values of question Q_j are multiple characterized options.
- There are two kinds of noisy parameters between concept C_i and used concept $C_{i,j}$. One is the slip parameter, that is, concept C_i is known, but it is not correctly used in

the question Q_j. The other is the guess parameter, that is, concept C_i is unknown, but it is correctly used in the question Q_j. The noisy parameters are as follows:

$$P(C_{i,j} = 0 | C_i = 1) = P_{slip}^{i,j} \qquad P(C_{i,j} = 1 | C_i = 0) = P_{guess}^{i,j} \tag{1}$$

- The deterministic function represents the relations between students' performances (correctly or incorrectly using concepts) and the possible answers (coded options). When a student correctly uses all the relevant concepts for answering a question, his/her answer is certainly the right option. When some concepts are not correctly used by the student, his/her answer is certainly the coded wrong option which corresponds to his/her performance. When the student's performance does not correspond to any coded option of the questions, and no more information can be obtained, we suppose that his/her answer can be any option with the same probability (see Table 1). In the example of Table 1, question Q_j involves three concepts (C_1, C_2, C_3) and the possible answers are the four coded options (A_1, A_2, A_3, A_4).

Table 1. An example of deterministic function $P(Q_j=A_k|C_{1,j}, C_{2,j}, C_{3,j})$

Used concepts ($C_{1,j}C_{2,j}C_{3,j}$)	A_1 (001)	A_2 (110)	A_3 (010)	A_4 (111)
000	1/4	1/4	1/4	1/4
001	1	0	0	0
010	0	0	1	0
011	1/4	1/4	1/4	1/4
...	...	...	...	...
111	0	0	0	1

The real situations are very complex. Students might exclude some bad options in terms of the relevant concepts which they can correctly use. In this case, some options might have higher probability to be chosen than others. Hence, if we can get more information from real educational settings, the deterministic function needs to be improved.

Our diagnostic NIDA model only requires the prior values of slip and guess parameters for each concept. Given a student's knowledge state, with the slip and guess parameters of concepts and the deterministic function, we can calculate the probability of the student choosing each option of a question by BN inference.

3 Relations between Concepts

In some ITSs, students are adapted to learn the difficult concepts only after they get the knowledge of some simple concepts. So the prerequisite relations might exist between different difficulty levels of concepts. We introduce the different levels of concepts into our diagnostic NIDA model (see Fig. 2). The trial BN contains fourteen concepts, which are classified into three levels according to the difficulty. The nodes L_1 and L_2 are the prerequisite levels of knowing the more difficult concepts, which have two values, 1 (students get the level) and 0 (students do not get the level).

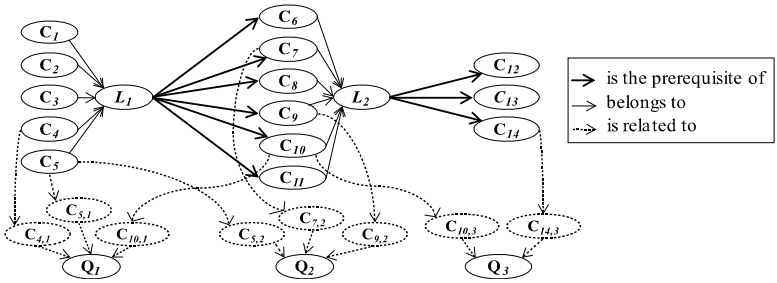


Fig. 2. A trial BN of different levels of concepts

4 Evaluation

In this part, we aim to evaluate whether the efficiency and accuracy of student modeling are improved by our diagnostic NIDA model. As empirical data is expensive, we preliminary evaluate our model by simulated students with predefined knowledge profiles. We have three experiments in the evaluation. For *experiment 1*, all the fourteen concepts are independent and there is no difference in difficulty or relation among them. The prior probability of knowing each concept is 0.5. For *experiment 2*, prerequisite relations exist between the fourteen concepts and we use the prerequisite network in [8]. The prior probability of the concepts without prerequisites is 0.75. For the concepts with prerequisites, if all the prerequisites are known, the prior probability of knowing the concepts is 0.5; otherwise, it is 0. For *experiment 3*, the relations between concepts are as Fig. 2. Students get a level if they know three or more related concepts. The prior probabilities of knowing the easy concepts are 0.75. And if students get the prerequisite level, the prior probabilities of knowing the more difficult concepts are 0.5; otherwise, it is 0.

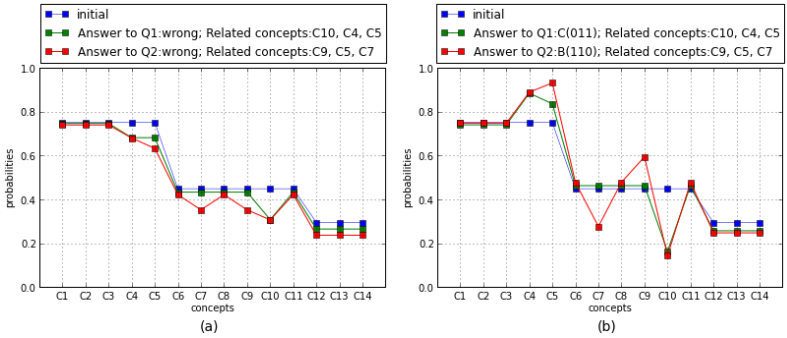


Fig. 3. Updating the probabilities of knowing the concepts in *experiment 3*: (a) binary NIDA model; (b) our diagnostic NIDA model

We randomly generate one hundred questions. Each question is related to two or three concepts. For each question, there are four options, the correct answer and three

coded wrong answers. For each experiment, we give the prior values of the slip and guess parameters. We randomly generate 180 simulated students with different predefined knowledge states. In *experiment 2* and *3*, the predefined knowledge states of students comply with the prerequisite relations between concepts or the different difficulty levels. The questions are randomly selected for students to test. We simulate students' answers in terms of their predefined knowledge states. When the student gives a right answer, the probabilities of knowing the relevant concepts are increased. When the student gives a wrong answer, according to our diagnostic BN, not all the probabilities of knowing the relevant concepts are reduced. The probabilities of the concepts which are correctly used in the wrong answer are increased and those which are not correctly used in the wrong answer are reduced (see Fig. 3).

Table 2. Rate of correctly diagnosed concepts

	Binary NIDA model	Our diagnostic NIDA model			
P_{slip}, P_{guess}	0.1	0.1		0.15	0.2
Questions	50	50	40		
Experiment 1	73.95%	96.23%	93.61%	87.33%	77.49%
Experiment 2	90.08%	98.08%	96.92%	93.64%	86.86%
Experiment 3	83.20%	97.39%	94.40%	89.60%	81.85%

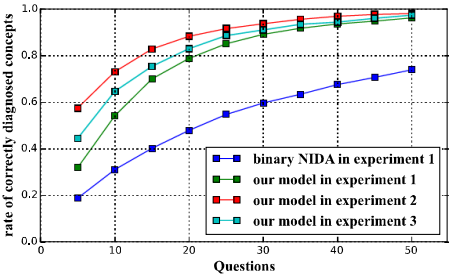


Fig. 4. Rate of correctly diagnosed concepts

After a certain number of questions are tested, the concepts can be evaluated into three categories (diagnosed as *known*, diagnosed as *unknown* and *undiagnosed*) by a threshold *e* (0.2 in the experiments). Comparing the result with the predefined knowledge state of the simulated students, the rates of *correctly diagnosed*, *incorrectly diagnosed* and *undiagnosed* concepts can be calculated. The final result is the average value of all the students' tested results. When the prior values of the slip and guess parameters are given as 0.1, in *experiment 1* (without the influence of relations between concepts), after 40 questions are tested by 180 students, 93.61% of concepts are correctly diagnosed (see Table 2), 1.19% incorrectly diagnosed and 5.20% undiagnosed by our diagnostic NIDA model. Comparing with the result of binary NIDA model tested by 50 questions, which is 73.95% correctly diagnosed, 2.37% incorrectly diagnosed and 23.68% undiagnosed, our model behaves well in improving the accuracy and efficiency of student modeling. Fig. 4 shows the rates of correctly diagnosed concepts tested by the binary NIDA model and our model with different numbers of

questions (P_{slip} and P_{guess} are 0.1). Without the influence of the relations between concepts (*experiment 1*), our diagnostic NIDA model shows a higher accuracy and efficiency in student modeling than the binary NIDA model. Fig. 4 also shows that based on our model, student modeling can be improved by introducing the relations between concepts (*experiment 2 and 3*). And in our trial BN, introducing the prerequisite relations between concepts (*experiment 2*) shows a slightly better behavior than introducing the different difficulty levels of concepts (*experiment 3*), but the latter requires less effort of knowledge engineering than the former.

5 Conclusion and Future Work

Considering the diagnostic information of students' wrong answers, we encode students' answers according to the presence or absence of the relevant concepts. We introduce the modified NIDA model to represent the relations between students' answers and their knowledge states. Our evaluation results show that comparing with binary NIDA model, the accuracy and efficiency of student modeling are improved by our model. In future work, we can improve our work in some aspects. Firstly, the prior values for the parameters are required in our model, which are usually very difficult to be acquired from expert knowledge. So we will consider how to derive them from data. And it is highly necessary to use the real data sets to evaluate our model. Secondly, some adaptive strategies will be introduced into our model to select appropriate test items. An available approach is to calculate the *utility* [3, 4] of test items.

References

1. Conati, C., Gertner, A., VanLehn, K.: Using Bayesian Networks to Manage Uncertainty in Student Modeling. User Modeling and User-Adapted Interaction 12(4), 371–417 (2002)
2. Millán, E., Pérez-de-la-Cruz, J.L., Suárez, E.: Adaptive Bayesian Networks for Multilevel Student Modelling. In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 534–543. Springer, Heidelberg (2000)
3. Millán, E., Pérez-de-la-Cruz, J.L.: A Bayesian Diagnostic Algorithm for Student Modeling and its Evaluation. User Modeling and User-Adapted Interaction 12(2-3), 281–330 (2002)
4. Collins, J.A., Greer, J.E., Huang, S.X.: Adaptive Assessment Using Granularity Hierarchies and Bayesian Nets. In: Frasson, C., Gauthier, G., Lesgold, A. (eds.) ITS 1996. LNCS, vol. 1086, pp. 569–577. Springer, Heidelberg (1996)
5. Tchétagni, J.M.P., Nkambou, R.: Hierarchical Representation and Evaluation of the Student in an Intelligent Tutoring System. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 708–717. Springer, Heidelberg (2002)
6. de la Torre, J.: A Cognitive Diagnosis Model for Cognitively Based Multiple-Choice Options. Applied Psychological Measurement 33(3), 163–183 (2009)
7. Desmarais, M.C., Baker, R.S.J.d.: A Review of Recent Advances in Learner and Skill Modeling in Intelligent Learning Environments. User Modeling and User-Adapted Interaction 22(1-2), 9–38 (2012)
8. Carmona, C., Millán, E., Pérez-de-la-Cruz, J.L., Trella, M., Conejo, R.: Introducing Prerequisite Relations in a Multi-layered Bayesian Student Model. In: Ardissono, L., Brna, P., Mitrović, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 347–356. Springer, Heidelberg (2005)

Learning Bayesian Knowledge Tracing Parameters with a Knowledge Heuristic and Empirical Probabilities

William J. Hawkins[1], Neil T. Heffernan[1], and Ryan S.J.D. Baker[2]

[1] Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA
{bhawk90,nth}@wpi.edu
[2] Department of Human Development, Teachers College, Columbia University, New York, NY
baker2@exchange.tc.columbia.edu

Abstract. Student modeling is an important component of ITS research because it can help guide the behavior of a running tutor and help researchers understand how students learn. Due to its predictive accuracy, interpretability and ability to infer student knowledge, Corbett & Anderson's Bayesian Knowledge Tracing is one of the most popular student models. However, researchers have discovered problems with some of the most popular methods of fitting it. These problems include: multiple sets of highly dissimilar parameters predicting the data equally well (identifiability), local minima, degenerate parameters, and computational cost during fitting. Some researchers have proposed new fitting procedures to combat these problems, but are more complex and not completely successful at eliminating the problems they set out to prevent. We instead fit parameters by estimating the mostly likely point that each student learned the skill, developing a new method that avoids the above problems while achieving similar predictive accuracy.

Keywords: Bayesian Knowledge Tracing Expectation Maximization Student Modeling.

1 Introduction

Within the field of Intelligent Tutoring Systems (ITSs), student modeling is important because it can help guide interaction between a student and an ITS. By having a model of student knowledge, an ITS can estimate how knowledgeable a student is of various knowledge components (or "skills") over time and use that to determine what the student needs to practice.

However, student modeling is also important to researchers. The parameters learned from BKT can be used to characterize how students learn and to evaluate ITS content. Examples of this include studying the effects of "gaming the system" on learning [8] and evaluating hint helpfulness [4], among many other studies.

While BKT is popular and useful, researchers have found problems with fitting BKT models. One such problem is identifiability: there may be multiple sets of parameters that fit the data equally well [3], making interpretation difficult. Additionally, the learned parameters may produce what is called a degenerate model, or a model

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 150–155, 2014.

that fits the data well but violates the assumptions of the approach, generally leading to inappropriate pedagogical decisions if used in a real system [1].

Two popular fitting methods in the literature, Expectation-Maximization (EM) [9] and brute force grid search, both suffer from identifiability. Additionally, EM can get stuck on local minima, and brute force comes with a high computational cost.

Researchers have attempted to deal with these issues through strategies like limiting the values brute force searching can explore [2], determining which starting values lead to degenerate parameters in EM [12], computing Dirichlet priors for each parameter and using these to bias the search [13], clustering parameters across similar skills [14], and using machine-learned models to detect two of the parameters [1].

This work introduces a simple method of estimating BKT parameters that sacrifices the precision of optimization techniques for the efficiency and interpretability of empirical estimation. Briefly, we estimate when students learn skills heuristically, and then use these estimates to help compute the four BKT parameters. Our goal is to efficiently produce accurate, non-degenerate BKT models.

2 Data

For this work, we used data from ASSISTments [7], an ITS used primarily by middle- and high-school students. In this dataset taken from the 2009-10 school year, 1,579 students worked on 61,522 problems from 67 skill-builder problem sets. The skill-builders used had data from at least 10 students, used default mastery settings (three consecutive correct answers to achieve mastery, ending the assignment), and had at least one student achieve mastery. A student's data was only included for a specific skill-builder if they answered at least three questions.

3 Methods

In this work, we developed and analyzed a new fitting procedure for BKT. We begin this section by describing BKT and then introduce our empirical approach to fitting BKT models. Finally, we describe the analyses we performed.

3.1 Bayesian Knowledge Tracing

Bayesian Knowledge Tracing [5] is a student model used in ITS research that infers a student's knowledge given their history of responses to problems, which it can use to predict future performance. Typically, a separate BKT model is fit for each skill. It assumes that a given student is always either in the known state or the unknown state for a given skill, with a certain probability of being in each. To calculate the probability that a student knows the skill given their performance history, BKT needs to know four probabilities: $P(L_0)$, the probability a student knows the skill before attempting the first problem; P(T), the probability a student who does not currently know the skill will know it after the next practice opportunity; P(G), the probability a

student will answer a question correctly despite not knowing the skill; and P(S), the probability a student will answer a question incorrectly despite knowing the skill.

According to this model, knowledge affects performance (mediated by the guess and slip rates), and knowledge at one time step affects knowledge at the next time step: if a student is in the unknown state at time *t*, then the probability they will be in the known state at time *t+1* is P(T). Additionally, BKT models typically assume that forgetting does not occur: once a student is in the known state, they stay there.

3.2 Computing Knowledge Tracing Using Empirical Probabilities

In this section, we present a new approach to fitting BKT models we call Empirical Probabilities (EP). EP is a two-step process that involves annotating performance data with knowledge, and then using this information to compute the BKT parameters.

Annotating Knowledge. The first step in EP is to annotate performance data for each student within each skill with an estimate of when the student learned the skill. We assume there are only two knowledge states: known (1) and unknown (0), and do not allow for forgetting (a known state can never be followed by an unknown state).

In this work, we use a simple heuristic for determining when a student learns a skill: we choose the knowledge sequence that best matches their performance. This is illustrated by Figure 1. A full description of this heuristic can be found online [6].

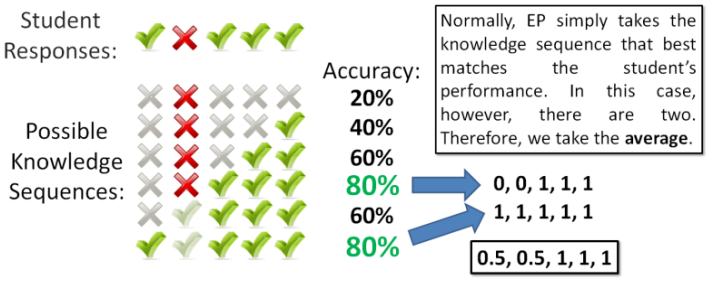


Fig. 1. Each of the six possible knowledge sequences are tried for a student's performance history, and in this case, the best two are averaged together to get the final sequence

Computing the Probabilities. Using the knowledge estimates, we were able to compute each of the four BKT parameters for each skill empirically from the data.

The first of these parameters is $P(L_0)$, the probability that the student knew the skill before interacting with the system. We can empirically estimate this by taking the average value of student knowledge on the first practice opportunity:

$$P(L_0) = \frac{\sum K_0}{|K_0|} \quad (1)$$

Equation (1) is similar to a heuristic in [11] for estimating individual student prior knowledge. While that paper used performance to compute a prior for each student as opposed to using knowledge to compute a prior for each skill as we do here, the idea that prior knowledge can be estimated mathematically in this way is similar.

Using K_i and C_i as knowledge and correctness at problem *i*, respectively, the following equations are used to compute the other three BKT parameters:

$$P(T) = \frac{\sum_{i\neq 0}(1-K_{i-1})K_i}{\sum_{i\neq 0}(1-K_{i-1})} \quad (2)$$

$$P(G) = \frac{\sum_i C_i(1-K_i)}{\sum_i(1-K_i)} \quad (3)$$

$$P(S) = \frac{\sum_i(1-C_i)K_i}{\sum_i K_i} \quad (4)$$

3.3 Experiments

In this paper, we compare BKT models fit with EM and EP in terms of predictive accuracy, model degeneracy, and training time. Due to space constraints, only the predictive accuracy results are reported here. Results for the other experiments as well as the code and data used in all the experiments are available online [6].

To fit EM, we used Murphy's Bayes Net Toolbox for MATLAB (BNT) [10]. For EM, it is necessary to specify a starting point. We chose an initial $P(L_0)$ of 0.5, and set the other three parameters to 0.1. Additionally, we set a maximum of 100 iterations and used the default BNT improvement threshold value of 0.001.

To compute the parameters using EP, we implemented the equations in the previous section in MATLAB using basic functionality. Then, we entered these values into the conditional probability tables of a BKT model constructed with BNT.

4 Results

First, we examine how predictive each method is of student performance under five-fold student-level cross-validation. We evaluated the methods using mean absolute error (MAE), root mean squared error (RMSE), and A'. These metrics were computed for each student and then used in two-tailed paired t-tests to determine the significance of the differences between the overall means of the two models. The degrees of freedom for the MAE and RMSE significance tests was one less than the number of students, whereas that of the A' significance test was lower due to some students being excluded (students who gave all correct or all incorrect answers for all skills were excluded since A' is undefined in such cases). The values below represent the average of the student metrics. Lower values of MAE and RMSE indicate better performance, whereas the opposite is true of A'. The results are shown in Table 1.

Table 1. Prediction results for the two methods of learning BKT parameters: Expectation Maximization and Empirical Probabilities

Learning Method	MAE	RMSE	A'
EM (BNT)	0.3830	0.4240	0.5909
EP	0.3742	0.4284	0.6145

Although the differences between these metrics are all statistically significant according to two-tailed paired t-tests (MAE: $t(1,578) = 10.88$, RMSE: $t(1,578) = -6.74$, A': $t(1,314) = -7.01$, $p < 0.00001$), the differences are small. Therefore, we believe the two methods are comparable in terms of predicting performance.

We also tested EM and EP in terms of model degeneracy and fitting time. In summary, we found that only EM learned degenerate parameters, and that EP runs significantly faster than EM. The full results are available online [6].

5 Conclusions and Future Work

From this work, it appears that a simple estimation of knowledge followed by computing empirical probabilities may be a reasonable approach to estimating BKT parameters. We found that EP had comparable predictive accuracy to that of EM. Additionally, it is mathematically impossible for EP to learn theoretically degenerate guess and slip rates (i.e. above 0.5) [6], and it is at least as good as EM at avoiding empirically degenerate parameters, based on tests suggested and used in [1]. We also found it was considerably faster than EM [6].

An improvement to EP would be to annotate knowledge more probabilistically. EP makes only binary inferences of knowledge based on predictive performance. For example, EP always considers incorrect responses on the first problem to be made in the unknown state, even though some of these are slips. Therefore, a more probabilistic approach may be able to produce better parameter estimates.

EP could be used as a tractable way to help improve accuracy by incrementally incorporating data into models as it becomes available during a school year. This would improve models for skills with little or no previous data and make use of student and class information. If a skill has little or no previous data, using current school year data may improve estimates of its parameters. Also, it has been shown that incorporating student [11] and class [15] information can improve predictive performance, which cannot be done before the start of a school year.

While EP achieves similar accuracy to EM and appears not to learn degenerate parameters, we did not perform any external validations of the learned parameters for either approach. Such an analysis would help determine how much we can trust EP parameters, especially when they differ from those learned by EM.

Acknowledgements. We acknowledge funding from NSF (#1316736, 1252297, 1109483, 1031398, 0742503), ONR's 'STEM Grand Challenges' and IES (# R305A120125 & R305C100024).

References

1. Baker, R.S.J.d., Corbett, A.T., Aleven, V.: More Accurate Student Modeling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 406–415. Springer, Heidelberg (2008)

2. Baker, R.S.J.d., Corbett, A.T., Gowda, S.M., Wagner, A.Z., MacLaren, B.A., Kauffman, L.R., Mitchell, A.P., Giguere, S.: Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 52–63. Springer, Heidelberg (2010)
3. Beck, J.E., Chang, K.-m.: Identifiability: A fundamental problem of student modeling. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 137–146. Springer, Heidelberg (2007)
4. Beck, J.E., Chang, K.-m., Mostow, J., Corbett, A.T.: Does help help? Introducing the Bayesian Evaluation and Assessment methodology. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 383–394. Springer, Heidelberg (2008)
5. Corbett, A., Anderson, J.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction 4, 253–278 (1995)
6. Empirical Probabilities, https://sites.google.com/site/whawkins90/publications/ep
7. Feng, M., Heffernan, N.T., Koedinger, K.R.: Addressing the assessment challenge in an Intelligent Tutoring System that tutors as it assesses. User Modeling and User-Adapted Interaction 19, 243–266 (2009)
8. Gong, Y., Beck, J.E., Heffernan, N.T., Forbes-Summers, E.: The impact of gaming (?) on learning at the fine-grained level. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 194–203. Springer, Heidelberg (2010)
9. Moon, T.K.: The expectation–maximization algorithm. IEEE Signal Process. Mag. 13, 47–60 (1996)
10. Murphy, K.: The bayes net toolbox for matlab. Computing Science and Statistics 33, 1024–1034 (2001)
11. Pardos, Z.A., Heffernan, N.T.: Modeling individualization in a bayesian networks implementation of knowledge tracing. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 255–266. Springer, Heidelberg (2010)
12. Pardos, Z.A., Heffernan, N.T.: Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. In: Baker, R.S.J.d., Merceron, A., Pavlik, P.I. (eds.) Proceedings of the 3rd International Conference on Educational Data Mining, pp. 161–170 (2010)
13. Rai, D., Gong, Y., Beck, J.: Using Dirichlet priors to improve model parameter plausibility. In: Barnes, T., Desmarais, M., Romero, C., Ventura, S. (eds.) Proceedings of the 2nd International Conference on Educational Data Mining, pp. 141–150 (2009)
14. Ritter, S., Harris, T.K., Nixon, T., Dickison, D., Murray, R.C.: Reducing the Knowledge Tracing Space. In: Barnes, T., Desmarais, M., Romero, C., Ventura, S. (eds.) Proceedings of the 2nd International Conference on Educational Data Mining, pp. 151–160 (2009)
15. Wang, Y., Beck, J.: Class vs. Student in a Bayesian Network Student Model. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 151–160. Springer, Heidelberg (2013)

Investigate Performance of Expected Maximization on the Knowledge Tracing Model

Junjie Gu, Hang Cai, and Joseph E. Beck

Department of Computer Science, Worcester Polytechnic Institute
Worcester, MA, USA
{jgu2,hcai,josephbeck}@wpi.edu

Abstract. The Knowledge Tracing model is broadly used in various intelligent tutoring systems. As it estimates the knowledge of the student, it is important to get an accurate estimate. The most common approach for fitting the model is Expected Maximization (EM), which normally stops iterating when there is minimal model improvement as measured by log-likelihood. Even though the model's predictive accuracy has converged, EM may not have come up with the right parameters when it stops, because the convergence of the log-likelihood value does not necessarily mean the convergence of the parameters. In this work, we examine the model fitting process in more depth and answer the research question: when should EM stop, specifically for the Knowledge Tracing model. While typically EM runs for approximately 7 iterations, in this work we forced EM to run for 50 iterations for a simulated dataset and a real dataset. By recording the parameter values and convergence states at each iteration, we found that stopping EM earlier leads to problems, as the parameter estimates continue to noticeably change after the convergence of the log-likelihood scores.

Keywords: Knowledge Tracing, Bayesian Networks, Intelligent Tutoring Systems, Expected Maximization.

1 Introduction

The Knowledge Tracing (KT) model is widely used in various intelligent tutoring systems. KT is based on two knowledge parameters: learning rate and prior knowledge, and two performance parameters: guess rate and slip rate. Prior knowledge is the initial probability that the student knows a particular skill, guess is the probability of guessing correctly given the student does not know the skill, slip is the probability of making a slip given the student does know the skill, and learning is the probability of learning the skill given the student does not know the skill. The goal of KT is to infer the knowledge state of students from their observed performances.

The most common model fitting procedure for KT is Expected Maximization (EM). EM is an iterative method for finding maximum likelihood or maximum a posteriori estimates of parameters in statistical models [6]. This method is guaranteed to improve the likelihood function at each iteration. In [3], the authors also claimed

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 156–161, 2014.

that KT + EM results in more accurate models than KT + BF (Brute Force). To sum up, EM has the following distinct attributes:

1. Convergence of likelihood does not equal to convergence of parameters.
2. Initial values for the parameters are critical.
3. Parameter values sometimes exhibit extremely sharp changes after convergence of log likelihood.

As the parameters of KT represent the knowledge (the prior knowledge node), intelligence (the learning node) and attitude (the guess and slip rates) of a student, obviously, an incorrect estimate of the parameters may result in a wrong evaluation of a student, possibly causing the tutor or teachers to give additional assignments. Also, researchers interpreting the models to draw scientific conclusions will reach inaccurate conclusions if the parameters are incorrect. Thus, the acquisition of the right parameter is essential, as it will give the researchers the true knowledge of how students learn. Regarding that the values of the parameters may vary at different EM iteration, it is valuable to know when to stop running EM in order to get the right parameters.

2 Methodology

There were two components to our study. The first involved simulated data. For the simulation we used 5,000 students giving 10 responses to a skill, for 50,000 total sample data points. We set up the KT parameter values for the simulated data to: prior: 0.5, learn: 0.4, guess: 0.15, slip: 0.2, based on our knowledge of student learning. The real data we considered came from the 2009-2010 school year of ASSISTments. We select those student-skill sequences with less than or equal to 10 attempted opportunities. The final dataset contains 1,775 distinct students, 123 distinct skills and 695,732 data points. The BNT toolbox [4] is used to implement EM on the KT model, and EM stops when either of the two conditions is met:

1. The slope of the log-likelihood function falls below the threshold, which is set to 10^{-3} by default.
2. The number of iterations reaches the maximum number of iterations (max_iter), 100 by default.

The first condition indicates the process should stop when the model's accuracy ceases to noticeably improve. The threshold for improvement is normally set up to a default value 10^{-3}. The second condition, typically not encountered fitting KT models, represents a model that is not behaving well, and is possibly stuck in an infinite loop. Thus, EM typically stops when the slope of the log-likelihood score reaches the threshold, which we suspect is not equivalent to convergence of parameters. As models like the Student Skill model [5] have complicated Bayesian Network structures and massive number of parameters, it is important for researchers to decide when to stop running EM, more specifically, how to set the proper max_iter and threshold for EM to search for the right parameters.

In order to know the best time for EM to stop, we set max_iter to 50 and modified the code to stop iterating only when the current number of iteration reaches max_iter. Consequently, EM will always run 50 times before its termination. At each iteration, we also recorded the parameter values, the log-likelihood scores, and checked if the log-likelihood converged using EM's default threshold to see when EM would stop normally. Based on the fact that we know in advance the real parameters of the simulated data, it is more straightforward to observe how the parameters of the KT model change over time. Meanwhile, it is still worthwhile to see how EM performs on the real dataset, since the results may have some common features with those from the simulated data. For our experiments, we used the same set of initial parameters for both the simulated and the real data: prior: 0.3, learning: 0.5, guess: 0.15, slip: 0.05.

Our hypothesis is that the parameters may still change later on after the convergence of the log-likelihood. Whether this change represents change overfitting or a better estimate of the parameter is the question we will now explore.

3 Results

We first ran our experiment on the simulated dataset. Fig. 1 shows the values of the four KT parameters for each iteration. In order to observe how the parameters change over time and how close they are to the true parameters used to generated the data, we set the initial values as the starting points and the real values as the ending points in the graph. The vertical dashed line indicates when EM would have stopped using its default stopping criteria. As we can observe, all four parameters converged by the 35th iteration, and they converged at different points. The slip rate converged comparably quickly; on the contrary, the other three parameters converged slowly, but almost at the same time. Note that the parameters still changed considerably after the dashed line, meaning we would get an inaccurate estimate of the parameters using EM's default threshold. Therefore, we argue that it is necessary to wait for all the parameters to converge before stopping EM. Finally, the parameters at the 50th iteration are very close to the true parameters, which confirms the additional iterations of EM are not causing overfitting but are actually causing the parameters to become more accurate.

We also inspected the log-likelihood values at each iteration, which was in accordance with our hypothesis that the log-likelihood value converged quickly at early iterations and only changed slightly after that. We confirm that the convergence of log-likelihood indeed does not equal to the convergence of parameters, especially for the KT model. We believe this is also the reason why EM outperforms BF, considering BF searches for the best set of parameters based only on the predictive accuracies on the test data. And there exist multiple global and local maximums for the KT model [7], and EM always push the values of parameters closer to the real values at each iteration.

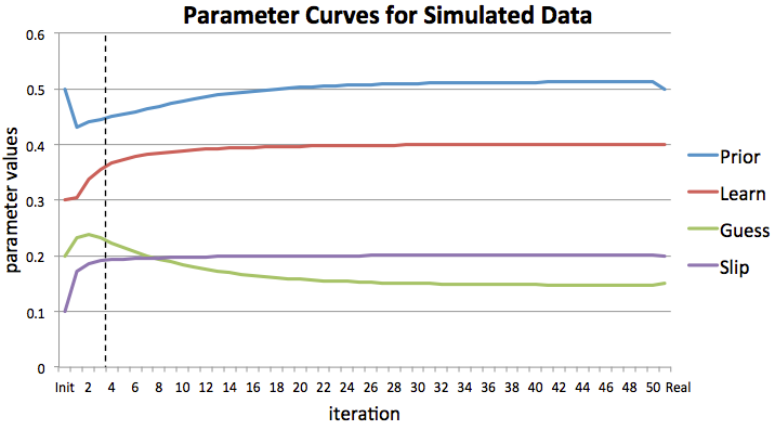


Fig. 1. Parameter curves for simulated data

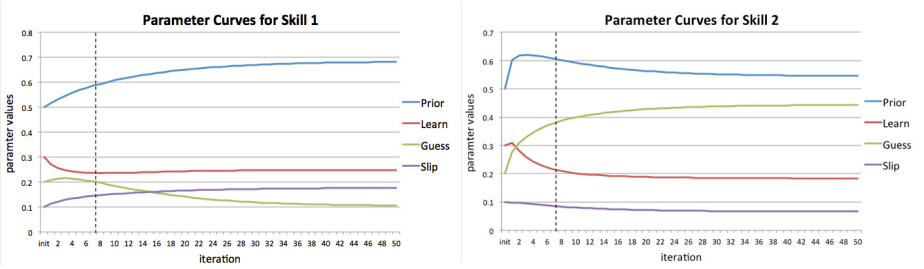


Fig. 2. Parameter curves for real data

For the simulated data, we then tested the predictive accuracies of the KT models on unseen test data. For AUC, EM's default setting gave an accuracy of 0.643, which was unchanged by running it for additional iterations. MAE slightly improved from 0.292 to 0.285, while RMSE was slightly worsened going from 0.459 for the default to 0.464 for running for 50 iterations. Based on our experience, the normal values for a KT model to predict a real dataset are around 0.35 for MAE, 0.46 for RMSE and 0.65 for AUC. Therefore, our simulated data behaves similarly to real data, and the model fit is similar to that on actual data. Although there is not much difference in predictive accuracy, Fig. 1 demonstrates that the parameter values are much closer to the real values when we let EM keep running.

We did a similar approach to examining model fit on real data. We performed a five-fold cross-validation, separating groups by skill. We compared EM under its default settings vs. the models we obtained after 50 iterations. Fig. 2 shows the parameter values at each iteration for two skills randomly selected. As these are real data, we cannot know their true parameter values. However, in agreement with the results from the simulated data (Fig. 1), the parameters continue to change after EM's default settings would cause it to stop. If the trend from Fig. 1 holds true, these

parameters are also more accurate. However, at a minimum we know there is no particular reason to believe the parameter estimates obtained after the default stopping criteria. Therefore, anyone using KT's parameter estimates for science, rather than for prediction, will encounter problems. A model stopping after 7 iterations and one stopping after 30 iterations have similar accuracy when making predictions, but in this case they make rather different claims about how quickly the skills are learned, and what students know when they begin working with the tutor. Besides the two skills showed here, we also inspected the graphs generated by all the other skills we tested, and found that generally parameter estimates were not stable at the point when EM in its default settings stopped its estimation process.

4 Conclusion and Future Work

In this work, we examined the popular model fitting process -- Expected Maximization for the Knowledge Tracing model in more depth and intended to answer the basic research question: when should EM stop. As the parameters represent the knowledge state of a student, it is crucial for the researchers to get an accurate estimate of the parameters. Although we cannot say when is the best time for EM to stop (which needs further exploration), we did find some valuable results, and most importantly, we found that stopping EM by its default threshold is definitely flawed, as the parameters still exhibit considerable changes after the convergence of the log-likelihood. From the predictive accuracy perspective, there is not much difference between the performances after 50 iterations and after default stopping, but simulation studies indicate that the parameter values are much closer to the real values when you let EM keep running. Although different datasets and parameters converge at different rates, our simulations indicate that 50 iterations are sufficient for parameter to converge. To sum up, we claim that EM definitely needs to run for more iteration to get the right values of the parameters. Overall, the results for all the datasets using both sets of initial parameters hold the following statements for the KT model:

1. Initial values (in large) do not affect the convergence of the parameters.
2. For different datasets, EM needs different number of iterations to make the parameters converging.
3. The parameters converge at different iteration and do not exhibit extremely sharp changes across one iteration after convergence of log likelihood.
4. Most importantly: *convergence in log likelihood space does not mean the convergence in the parameter space*.

The largest limitation of this work is that we only tested Expected Maximization on one particular Bayesian network model – Knowledge Tracing. However, the results may differ for other models. We intend to test EM on more invariants of the KT model like the Student Skill model, to check if the same results hold. For example, if the other models also don't show extremely sharp change after the convergence of the log-likelihood? Furthermore, although different initial values didn't make a difference in our experiments, they did affect the time for convergence. Thus we may integrate

the work with parameter plausibility such as Dirichlet priors in the future. We also wish to understand better rules for when to terminate search, and propose using the parameter curve graphs generated by the simulated data to assist searching for the parameters for the real dataset, because we believe, how the parameters change over time is also a factor in determining their true values.

Acknowledgements. We acknowledge funding from NSF (#1316736, 1252297,1109483, 1031398,0742503), ONR's 'STEM Grand Challenges' and IES (# R305A120125 & R305C100024).

References

1. Cen, H., Koedinger, K.R., Junker, B.: Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 164–175. Springer, Heidelberg (2006)
2. Corbett, A., Anderson, J.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction 4, 253–278 (1995)
3. Gong, Y., Beck, J.E., Heffernan, N.T.: Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 35–44. Springer, Heidelberg (2010)
4. Murphy, K.P.: The Bayes Net Toolbox for Matlab. Computing Science and Statistics (2007) DOI= http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html
5. Wang, Y., Heffernan, N.T.: The Student Skill Model. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 399–404. Springer, Heidelberg (2012)
6. Wikipedia, http://en.wikipedia.org/wiki/Expectation-maximization_algorithm
7. Beck, J.E., Chang, K.-m.: Identifiability: A Fundamental Problem of Student Modeling. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 137–146. Springer, Heidelberg (2007)
8. Cen, H., Koedinger, K.R., Junker, B.: Comparing two IRT models for conjunctive skills. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 796–798. Springer, Heidelberg (2008)

Understanding Wheel Spinning in the Context of Affective Factors

Joseph Beck[1] and Ma. Mercedes T. Rodrigo[2]

[1] Worcester Polytechnic Institute, Worcester, MA, USA
[2] Ateneo Laboratory for the Learning Sciences, Department of Information Systems and Computer Science, Ateneo de Manila University, Loyola Heights, Quezon City, Philippines
joseph.beck@gmail.com, mrodrigo@ateneo.edu

Abstract. The notion of wheel spinning, students getting stuck in the mastery learning cycle of an ITS without mastering the skill, is an emerging issue. Although wheel spinning has been analyzed, there has been little work in understanding what factors underlie it, and whether it occurs in cultural contexts outside that of the United States. This work analyzes data from 116 students in an urban setting in the Philippines. We found that Filipino students using the Scatterplot Tutor exhibited wheel spinning behaviors. We explore the impact of an intervention, Scooter the Tutor, on wheel spinning behavior and did not find that it had any effect. We also analyzed data from quantitative field observations, and found that wheel spinning is negatively correlated with flow, positively correlated with confusion, but not correlated with boredom. This result suggests that the problem of wheel spinning is primarily cognitive in nature, .and not related to student motivation. However, wheel spinning is positively correlated with gaming the system, so those constructs seem to be related.

Keywords: wheel spinning, affect, quantitative field observations, gaming the system.

There has been a long history of work in on mastery learning with computer-based education [1, 2], and this model makes intuitive sense and certainly realizes the maxim of "practice makes perfect," particularly as most tutors provide assistance to the student in the form of hints or breaking the problem into steps. However, a bit of thought reveals some hidden weaknesses in the model. If a student requires assistance to solve the first two problems, presenting a third with the hope the student will learn the skill could very well be a sensible strategy. If the student has been unable to solve twenty practice opportunities, and required considerable help on all of them, it is probably rather optimistic to believe that the twenty-first opportunity will enable the student to suddenly acquire the skill. Using data from 116 students in an urban setting in the Philippines [9], this paper explores wheel spinning--the phenomenon of students being stuck on a particular skill--investigates what other constructs relate to it, and discusses possible approaches for remediation.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 162–167, 2014.

1 Investigating Student Mastery in the Scatterplot Tutor

The testbed for this study was the Cognitive Tutor unit on scatterplot generation and interpretation [3]. Sixty-four of the participants (experimental) were randomly assigned to use a version of the tutor with an embodied conversational agent, "Scooter the Tutor". Scooter was designed to both reduce gaming the system and to help students learn the material that they were avoiding by gaming while affecting non-gaming students as minimally as possible [4]. In order to investigate how students mastered content in the Scatterplot Tutor, we made use of the log files recorded during the study to analyze student performance.

How did students spend their time in the Scatterplot Tutor? We separate students into three categories of learners on any given problem. The first type is those still working towards mastery. The second type is those who have just mastered the skill on that problem. The third type is those learners who have mastered the skill on a previous problem. For purposes of this paper, we use a definition of mastery to be defined as three correct responses in a row. Figure 1 shows how many students were engaged in each of these three activities for the first 20 practice opportunities of each skill. The graph goes up to 2610, since there are 24 skills in the Scatterplot Tutor, and 116 students (some students did not attempt all of the skills). Therefore, on the first practice opportunity, all students are working towards mastering the skill, as none could have mastered it yet (since the definition is three correct responses in a row). On the third practice opportunity, a fair proportion of the students master the skill. By the seventh practice opportunity, relatively few students are still working towards mastery, and those students are unlikely to master the skill. The majority of students are working on additional practice of the skills, and possibly overpractice [6]. Whether all of this overpractice is wasted or even preventable [7] is debatable; however we were surprised at the low number of students, both in absolute terms and as a relative proportion, still working towards mastering the skill by the 9th practice opportunity.

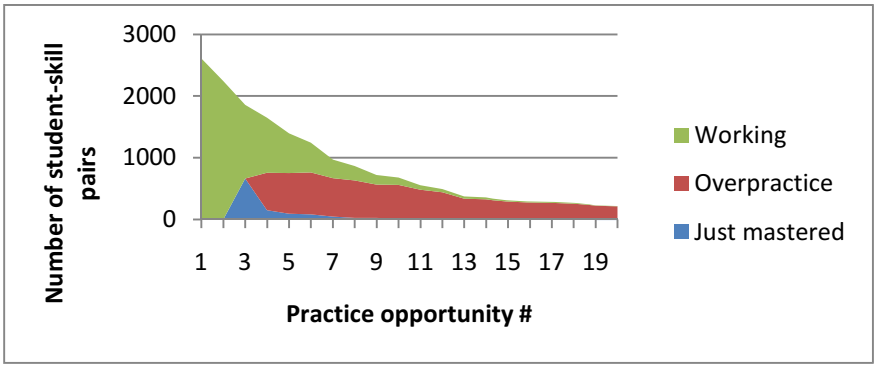


Fig. 1. Number of students engaged in each type of activity as a function of practice opportunity

Do students wheelspin within the scatterplot tutor? The phenomenon of wheel spinning [8] refers to students who fail to master a skill in a computer tutor in a timely manner. If we only consider students who attempt 3 or more problems, and wait until the 15th practice opportunity, only 63.2% of students will have mastered the skill. Waiting until 20 practice opportunities results in very few additional students mastering the skill (63.4%), as relatively few students will attempt to solve that many problems. In short, students who master the material in the Scatterplot Tutor tend to do so quickly; after 7 practice opportunities 90% of the students who will eventually master the skill have already done so.

These results, combined with Figure 1, which demonstrates that relatively few students succeed in mastering a skill relative to the number working on it, suggest that Filipino students working in the Scatterplot Tutor are capable of exhibiting wheel spinning behavior. After a student has attempted 10 problems on a skill, if he has not yet mastered it, then he has little hope of doing so through additional interaction with the ITS. For consistency with prior research, we also adopt a threshold of 10 problems for our cutpoint for wheel spinning. That is, if a student reaches 10 problem attempts on a skill without mastery, we define him as exhibiting wheel spinning behavior on that skill.

2 Understanding the Interplay of Scooter the Tutor, Affect, and Wheel Spinning

For interpreting the affect data, in order to obtain scientifically meaningful results, we restricted the data in two ways. First, as mentioned previously, we excluded students who solved a small number of problems (<60). Second, we found that certain affective states were rarely observed by our coders.

Given the lack of statistical power, and extreme non-normality of the data, we did not examine the affect states of Frustration or Surprise. That left us with Confusion, Flow, Boredom, Neutral, and Delight, as well as our measure of percent time gaming the system and percent of skills on which the student wheel spun.

Both the control and experimental groups worked with the Scatterplot Tutor. In addition, the experimental group received feedback and assistance from Scooter the Tutor. One question is whether Scooter had an impact on the affective states or on the amount of wheel spinning. The impact of Scooter on affective states has been previously studied [9], and this work replicates the finding of no statistically reliable differences as a result of Scooter. We also measured Scooter's impact on wheel spinning, and found that in both conditions the mean was 0.37; so there appears to be no impact from Scooter. Given that Scooter also included instruction, it is somewhat surprising that the rate of wheel spinning was not affected.

What is the interrelationship between affect and wheel spinning? To further explore wheel spinning, we examined how the other constructs we measured correlated with it. As we expected student incoming knowledge to directly affect both wheel spinning and affective measures such as confusion and flow, we computed partial

correlations, partialing out the student's pretest score. Table 1 provides the results of the partial correlations. Wheel spinning is moderately related to flow and confusion, both in the expected direction with flow meaning a student is less likely to wheel spin and confusion making a student more likely to wheel spin. There is also a moderately strong relationship with gaming the system. Perhaps most interestingly, boredom was not strongly related to wheel spinning, with a partial correlation of 0.145.

Table 1. Partial correlations vs. wheel spinning

Construct	Partial correlation	p-value
Flow	-0.523	1.03×10^{-8}
Confusion	0.476	2.91×10^{-7}
Gaming the system	0.437	3.24×10^{-6}
Boredom	0.145	0.14
Delight	0.053	0.59

Combined, these results suggest that students are wheel spinning not because of affective factors where they are not motivated to do the work, but rather, students are genuinely stuck on the material and need additional instructional support. To test this intuition formally, we modeled the problem in Tetrad, a freely available tool for causal discovery in datasets[1]. We restricted our analysis to only consider confusion, boredom, and flow, as these variables were the most related to wheel spinning, and they were also the states that had the most observations by the human coders. In addition, we included domain pretest score, wheel spinning, and gaming the system in the model. Figure 2 provides the result of our analysis within Tetrad. We used the Tetrad's PC search algorithm to discover the structure, and its estimator functionality to estimate the model coefficients. We first normalized the data to make the coefficient magnitudes comparable. In addition, we set as background knowledge that the pretest score was causally upstream from all of the other variables.

The interpretation of Figure 2 is that an arrow from one node to another means there is a *direct* relation between the two. There are several interesting implications from Figure 2. First, Tetrad's search agrees that wheel spinning is related to cognitive factors such as confusion, but not to boredom. Second, the search suggests that gaming is causally downstream of wheel spinning, and is a function of both affective (boredom) and cognitive (confusion) factors. This analysis of course is limited by the statistical power of the dataset, and by the variables entered into the analysis.

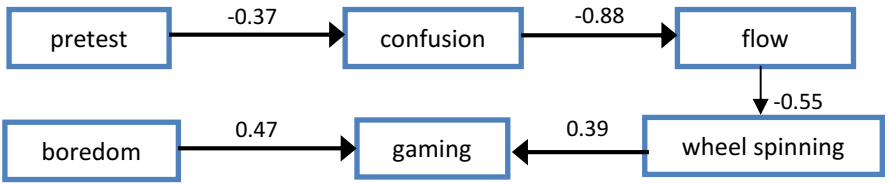


Fig. 2. Path model of wheel spinning, gaming, and affective states

[1] http://www.phil.cmu.edu/projects/tetrad/

3 Contributions, Future Work, and Conclusions

This work advances the state of knowledge for the field in several ways. First, it places the phenomenon of wheel spinning in a broader research context. Prior work was restricted to exploring students in the United States [8], while this work establishes that it occurs in at least one non-Western population. Futhermore, this work examines overpractice and wheel spinning, and finds that there are many more students engaged in overpractice than are making progress towards mastery.

This work also examined factors that could influence the rate of wheel spinning. This work replicates and extends prior research linking gaming and wheel spinning [8]. The prior research used a custom-built gaming detector that had not been well validated [10]. This work uses a well-validated detector of gaming [5] with broadly similar results in that wheel spinning and gaming appear to be linked. In addition, the direction of causality between gaming the system and wheel spinning was unclear. This work presents evidence that wheel spinning is caused by a deficit in student knowledge, which in turn causes gaming the system. In addition to cognitive factors, gaming the system also appears to be caused by affective factors, such as student boredom. These findings are consistent with prior work that found that boredom was more likely than chance to lead to gaming the system [11].

Third, this work investigated whether a tutorial intervention, Scooter the Tutor, could influence the amount of wheel spinning. Scooter addresses both behavioral issues as he is triggered by gaming behavior, as well as cognitive deficits through his instructional lessons. Although wheel spinning is related to cognitive deficits, Scooter was not found to be an effective intervention in this study.

There are several interesting next steps to take from this work. One avenue is to find an intervention that is capable of affecting the rate of wheel spinning. It would also be interesting to perform a fuller analysis of how wheel spinning relates to affective states. For this study, we were limited by the low rates of frustration and surprise in the set of analyses we were able to conduct. In particular, we suspect frustration and wheel spinning are related.

In summary, this paper investigates wheel spinning. We have found that wheel spinning exists in non-Western populations, and is related to knowledge deficits rather than student boredom. As a consequence, wheel spinning is best addressed via cognitive, rather than affective, interventions.

Acknowlegements. We thank Ryan Shaun Baker, Ma. Ofelia San Pedro, Jenilyn Agapito, Julieta Nabos, Ma. Concepcion Repalam, Salvador Reyes, Jr. Ramon Magsaysay Cubao High School, Carmela Oracion and the Ateneo Center for Educational Development, and the Ateneo Laboratory for the Learning Sciences.

References

1. Frick, T.W.: A comparison of three decision models for adapting the length of computer-based mastery tests. Journal of Educational Computing Research 6(4), 479–513 (1990)
2. Corbett, A., Anderson, J.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and user-Adapted Interaction 4, 253–278 (1995)

3. Ramon Magsaysay Cubao High School: School Profile Report, Ateneo de Manila University, Loyola Heights, Quezon City, Philippines (2009)
4. Baker, R.S.J.d., et al: Adapting to When Students Game an Intelligent Tutoring System. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 392–401. Springer, Heidelberg (2006)
5. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z.: Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System". In: Proceedings of ACM: Computer Human Interaction (2004)
6. Cen, H., Koedinger, K., Junker, B.: Is More Practice Necessary? - Improving Learning Efficiency with the Cognitive Tutor through Educational Data Mining. In: Proceedings of International Conference on Artificial Intelligence in Education (2007)
7. Fancsali, S.E., Nixon, T., Ritter, S.: Optimal and Worst-Case Performance of Mastery Learning Assessment with Bayesian Knowledge Tracing. In: Proceedings of Educational Data Mining, pp. 35–42 (2013)
8. Beck, J.E., Gong, Y.: Wheel-spinning: student who fail to master a skill. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 431–440. Springer, Heidelberg (2013)
9. Rodrigo, M.M.T., Baker, R.S.J.d., Agapito, J., Nabos, J., Repalam, M.C., Reyes, S.J., San Pedro, M.O.C.Z.: The effects of an embodied conversational agent on student affective dynamics while using an intelligent tutoring system. IEEE Transactions on Affective Computing 2(4), 18–37 (2011)
10. Gong, Y., Beck, J.E., Heffernan, N.T., Forbes-Summers, E.: The Impact of Gaming (?) on Learning. In: Proceedings of International Conference on Intelligent Tutoring Systems (2010)
11. Baker, R.S.J.d., DMello, S., Rodrigo, M.M.T., Graesser: Better to be Frustrated than Bored: The Incidence, Persistence, and Impact of Learners' Cognitive-Affective States During Interactions with Three Different Computer-Based Learning Environments. International Journal of Human-Computer Studies 68(4), 223–241 (2010)

The Usefulness of Log Based Clustering in a Complex Simulation Environment

Samad Kardan[1], Ido Roll[2], and Cristina Conati[1]

[1] Department of computer Science, University of British Columbia
[2] Centre for Teaching, Learning, and Technology, University of British Columbia
2329 West Mall, Vancouver, BC, V6T 1Z4, Canada
skardan@cs.ubc.ca

Abstract. Data mining techniques have been successfully employed on user interaction data in exploratory learning environments. In this paper we investigate using data mining techniques for analyzing student behaviors in an especially-complex exploratory environment, with over one hundred possible actions at any given point. Furthermore, the outcomes of these actions depend on their context. We propose a multi-layer action-events structure to deal with the complexity of the data and employ clustering and rule mining to examine student behaviors in terms of learning performance and effects of different degrees of scaffolding. Our findings show that using the proposed multi-layer structure for describing action-events enables the clustering algorithm to effectively identify the successful and unsuccessful students in terms of learning performance across activities in the presence or absence of external scaffolding. We also report and discuss the prominent behavior patterns of each group and investigate short term effects of scaffolding.

Keywords: Educational Data Mining, Clustering, Scaffolding.

1 Introduction

A major component of any Intelligent Tutoring System (ITS) is the learner model (see [1, 2]). The learner model is in charge of estimating the learners' proficiency and adapting the instruction accordingly. Building a learner model is especially challenging in exploratory environments and ill-defined domains in which students' responses do not have a well-defined accuracy. These environments and domains include games (e.g., Newton's Playground [3]), simulations (e.g., [4]), open-ended activities (e.g., [5, 6]), and meta-cognitive tutoring (e.g., The Help Tutor [7]), to name a few. The challenge of modeling learners becomes even more acute in complex environments, where students can engage in a variety of behaviors. One solution in these environments has been to group similar actions together. For example, in Betty's Brain [5], an environment that supports learning by drawing causal diagrams, all actions that involve editing the diagram are labeled as Edit Map. A further complication is introduced in environments which are used as platforms with a large variety of activities.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 168–177, 2014.

The current work applies a clustering approach to learning in an open-ended physics simulation which enables complex behaviors and is used with diverse activities. Specifically, we address the following research questions:

1. How can a clustering approach be applied to complex data from an exploratory learning environment?
2. What can the data mining tell us about the relationship between student behaviors in the environment, their learning, and the given activity?

We first discuss related work on clustering and describe the learning environment. We then describe the experimental design and data handling. Last, we describe the clusters and associated rules, and discuss their meaning.

2 Related Work

In the field of Educational Data Mining, clustering has been applied to different applications for discovering groups of similar users. Relevant to our work, in problem solving activities, clustering has been used to find better parameter settings for models that assess student knowledge [8], as well as discovering student learning tactics [9]. In [10] clustering and rule mining were successfully used to investigate student behaviors in an interactive simulation. However, to date, clustering and rule mining were typically applied to data from relatively constrained environments.

The current work extends the scope of using clustering by analyzing data from a high-complexity environment, the DC Circuit Construction Kit simulation, which is part of the PhET project. PhET (phet.colorado.edu [11]) is a freely-available and widely-used suite of simulations in different science and math topics. These 120 simulations are used over 45,000,000 times a year by a community of middle-school to college students. Figure 1 shows the DC Circuit Construction Kit, one of the more popular simulations of the PhET family[1]. In this specific simulation, students explore basic properties of DC circuits by connecting wires, light bulbs, resistors, switches, and measurement instruments, on a virtual test bed.

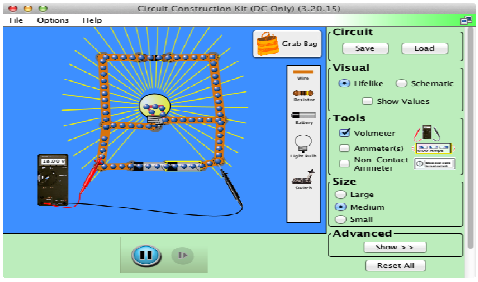


Fig. 1. The DC Circuit Construction Kit simulation

[1] http://phet.colorado.edu/en/simulation/circuit-construction-kit-dc

Several microworlds and simulations offer detailed scaffolding and explicit feedback (e.g., using cognitive tools such as hypotheses builders [4, 12]). However, PhET Simulations attempt to stay closer to an authentic inquiry environment, and thus offer neither explicit scaffolding nor explicit feedback. PhET Simulations are used as open-ended platforms for investigation. Teachers and instructors who assign the simulations to their students create their own activities, usually on paper. As a result, PhET simulations are used in a large variety of contexts and populations, using a large variety of activities. While some activities include very detailed directions for students, other activities let students explore the topic without much guidance.

3 User Study

One hundred students from first-year physics courses in a large Canadian university volunteered for a study which took place outside their normal classroom hours. The study included two activities on the topic of DC circuits, each of which took 25 minutes. The first activity focused on the effect of combining light bulbs in different arrangements. The second activity focused on the effect of combining resistors with different resistances. As PhET simulations are typically used with a large variety of activities, students were assigned to one of the two following conditions for the first activity: Low Scaffolding (LS) and High Scaffolding (HS). Students in the LS condition received only the general learning goal and a general recommendation to explore several light bulbs on the same loop, on different loops, and a combination of the two. Students in the HS condition received the same learning goal and recommendation, and in addition, were given diagrams, tables, and guiding questions. The diagrams instructed students which circuits to build; the tables asked them to document the parameters of the different circuits; and the guiding questions asked students to reflect, compare, and contrast the different circuits. The HS condition was modeled after the recommended activities for this context by the PhET project team. The study began with a short pre-test, following which students were randomly assigned to either the LS or the HS version of the light-bulb activity (see Figure 2). All students received a LS activity for the second activity on resistors. This allows us to evaluate how the same students alter their behaviors based on the given scaffolding. Last, a post-test on both activities was given, together with a survey. Three students had a perfect score on the pre-test and were removed from the analysis.

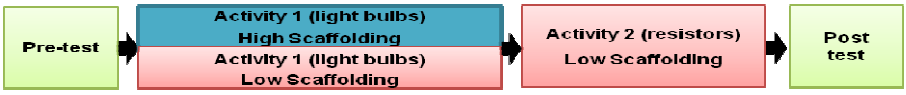


Fig. 2. The study structure

4 Analysis of User Actions

Students in the simulation work with a variety of components that include batteries, wires, light bulbs, resistors, and measurement instruments such as ammeters and

voltmeters. Overall, there are 124 different types of actions that students can perform at each moment. These actions include adding, moving, connecting, splitting, and removing components, as well as changing the attributes of components (such as resistance). Additional actions relate to the interface (such as changing views or zooming in and out), or the simulation itself (such as pausing or resetting the simulation). In addition, the outcomes of these actions depend on the state of the simulation. For example, a student will get different feedback depending on whether a testing instrument is connected to the circuit or not when s/he is changing the resistance of a resistor. This makes it quite difficult to relay on the analysis of user actions alone for the purpose of understanding the learning performance of users. In fact, clustering students according to the raw data did not support inferences about learning, as explained further below. Thus, we have constructed a multi-layer structure to capture the context of each action. In this section we introduce this structure and briefly describe the method used for behavior discovery.

In order to go beyond the raw action types recorded in the log files, we define an "action-event" as the entity that is formed by a combination of the user action and relevant contextual information. Each action-event consists of a user action (not to be confused with raw action types), the component involved in that action, the family that this specific action in the given context belongs to, and finally, outcome of the action (Figure 3). Overall, we have identified 226 action-events. Notably, these features do not create a hierarchy. For example, *joining* (action) a *wire* (component) may lead to a *current-change* (outcome) in some cases when *revising* a circuit (family), and to *no-change* (outcome) when *organizing* the circuit (family).

It is important to note that by creating this structure we have added semantic information to the data. Converting the data is done automatically by a parser which keeps track of the context (e.g., if a component is connected to the circuit) and based on over 100 conditions, assigns a value from each layer to each line of log records.

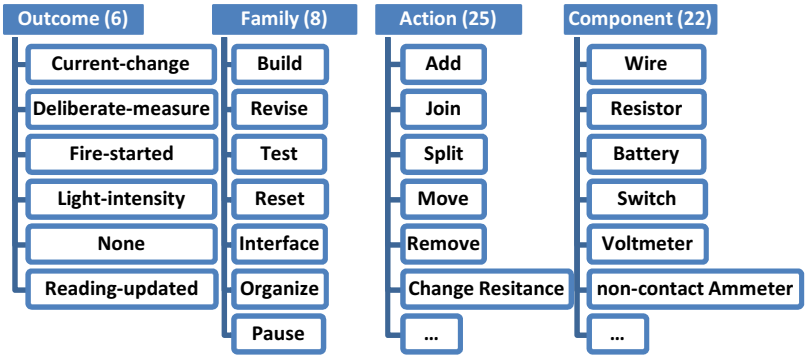


Fig. 3. The four-layer structure of the action-events

For each of the action-events we calculate three features: (i) frequency of the action-event (proportion of the number of times each action-event is used compared to total count of all action-events) denoted by _f, (ii) mean, and (iii) standard deviation of the time spent before each action-event (denoted by _m and _s respectively).

Generating features using different number of layers (e.g., Outcome only, Outcome and Family, etc.) would result in different feature-sets which contain different levels of detail about the action-events. Using only the outcome layer would generate 18 features while using all four layers of the action-event structure would result in 678 features. Interestingly, including only a subset of layers in the cluster analysis did not lead to meaningful results. Thus, we have clustered students based on all 4 layers of information (and 678 features). This highlights the importance of the semantic information that was added to the data in the preparation phase. In order to model the behaviors of the students we use the user modeling framework proposed in [10] for discovering groups of students who showed similar interaction behaviors as well as finding the representative behaviors of each group. Specifically, we look at whether the identified clusters can detect differences with regard to students' learning outcomes and the given activity (high vs. low scaffolding). The mentioned user modeling framework is used for providing support during interaction with an interactive simulation, personalized to each student's needs [10]. We will only focus on the *Behavior Discovery* phase of the framework in this paper (see [10, 13] for more details on the complete framework).

In *Behavior Discovery* user interaction data is first pre-processed into feature vectors representing each user. In our case, each vector includes the (i) frequency, (ii) mean, and (iii) standard deviation of time before each action-event (thus, 226 action-events × 3 measures per action-event = 678 features). Then, these vectors are clustered in order to identify users with similar interaction behaviors. The distinctive interaction behaviors in each cluster are identified via association rule mining [14]. This process extracts the common behavior patterns in terms of class association rules in the form of $X \rightarrow c$, where X is a set of feature-value pairs and c is the predicted class label for the data points where X applies. A confidence value is assigned to each rule calculated as the proportion of cases where X is true and class label is c over all cases where X is true. We use the Hotspot algorithm from the Weka data mining toolkit [15] for association rule mining.

In order to associate behaviors to learning performance, it is first necessary to establish how the user groups generated by clustering relate to learning. If learning performance measures are available, then we can assign a label to each cluster by comparing the average learning performance of the users in that cluster with the performance of the users in the other clusters. This is the approach we successfully adopted in [10] and will be used in this paper (see [16] for an alternative approach and related discussion). Introduction of the multi-layer action-events in this work enables us perform the clustering at different levels with different degrees of details and find the right amount of details that describes the user behaviors effectively.

5 Results and Discussion

As described in the previous section, we apply clustering on user interaction data to find groups of users in terms of how they interacted with the simulation. Similar to [10], we are interested to see if the discovered clusters of users correspond to different

levels of learning performance. However, unlike [10], employing user actions alone (i.e., either the action layer or the combination of action and component layers in the action-events structure) did not lead to meaningful results. We attribute this to the complexity of the interactions in the simulation under study here compared to the one used in [10]. Thus, we use the full 4-layer action-events structure in our analyses.

In addition to learning performance, we are interested in finding any difference in distribution of the students in the HS vs. LS conditions between the discovered clusters. Due to performing two simultaneous comparisons on the data, α for the tests (described below) is adjusted to 0.025 using Bonferroni correction. Furthermore, we will discuss the association rules describing the behaviors of users in each cluster. Our analysis first focuses on Activity 1 (A1) and Activity 2 (A2) individually and then we compare the results between the two activities.

For each activity, the optimal number of clusters is the lowest number suggested by C-index, Calinski and Harabasz [17], and Silhouettes [18] measures of clustering validity. The summary statistics of the clusters discovered for A1 and A2 are presented in Table 1 (from left the columns describe: the activity, optimal number of clusters, cluster labels (HL and LL are described later), and for each cluster: number of students, average of the standardized pre-test and post-test scores, and number of students from the HS and LS conditions). When performing clustering we faced cases in which the final clusters had only one member (singletons), therefore we had to remove the outlier user forming the singleton and repeat the clustering. This process reduced the number of students to 86 for A1 and 94 for A2.

Table 1. Summary statistics of the clusters for each activity

Activity	Number of clusters	Cluster Label	Overall Number of students	Average Pre-test Performance (SD)	Average Post-test Performance (SD)	Number of HS students	Number of LS students
A1	4	1	3	-0.9 (.1)	-1.2 (.3)	1	2
		2	3	0.7 (1.1)	1.2 (.4)	0	3
		3 (LL_1)	22	0.2 (.9)	-0.5 (1.1)	2	20
		4 (HL_1)	58	0.0 (1.1)	0.2 (.9)	42	16
A2	3	1 (LL_2)	21	-0.2 (.9)	-0.5 (.8)	11	10
		2 (HL_2)	65	0.0 (1.0)	0.2 (1.0)	36	29
		3	8	0.2 (1.2)	-0.3 (1.2)	1	7

In order to compare the learning performance of the students in each cluster we use the standardized post-test scores of the students while using pre-test scores as a covariate in our analysis (using ANCOVA). For the post-hoc analysis, the p values are again adjusted using Bonferroni correction. We apply χ^2 tests in order to see whether the distribution of students in the LS and HS conditions for the discovered clusters is different from the even distribution of the two conditions in the whole sample.

5.1 Analyzing Behaviors in Activity 1

There is a significant difference in post-test performance of the students in the four clusters ($p = .001$) with a large effect size ($\eta^2 = 0.181$) after controlling for the pre-test performance. Since the first two discovered clusters are very small ($n = 3$), we exclude

them from post-hoc analysis. For clusters 3 and 4 there is a significant difference in learning performance of students ($p = 0.006$). The students in cluster 4 are doing more than half a standard deviation better in post-test (estimated mean difference is 0.718) while there is no significant difference in pre-test scores. We will refer to the cluster 3 as Lower Learning (LL_1) and cluster 4 as Higher Learning (HL_1).

A χ^2 test on distribution of students from the LS and HS conditions shows a significant difference with the expected distribution for the four clusters discovered for A1 ($p < .001$). The same test performed only on the LL_1 and HL_1 clusters also provides similar results ($p < .001$). The majority (over 90 percent) of LL_1 students are from the LS condition. While HL_1 cluster is somewhat more balanced in terms of HS to LS ratio, it comprises over 90 percent of all students in the HS condition. The concentration of the students from the HS condition in a single cluster shows that the scaffolding provided to them encouraged them to behave similarly.

The output of association rule mining process for the LL_1 and HL_1 clusters of A1 is shown in Table 2. Rules that applied to at least 50 percent of the members of the cluster and achieved a confidence level over 0.6 were selected. Each part of the association rules is in form of a feature and a corresponding value assigned to it, for example "None.Build.join.resistor_f = Low" indicates that the (f)requency of using the resistor component when building the circuit was low.

Table 2. Selected Rules for A1 (confidence values in brackets)

A1 Cluster 3 (LL_1) 4 rules overall:
1. Reading_updated.Test.endMeasure.nonContactAmmeter_f = Low [0.625]
2. Reading_updated.Test.endMeasure.nonContactAmmeter_f = Low AND None.Build.join.seriesAmmeter_m = High [1.0]
3. Reading_updated.Test.endMeasure.nonContactAmmeter_f = Low AND None.Revise.remove.lightBulb_m = Medium [1.0]
A1 Cluster 4 (HL_1) 6 rules overall:
1. Reading_updated.Test.endMeasure.nonContactAmmeter_f = High [0.919]
2. Reading_updated.Test.endMeasure.nonContactAmmeter_f = High AND None.Build.join.resistor_f = Low [0.971]
3. Deliberate_measure.Test.startMeasure.nonContactAmmeter_f = High [0.856]

Rules 1-3 for the LL_1 cluster (Table 2) show that LL_1 students did not use one of the main measurement devices, the nonContactAmmeter, frequently enough. Rules 2 and 3 include additional conditions which are hard to interpret at this point. The HL_1 cluster includes mainly students in the HS condition. Thus, it is of no surprise that all selected rules include a frequent use of the nonContactAmmeter, which was required in order to fill out the tables successfully. Rule 2 also describes infrequent addition of a resistor. This behavior makes sense, as A1 focuses on light bulbs, and not resistors.

5.2 Analyzing Behaviors in A2

Similar to A1, there is a significant difference in post-test performance of the students in the three clusters discovered for A2 ($p = .011$) with a medium effect size ($\eta^2 = 0.096$)

after controlling for the pretest performance. The post-hoc analysis for A2 shows a significant difference in learning performance between clusters 1 and 2 (estimated mean difference in post-test is 0.646). Cluster 3 was excluded due to its small size (n =8). Similar to A1, there is no significant difference in pre-test scores between clusters 1 and 2. We will refer to the cluster 1 as Lower Learning (LL_2) and cluster 2 as Higher Learning (HL_2). Unlike A1, the χ^2 test for A2 does not show a significant difference in distribution of students to clusters by conditions. This means that the cluster analysis was not able to identify any differences among students who received different levels of scaffolding prior to the task (unlike A1, all students received the same scaffolding in A2).

Table 3. Selected Rules for A2 (confidence values in brackets)

A2 Cluster 1 (LL_2) 13 rules overall:
1. None.Build.join.lightBulb_m = Average [0.923]
2. Current_change.Revise.join.wire_f = Low AND None.Pause_f = Low AND Current_change.Revise.join.resistor_m = Low [0.778]
3. Current_change.Revise.join.wire_f = Low AND None.Pause_f = Low AND None.Test.endMeasure.nonContactAmmeter_f = Low [0.75]
A2 Cluster 2 (HL_2) 3 rules overall:
1. Current_change.Revise.join.wire_f = High [0.957]
2. None.Build.join.lightBulb_m = Low [0.853]
3. None.Build.join.lightBulb_m = Low AND Current_change.Revise.join.wire_f = High [0.978]

The selected rules extracted from LL_2 and HL_2 clusters are shown in Table 3 (with the same selection criteria used for A1). The second rule for LL_2 talks about students who do not revise circuits by adding wires, do not pause to study their outcomes, and last, when joining resistors to existing circuits, they do so rapidly. These three conditions suggest that students in the LL_2 cluster, test relatively simple circuits (without adding wires and loops to existing circuits), and do so hastily – without taking sufficient time to reflect. Rule 3 shared many of these characteristics. Students join few loops to working circuits, take only few pauses, and use one of the instrument devices, the nonContactAmmeter, only rarely. Put together, rules 2 and 3 of the LL_2 cluster match current theories of learning. To learn, students should take time to reflect, compare similar circuits, and measure the outcomes of their methods. Students in this cluster only rarely engaged in these behaviors. Notably, the rules talk about specific aspects of extending circuits and using measurement instruments (e.g., nonContactAmmeter is included, but not Voltmeter). Additional data is required to understand these characteristics of the rules.

The rules for the HL_2 cluster are at sharp contrast with the LL_2 cluster. As the first rule shows, these students often extended working circuits by adding loops. The last two rules talk about students who take little time before adding light bulbs. These rules are somewhat surprising, as the activity was about resistors and not about light bulbs. Additional data is required before these rules can be interpreted.

5.3 Comparing A1 and A2

Comparing the discovered rules for A1 and A2 helps us to understand the behaviors that are specific to an activity vs. the ones that transfer across all activities and levels of scaffolding. Additionally, such comparison can highlight the advantages and limitations of using clustering to identify learners in a complex simulation.

Overall, the rules for the four clusters show one clear trend that repeats across activities and levels of scaffolding: A frequent use of the measurement devices, and especially the nonContactAmmeter, is associated with higher learning. The converse is true, too – an infrequent use of the instrument is associated with low learning.

While the trend can be seen in three of the four clusters, it is notable that the rules themselves are dissimilar. It may be that our search space included too many similar features, so that alternative features that hold similar meanings appeared in different rule sets. An alternative explanation is that the behaviors as captured by user actions is dependent on the task, which means although users with similar learning performance tend to show similar behaviors, these behaviors vary from task to task. In this case, transferability of cluster-based user models in simulation environments may be limited. We plan to collect additional data from other simulations to evaluate the transferability of the identified behaviors. A statistical analysis of effects of changes in scaffolding levels between A1 and A2 is presented in [19].

6 Conclusions

We clustered students who worked with two activities and two levels of scaffolding in an open-ended simulation. Our results show that the clusters gave us meaningful information about learning, but only when the raw data was augmented with semantic, contextual data.

Analysis of the clusters also revealed several interesting patterns in the data. All students who received high scaffolding were clustered in the same group, suggesting that the scaffolding directed them to a certain behavioral style. Notably, students who received low levels of scaffolding were distributed across four clusters.

In addition, one main behavior was associated with better learning across activities: the frequent use of measurement devices. At the same time, while the interpretation of the rules may be similar, the actual rules are different, and thus their transferability across activities should be further studied. For example, it is not yet clear why only certain aspects of testing appear in the clusters, and not others. Furthermore, some rules remain hard to interpret. It may be that shrinking the feature list without losing semantic information may lead to more consistent rules across activities.

Acknowledgements. This work is supported by the Social Sciences and Humanities Research Council of Canada (SSHRC) grant #430-2012-0521 and by the Betty and Gordon Moore Foundation. We would like to thank the PhET project team for their assistance.

References

1. Koedinger, K.R., Corbett, A.T.: Cognitive tutors: Technology bringing learning science to the classroom. In: The Cambridge Handbook of the Learning Sciences, pp. 61–78 (2006)
2. VanLehn, K.: The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education 16, 227–265 (2006)
3. Shute, V.J., Ventura, M., Kim, Y.J.: Assessment and Learning of Qualitative Physics in Newton's Playground. The Journal of Educational Research 106, 423–430 (2013)
4. Gobert, J.D., Pedro, M.A.S., Baker, R.S.J.d., Toto, E., Montalvo, O.: Leveraging Educational Data Mining for Real-time Performance Assessment of Scientific Inquiry Skills within Microworlds. JEDM - Journal of Educational Data Mining 4, 111–143 (2012)
5. Leelawong, K., Biswas, G.: Designing Learning by Teaching Agents: The Betty's Brain System. International Journal of Artificial Intelligence in Education 18, 181–208 (2008)
6. Roll, I., Aleven, V., Koedinger, K.R.: The Invention Lab: Using a Hybrid of Model Tracing and Constraint-Based Modeling to Offer Intelligent Support in Inquiry Environments. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 115–124. Springer, Heidelberg (2010)
7. Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R.: Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. Learning and Instruction 21, 267–280 (2011)
8. Gong, Y., Beck, J.E., Ruiz, C.: Modeling Multiple Distributions of Student Performances to Improve Predictive Accuracy. In: Masthoff, J., Mobasher, B., Desmarais, M.C., Nkambou, R. (eds.) UMAP 2012. LNCS, vol. 7379, pp. 102–113. Springer, Heidelberg (2012)
9. Shih, B., Koedinger, K.R., Scheines, R.: Unsupervised Discovery of Student Strategies. In: Proceedings of the 3rd Intl. Conf. on Educational Data Mining, pp. 201–210 (2010)
10. Kardan, S., Conati, C.: A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. In: Proc. of the 4th Int. Conf. on Educational Data Mining, Eindhoven, The Netherlands, pp. 159–168 (2011)
11. Wieman, C.E., Adams, W.K., Perkins, K.K.: PhET: Simulations That Enhance Learning. Science 322, 682–683 (2008)
12. De Jong, T., Van Joolingen, W.R.: Scientific discovery learning with computer simulations of conceptual domains. Review of Educational Research 68, 179–201 (1998)
13. Kardan, S.: Data mining for adding adaptive interventions to exploratory and open-ended environments. In: Masthoff, J., Mobasher, B., Desmarais, M.C., Nkambou, R. (eds.) UMAP 2012. LNCS, vol. 7379, pp. 365–368. Springer, Heidelberg (2012)
14. Zhang, C., Zhang, S.: Association rule mining: Models and algorithms. Springer, Heidelberg (2002)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter 11, 10–18 (2009)
16. Kardan, S., Conati, C.: Comparing and Combining Eye Gaze and Interface Actions for Determining User Learning with an Interactive Simulation. In: Carberry, S., Weibelzahl, S., Micarelli, A., Semeraro, G. (eds.) UMAP 2013. LNCS, vol. 7899, pp. 215–227. Springer, Heidelberg (2013)
17. Milligan, G.W., Cooper, M.C.: An examination of procedures for determining the number of clusters in a data set. Psychometrika 50, 159–179 (1985)
18. Rousseeuw, P.J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, 53–65 (1987)
19. Roll, I., Yee, N., Briseno, A.: Students' Adaptation and Transfer of Strategies Across Levels of Scaffolding in an Exploratory Environment. In: Proc. of the 12th Intl. Conf. on Intelligent Tutoring Systems (2014)

Survival Analysis on Duration Data in Intelligent Tutors

Michael Eagle and Tiffany Barnes

North Carolina State University, Department of Computer Science,
890 Oval Drive, Campus Box 8206 Raleigh, NC 27695-8206
{mjeagle,tmbarnes}@ncsu.edu

Abstract. Effects such as student dropout and the non-normal distribution of duration data confound the exploration of tutor efficiency, time-in-tutor vs. tutor performance, in intelligent tutors. We use an accelerated failure time (AFT) model to analyze the effects of using automatically generated hints in Deep Thought, a propositional logic tutor. AFT is a branch of survival analysis, a statistical technique designed for measuring time-to-event data and account for participant attrition. We found that students provided with automatically generated hints were able to complete the tutor in about half the time taken by students who were not provided hints. We compare the results of survival analysis with a standard between-groups mean comparison and show how failing to take student dropout into account could lead to incorrect conclusions. We demonstrate that survival analysis is applicable to duration data collected from intelligent tutors and is particularly useful when a study experiences participant attrition.

Keywords: ITS, EDM, Survival Analysis, Efficiency, Duration Data.

1 Introduction

Intelligent tutoring systems have sizable effects on student learning efficiency — spending less time to achieve equal or better performance. In a classic example, students who used the LISP tutor spent 30% less time and performed 43% better on posttests when compared to a self-study condition [2]. While this result is quite famous, few papers have focused on differences between tutor interventions in terms of the total time needed by students to complete the tutor. In many studies of intelligent tutoring systems, time is simply held constant for two groups, and efficiency then boils down to comparing the number of problems each group could solve in the given time and the results of posttest measures. However, it is not clear how to factor students who were not able to complete the tutor into this analysis. In this work, we explore tutor efficiency in terms of time and performance, while taking student *dropout* (ceasing to interact with the tutor before completion) into account.

College students often use computer-based tools to complete homework assignments, but no specific time limits apply. Typical time duration distributions

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 178–187, 2014.

violate the normality assumptions of many statistical tests and measures of central tendency. Anderson, Corbett, Koedinger, and Pelletier used mean duration data to compare differences between groups of students with and without intelligent feedback in the LISP tutor [1]. The authors state that the mean times (for the control group) are underestimates, as many students in the control (no-feedback group) did not complete all assignments. In other words, if the control group persisted, the time they took to complete tasks would have been longer than the observed durations for the few high-performing students who were able to persist without feedback. This study illustrates how dropout can obscure the true impact of an intervention.

Our exploration of tutor efficiency has three important elements: performance (tutor completion percentage), duration (total time spent interacting with the tutor), and dropout (whether stopped before completion). Dropout can easily confound the results of duration and performance. Different dropout rates between experimental groups can cause attrition bias [10], where groups completing the study are self-selected due to achievement levels; this self-selection causes the sample to become different than the target population and hampers the study's generalizability [8]. When dropout exists, more complex analyses are needed to study learning efficiency; not only are results suspect for generalization purposes, but the data itself contains missing values because of high dropout rates. By modeling tutor data with high dropout rates using survival analysis, we hypothesize that we can build a more detailed understanding of tutor efficiency and explain differences between groups in an educational intervention.

In this study, we investigate data from a prior study of the Deep Thought logic tutor comparing versions with and without hints. Stamper et al. found that the odds of a student in the control group dropping out of the tutor after the first six problems were over 3.6 times higher when compared to the group provided with (data-driven and automatically generated) hints [12]. Students given access to hints also had better tutor performance, as well as higher overall course scores. However, comparison of duration means showed no differences in overall time spent in the Deep Thought logic tutor between the hint and control groups. This is likely because this comparison does not take into account student dropout. In this study, we applied survival analysis to data from Stamper et al.'s study to more fully explore the impact of hints on performance, duration, and dropout. We hypothesize that students given access to hints in the Deep Thought logic tutor, spend less time in tutor while also performing better than students without hints. In other words, the tutor efficiency for Deep Thought with hints is higher than that for they system without hints. We found that students given automatically generated hints take 55% of the time that students in the control needed to complete the tutor.

1.1 Methods and Materials

We perform our experiments on the Spring and Fall 2009 Deep Thought propositional logic tutor [6] dataset as analyzed by Stamper, Eagle, and Barnes in 2011[13]. Data was collected from six deductive logic courses, taught by three professors.

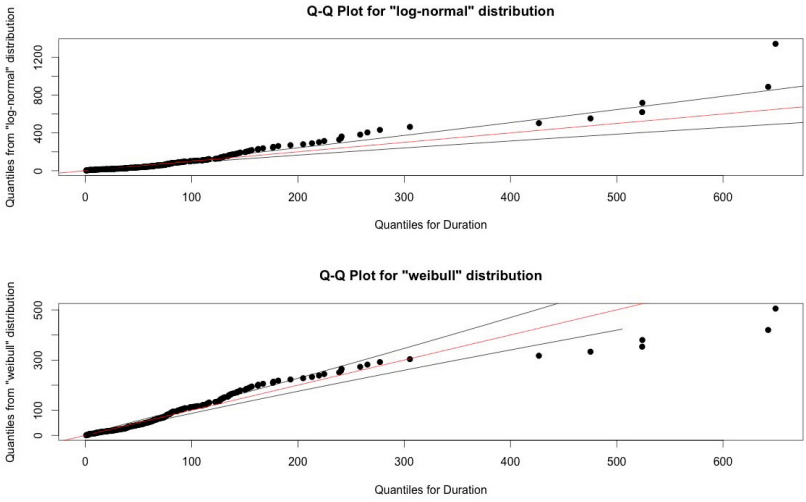


Fig. 1. QQ-Plots for the log-normal and Weibull distributions, the primary difference appears to be that the Log-normal is sensitive to very small durations, while the Weibull distribution is sensitve to very large durations

Each instructor taught one class using Deep Thought with automatically-generated hints available (hint group) and one without any additional feedback (control). The dataset includes 105 students in the Hint group and 98 students in the Control group. In Deep Thought, students choose the amount of time they spend using the online tutor; however, they were graded on the completion of 13 specific proofs.

The variables we use for this study are:

Group a two level factor (Hint, Control) depicting the student's experimental condition

ProblemDuration the sum of the time taken over all steps in a problem until 1st completed (max 3min per step)

Duration the sum of problem durations over all 13 problems

Performance a number between 0–13 representing the number of proofs solved by the student

Dropout a boolean (True, False) defined as true for students who stop engaging with the tutor without completing the assignment ($Performance \neq 13$)

Duration data often falls into a set of known distributions [3] [9]. Q-Q plots (figure 1) and histogram/density plots (figure 2) allowed us to narrow the possible distributions down to log-normal[5] or Weibull[16]. The primary difference appears to be that the log-normal does not fit well to early dropout (small durations), while Weibull does not fit as well for extremely long durations.

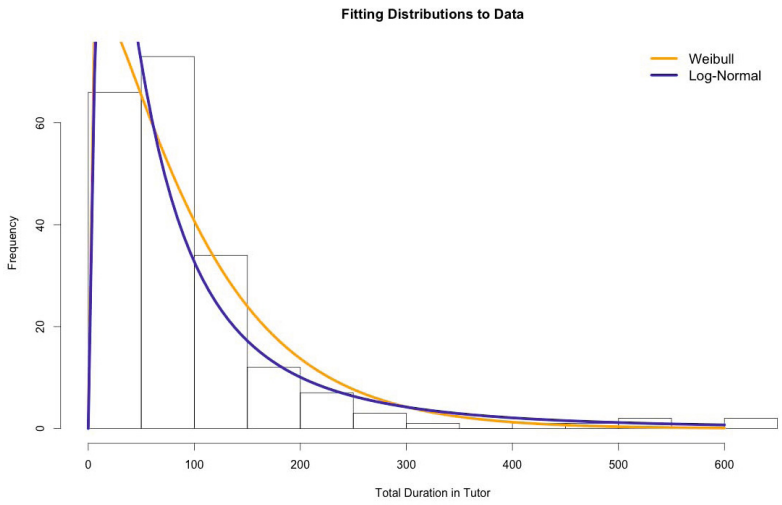


Fig. 2. Histogram with density plots for the Weibull and log-normal distributions. Both seem to fit reasonably well.

1.2 Survival Analysis

Survival analysis is a series of statistical techniques that deal with the modeling of "time to event" data [7]. Survival analysis, also known as reliability analysis or duration analysis in economics, is named for its start as a method to measure survival after applying a medical intervention.

Survival analysis includes techniques for unknown values, non-parametric data, log-normal and Weibull probability distributions, and between-groups testing. We use the survival package for R [15] to perform our analysis of learning efficiency, where the event for survival analysis is tutor completion.

"Censoring" allows for modeling duration with unknown values. Right censoring occurs when participant data is lost before tutor completion, while left censoring would be when completion time is known but start time is not. For our data, the duration for students who drop out, or stop using the tutor is right censored, since we know the start time but do not know how long it would have taken the student to complete the tutor. For example, a student who has completed 5 problems but then quits is considered right censored as we do not know how long it would have taken the student to complete all 13 problems.

The survival function is defined as:

$$S(t) = Pr(E > t) = 1 - F(t) \tag{1}$$

where t is the time in question, E is the time of the event (tutor completion), Pr is probability, F(t) is the duration distribution. This function gives the probability that the time of the tutor completion event, E, is later than t. That is, the probability that the student has not completed the tutor.

The duration distribution function, which is found via the cumulative distribution function $cdf(t)$, is the probability of observing a problem completion time E less than or equal to some time

$$F(t) = Pr(E \leq t) = 1 - S(t) = cdf(t) \tag{2}$$

The derivative of F(t) is the probability density function (pdf) of the duration distribution,

$$f(t)Pr(E = t) = F'(t) = \frac{d}{dt}F(t) = pdf(t), \tag{3}$$

which provides us with the probability of observing a single tutor completion time E at some time t. The hazard function, which tells us the instantaneous completion rate at time t, is:

$$\lambda(t) = \lim_{dt \to 0} \frac{Pr(t \leq E < t + dt | E \geq t)}{dt} = \frac{f(t)}{S(t)} = \frac{pdf(t)}{1 - cdf(t)}. \tag{4}$$

This is the probability of the event occurring at time t given that the event has not yet occurred.

There are two models we consider for measuring effects of covariates: the accelerated failure time (AFT) model and the Cox proportional hazards model. AFT assumes that the effect results in one group that completes the tutor more quickly, while the Cox proportional hazards model assumes that the tutor completion rate for one group is a constant multiple of that for the other. We have chosen the AFT model, which assumes that the effect of the covariates, θ, is to accelerate the time to tutor completion by some constant factor [14].

$$S(t|\theta) = S(\theta t) \tag{5}$$

The AFT model assumptions fit with our hypothesis that hints shorten the time it takes to finish the tutor. In addition, it is easy to interpret θ as a direct modifier to tutor completion time, and AFT facilitates using data from log-normal and Weibull distributions.

2 Results

To explore the differences between the hint and control groups we submitted the data to an AFT model as both a log-normal and Weibull distributions. The log-likelihood scores were -336.6 and -341.2 respectively. We chose to use the log-normal distribution, however both models fit similarly well and had similar results. Investigation showed the log-normal fit less well for early dropout students, while Weibull fit less well for students with extremely long durations. The probability distribution function (pdf) and cumulative distribution function (cdf) for the log-normal distribution are:

$$pdf(t) = \frac{1}{\sqrt{2\pi}\sigma t} e^{-\frac{[\ln(t)-\mu]^2}{2\sigma^2}}, cdf(t) = \Phi(\frac{\ln t - \mu}{\sqrt{2\sigma^2}}). \tag{6}$$

where Φ(x) is the cumulative distribution function (cdf) of the standard normal distribution. Note that we use $\ln(t)$ when using Φ, we can do this thanks to the assumption that the log of the duration data shows a normal distribution.

The AFT model was statistically significant $\chi^2 = 9.21$ on 1 degree of freedom, $p = 0.0024$, $n = 202$, the coefficients of the model had the intercept (mean) as 5.655, the effect of Hint θ as $-.599$, and the SD (scale) as 0.948. The effect of hints is $e^{-.599} = 0.55$; this means that it takes the Hint group 55% of the time it takes the control group to solve all 13 tutor problems. We have plotted the inverse of the survival curve in figure 4.

Figure 3 shows the hazard function for the duration data, in other words, the instantaneous completion rate for each of the groups. It also shows the probability density function for the completion rate. Overall, these plots give us a good overview of the shape of the duration data, showing that the probable total duration for students in the control group, if they were to complete the tutor, would be much longer than that for students in the hint group. One concrete measure of this is illustrated by the median of the survival function, the location where 50% of people have completed the tutor. The median is found by e^{μ}, which is $e^{5.65} = 284.29$ for the control group and $e^{5.65-.599} = 156.18$ for the hint group. Comparing these medians illustrates again the considerable difference in duration, or time to tutor completion, between the groups.

We measure the difference between groups with a Student's t test to explore possible differences in performance between the two groups. We have no reason to believe that the total tutor scores are not normally distributed. We find that the total performance in tutor between the hint group ($M = 9.26$, $SD = 4.26$) and the control group ($M = 6.78$, $SD = 4.62$) was significant, $t(200) = 3.98, p < .001, 95CI = (1.25, 3.71)$, with the Hint group solving between 1.25 and 3.71 more problems than the control group. To illustrate these differences at different points in time, we have added points to the survival curve in figure 4 indicating the mean performance score for students who left the tutor (by completing or dropping out) within the 20%, 40%, 60%, and 80% quantiles of the maximum duration. This lets us compare relative performance in the tutor between the two groups. Both groups have similar scores at about the 30 minute mark, but the hint group experiences a large increase in performance by the 60 minute mark. After this, the rate of growth in score decreases, this is likely because students that take an exceptionally long time are less skilled.

To illustrate the impact of dropout, we compare the results of survival analysis to a more traditional between-groups testing method. To explore differences in overall time in tutor between the two groups, we subjected the total elapsed time on all 13 problems to a 2-tailed Student's t-test. The total time in tutor between the hint group ($M = 86.05$, $SD = 69.80$) and the control group ($M = 122.95$, $SD = 122.94$) was not significant, $t(200) = -1.34, p = 0.183$.

However, since we know the data isn't from a normal distribution, we can improve on this accuracy by using a data transformation. To normalize the data, we use a logarithmic transformation (common log, base 10) to make the data more symmetric and homoscedastic. We subjected the log-transformed data

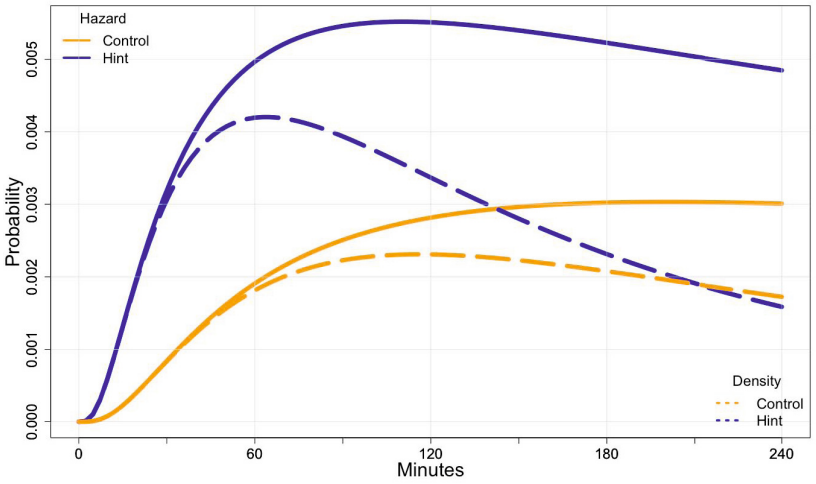


Fig. 3. The probability density functions, represented by the dashed lines, provide the probability of observing tutor completion at a specific time. The hazard functions, the solid lines, are the probability of observing tutor completion at a specific time, given that it has not occurred yet. The probability of completion grows rapidly before becoming stable and eventually decreasing.

to a 2-tailed Student t-test. The difference in the logs of duration between the hint group ($M = 8.20$, $SD = 1.02$) and the control group ($M = 8.17$, $SD = 1.21$) was not significant, $t(200) = .168, p = 0.867$. The ratio of the duration between groups is calculated by taking the difference between the means of the groups, since $\lg(x) - \lg(y) = \lg(\frac{x}{y})$. The confidence interval from the log-data estimates the difference between the population means of log transformed data. Therefore, the anti-logarithms of the confidence interval provide the confidence interval for the ratio. The anti-log of the log-transformed means provides us with the geometric mean, the anti-log of the transformed standard deviation is not interpretable. However, we can use the anti-log of the confidence intervals. The most useful statistic we can derive is the difference ratio, and its corresponding confidence intervals. A difference ratio of 0.026 between the means of the logged data equates to $10^{.026} = 1.06$ with a 95% confidence interval of CI (0.52, 2.19).

3 Discussion

The results of the survival analysis allow us to reveal striking differences between the hint and control groups in terms of the time needed to complete the tutor. Students in the hint group complete the tutor in less than half the time needed for students in the control group. It is interesting that the control group and the hint group do not have *observed* differences in overall tutor time; in other

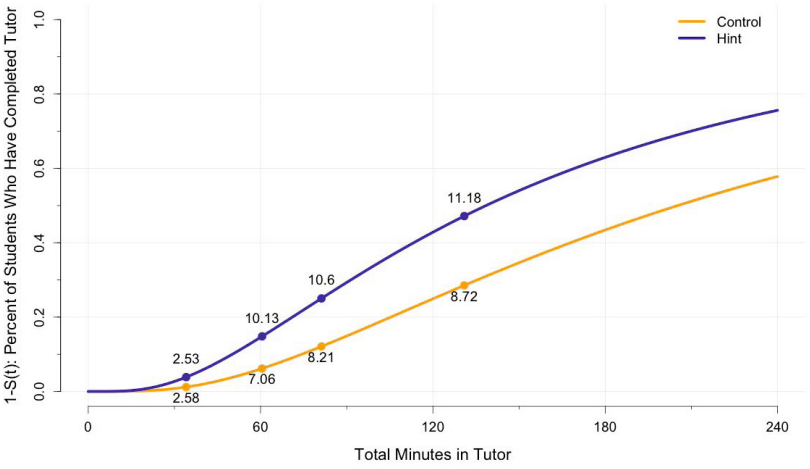


Fig. 4. The percent of students who have completed the tutor over time. We have added points with the mean tutor score (max of 13) to each curve at the 20%, 40%, 60%, and 80% quantiles of total duration.

words - students in the control group don't generally complete the tutor, so we can't observe that they would take twice as long. It is likely that, given the nature of the online access tutor, students are only willing to spend a certain amount of time on this homework assignment. This can explain the observations of differences in tutor progress observed at different times in figure 4.

Using survival analysis, we have estimated that the median duration (tutor completion time) is 284 minutes for the control group and 156 minutes for the hint group. Dividing this by the number of problems in the tutor (13) gives us an estimate of efficiency, since it gives a time per problem needed for solution. The control group is therefore spending about 21 minutes per problem on average, while the hint group is spending an average of 12 minutes per problem. Although this estimate is derived using curves to estimate the (unknown) completion time for the control group, it does in fact fit with the observed data in the first several problems, before significant dropout in the control group occurred. Given these very different rates, we can see that the control group could be discouraged by solving less than 3 problems in an hour, while the hint group could solve 5 in the same amount of time. We were in fact surprised, after realizing these estimates, that students in the control group did not drop out sooner than they did!

This back-of-the-napkin estimate of efficiency is one objective measure that suggests reasons for differences in student behavior (e.g. choice to persist or not). Perception may also play a role in explaining why students in the control group drop out. One possible reason is that the students perceive that the time they are spending is not "worth it." Breen et al. [4] defined the efficiency of a tutor,

for how the student perceives it, as the "belief or judgement that information can be accessed without wasting time or effort." Scanlon and Issroff [11] posit that computer-based instruction can conflict with the student's perceptions of division of labour within learning context. In other words, students using computer-based instruction must be more self-directed and manage their own learning. The feedback provided by the tutor with hints might have helped students in the hint group feel more directed, while also helping them when they were stuck. This could have led to improved student perceptions of efficiency.

Without survival analysis, we would not be able to use observed duration to make any conclusions regarding potential differences between the hint and control groups. Using survival analysis, we can estimate the differences between groups by accounting for unknown values - the total time it would have taken students who dropped out (in both groups) to complete the tutor. Survival analysis has also enabled us to answer questions like "How much time is needed so that 50% of the students can complete the tutor". Using the survival function $S(t) = .5$, we can estimate that the control group needs about 4.76 hours before 50% of students are done, while the hint group needs just 2.61 hours for half the group to complete the tutor. The survival function can be used to decide how much time needs to be allocated in schools for students to use a tutor. We are considering using these estimates to proactively indicate to students when they might need to seek outside help. For example, if a student has taken more than the estimated time for half of students in their group to complete the tutor, we could suggest they speak to a teaching assistant.

4 Conclusions and Future Work

As more learning systems become used outside of traditional classrooms it is imperative that educational data mining researchers leverage methods such as survival analysis that can handle non-normal data with high dropout rates. In this paper, we have used survival analysis to re-analyze the data from six 2009 logic courses using the Deep Thought logic tutor both with and without hints. The original paper showed that students without hints were over 3.6 times more likely to drop after the first six problems when compared to students offered hints. However, standard analyses were insufficient to show the impact of hints on the time needed to complete the tutor between the two groups. Using survival analysis, we have been able to estimate the total duration for both hint and control groups while taking into account dropout data, showing that students in the hint group take 55% of the time to complete the tutor than students in the control group. Using these estimates, we were able to explain approximate time per problem in the tutor for each group. This analysis sheds light on the probable reasons for dropout in the control group. Without these analyses, we might have concluded that students in the control group gave up sooner or were not persistent. However, in reality we see that these students are in fact persistent and spend a considerable amount of time in the tutor - equal to the amount of time spent in the tutor by the hint group. The difference is tutor efficiency:

students in the hint group performed more efficiently, and were therefore able to complete the tutor, while the control group spent a similar amount of time but was less likely to be able to finish. This is a much richer understanding of the differences in effects between the two groups than traditional methods provide. The survival function also allows us to make predictions on how much time is needed for tutor completion, both for teacher planning and student feedback. These results suggest that survival analysis is a powerful toolbox for investigating the impact of interventions on learning efficiency while accounting for performance, duration, and dropout.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
2. Anderson, J.R., Reiser, B.J.: The lisp tutor. Byte 10(4), 159–175 (1985)
3. Blischke, W.R., Murthy, D.P.: Reliability: Modeling, prediction, and optimization, vol. 767. Wiley (2011)
4. Breen, R., Lindsay, R., Jenkins, A., Smith, P.: The role of information and communication technologies in a university learning environment. Studies in Higher Education 26(1), 95–114 (2001)
5. Crow, E.L., Shimizu, K.: Lognormal distributions: Theory and applications, vol. 88. CRC Press, LLC (1988)
6. Croy, M.J.: Graphic interface design and deductive proof construction. J. Comput. Math. Sci. Teach. 18, 371–385 (1999)
7. Hosmer, D.W., Lemeshow, S., May, S.: Applied Survival Analysis: Regression Modeling of Time to Event Data, 2nd edn. Wiley-Interscience, New York (2008)
8. McGuigan, K.A., Ellickson, P.L., Hays, R.D., Bell, R.M.: Adjusting for attrition in school-based samples bias, precision, and cost trade-offs of three methods. Evaluation Review 21(5), 554–567 (1997)
9. Meeker, W.Q., Escobar, L.A.: Statistical methods for reliability data, vol. 314. Wiley. com (1998)
10. Miller, R.B., Hollist, C.S.: Attrition bias (2007)
11. Scanlon, E., Issroff, K.: Activity theory and higher education: evaluating learning technologies. Journal of Computer Assisted Learning 21(6), 430–439 (2005)
12. Stamper, J., Eagle, M., Barnes, T., Croy, M.: Experimental evaluation of automatic hint generation for a logic tutor. International Journal of Artificial Intelligence in Education (IJAIED) 22(1), 3–18 (2012)
13. Stamper, J.C., Eagle, M., Barnes, T., Croy, M.: Experimental evaluation of automatic hint generation for a logic tutor. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 345–352. Springer, Heidelberg (2011)
14. Therneau, T.M., Grambsch, P.M.: Modeling Survival Data: Extending the Cox Model. Springer, New York (2000)
15. Therneau, T.M.: A Package for Survival Analysis in S, R package version 2.37-7 (2014)
16. Weibull, W., et al.: A statistical distribution function of wide applicability. Journal of Applied Mechanics 18(3), 293–297 (1951)

Beyond Knowledge Tracing: Modeling Skill Topologies with Bayesian Networks

Tanja Käser[1], Severin Klingler[1],
Alexander Gerhard Schwing[1,2], and Markus Gross[1]

[1] Department of Computer Science, ETH Zurich, Switzerland
[2] Department of Computer Science, University of Toronto, Canada

Abstract. Modeling and predicting student knowledge is a fundamental task of an intelligent tutoring system. A popular approach for student modeling is Bayesian Knowledge Tracing (BKT). BKT models, however, lack the ability to describe the hierarchy and relationships between the different skills of a learning domain. In this work, we therefore aim at increasing the representational power of the student model by employing dynamic Bayesian networks that are able to represent such skill topologies. To ensure model interpretability, we constrain the parameter space. We evaluate the performance of our models on five large-scale data sets of different learning domains such as mathematics, spelling learning and physics, and demonstrate that our approach outperforms BKT in prediction accuracy on unseen data across all learning domains.

Keywords: Bayesian networks, parameter learning, constrained optimization, prediction, Knowledge Tracing.

1 Introduction

Intelligent tutoring systems (ITS) are successfully employed in different fields of education. A key feature of these systems is the adaptation of the learning content and the difficulty level to the individual student. The selection of problems is based on the estimation and prediction of the student's knowledge by the student model. Therefore, modeling and predicting student knowledge accurately is a fundamental task of an intelligent tutoring system.

Current tutoring systems use different approaches to assess and predict student performance. Two of the most popular approaches for estimating student knowledge are performance factors analysis [20] and Bayesian Knowledge Tracing (BKT) as presented by Corbett and Anderson [4].

As the prediction accuracy of a probabilistic model is dependent on its parameters, an important task when using BKT is parameter learning. Recently, the prediction accuracy of BKT models has been improved using clustering approaches [19] or individualization techniques, such as learning student- and skill-specific parameters [18,24,25] or modeling the parameters per school class [23].

Exhibiting a tree structure, BKT allows for efficient parameter learning and accurate inference. However, tree-like models lack the ability to represent the hierarchy and relationships between the different skills of a learning domain.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 188–198, 2014.

Employing dynamic Bayesian network models (DBN) has the potential to increase the representational power of the student model and hence further improve prediction accuracy. In ITS, DBNs have been used to model and predict students' performance [3,17] engagement states [2,9] and goals [3]. DBNs are also employed in user modelling [8]. In cognitive sciences, DBNs are applied to model human learning [5] and understanding [1]. Despite their beneficial properties to represent knowledge, DBNs have received less attention in student modeling as they impose challenges for learning and inference.

Recently, [12] showed that a constrained latent structured prediction approach to parameter learning yields accurate and interpretable models. Based on these findings, this paper proposes the use of DBNs to model skill hierarchies within a learning domain. Similar to [12], we use a log-linear formulation and apply a constrained optimization to identify the parameters of the DBN. We define domain-specific DBN models for five large-scale data sets from different learning domains, containing up to 7000 students. Students' age ranges from elementary school to university level. Our results show that even simple skill hierarchies lead to significant improvements in prediction accuracy of up to 10% over BKT across all learning domains. By using the same constraints and parameterizations for all experiments, we also demonstrate that basic assumptions about learning hold across different learning domains and thus our approach is easy to use.

2 Methods

Subsequently, we first give an overview of the BKT model before discussing more complex graphical models that are able to represent skill topologies.

2.1 Bayesian Knowledge Tracing

BKT models are a special case of DBNs [21] or more specifically of Hidden Markov Models (HMM), consisting of observed and latent variables. Latent variables represent student knowledge about one specific skill and are assumed to be binary, *i.e.*, a skill can either be mastered by the student or not. They are updated based on the correctness of students' answers to questions that test the skill under investigation, hence observations are also binary.

There are two types of parameters in an HMM: transition probabilities and emission probabilities. In BKT, the emission probabilities are defined by the slip probability p_S of making a mistake when applying a known skill and the guess probability p_G of correctly applying an unknown skill. The transition probabilities are described by the probability p_L of a skill transitioning from unknown to known state, while p_F is the probability of forgetting a previously known skill. In BKT, p_F is assumed to equal zero. The last parameter required to describe the BKT model is the initial probability p_0 of knowing a skill a-priori.

Employing one BKT model per skill, the learning task amounts to estimating the parameters given some observations: given a sequence of observations $\mathbf{y_m} = (y_{m,1}, ..., y_{m,T})$ with $y_{m,t} \in \{0, 1\}$ and time $t \in \{1, \ldots, T\}$ for the m-th student with $m \in \{1, \ldots, M\}$, what are the parameters $\theta = \{p_0, p_L, p_F, p_S, p_G\}$ that maximize the likelihood $\prod_m p(\mathbf{y_m} \mid \theta)$ of the available data.

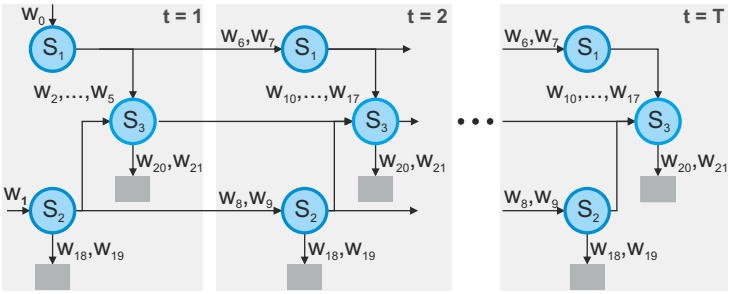


Fig. 1. Structure of the graphical model for a DBN with T time steps. Circular nodes represent skills, while the rectangles are the tasks associated with those skills.

2.2 Dynamic Bayesian Networks

When employing DBNs, we consider the different skills of a learning domain jointly within a single model. Student knowledge is again represented using binary latent variables (one per skill), which are updated based on observations associated with the skill under investigation. However, we now also model the dependencies between the different skills, *e.g.*, two skills S_A and S_B are conditionally dependent if S_A is a prerequisite for mastering S_B.

Probabilistic Notation. The learning task of a DBN model is described as follows: let the set of N variables of the model be denoted by $X = \{X_i \mid i \in \{1, \ldots, N\}\}$. In addition, let $\mathcal{H}$ denote the domain of the unobserved variables, *i.e.*, missing student answers and the binary skill variables, while $\mathcal{Y}$ refers to the observed space, disjoint from the latent space $\mathcal{H}$. During learning, we are interested in finding the parameters θ that maximize the likelihood of the observed data $\bigcup_m \mathbf{y_m}$ with $\mathbf{y_m} = (y_{m,1}, ..., y_{m,T})$ representing a sequence of T binary answers from the m-th student. The log-likelihood of a DBN [6] is then given by

$$L(\theta) = \sum_m \ln \left(\sum_{\mathbf{h_m}} p(\mathbf{y_m} \mid \mathbf{h_m}, \theta) \right), \tag{1}$$

where we marginalize over the states of the latent variables $\mathbf{h_m}$ for student m. The joint probability $p(\mathbf{y_m} \mid \mathbf{h_m}, \theta)$ of the model for student m is defined as

$$p(\mathbf{y_m} \mid \mathbf{h_m}, \theta) = \prod_i p(X_{m,i} = x_{m,i} \mid pa(X_{m,i}) = \mathbf{x_{m,pa(X_{m,i})}}) = \prod_i p_{ij_{m,i}\mathbf{k_{m,i}}}, \tag{2}$$

where $pa(X_{m,i})$ are the parents of $X_{m,i}$, while $x_{m,i}$ and $\mathbf{x_{m,pa(X_{m,i})}}$ are the realizations of the random variables $X_{m,i}$ and $pa(X_{m,i})$, *i.e.*, the states assigned to $X_{m,i}$ and $pa(X_{m,i})$ given by $(\mathbf{y_m}, \mathbf{h_m})$. Furthermore, we let $j_{i,m} := x_{m,i}$ and $\mathbf{k_{m,i}} := \mathbf{x_{m,pa(X_{m,i})}}$ to simplify the notation. Therefore, $p_{ij_{m,i}\mathbf{k_{m,i}}}$ denotes exactly one entry in the conditional probability table (CPT) of $X_{m,i}$.

Log-Linear Models. The log-likelihood of a DBN can alternatively be formulated using a log-linear model. This formulation is flexible and predominantly used in recent literature [16,22]. Therefore, we reformulate the learning task in the following. Let $\boldsymbol{\phi} : \mathcal{Y} \times \mathcal{H} \rightarrow \mathbb{R}^F$ denote a mapping from the latent space $\mathcal{H}$

and the observed space $\mathcal{Y}$ to an F-dimensional feature vector. The log likelihood from Eq. (1) can then be reformulated to

$$L(\mathbf{w}) = \sum_m \ln \left(\sum_{\mathbf{h_m}} \exp \left(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{y_m}, \mathbf{h_m}) - \ln(Z) \right) \right), \tag{3}$$

where Z is a normalizing constant and $\mathbf{w}$ denotes the weights of the model. Next, we show that this log-linear formulation of the log-likelihood is equivalent to the traditional notation. Comparing Eq. (3) to Eq. (1), it follows that

$$\prod_i p_{ij_{m,i}\mathbf{k_{m,i}}} = \frac{1}{Z} \exp \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{y_m}, \mathbf{h_m}) = \frac{1}{Z} \exp \sum_i w_i^\top \phi_i(\mathbf{y_m}, \mathbf{h_m}), \tag{4}$$

and therefore

$$\forall i, j, \mathbf{k} : p_{ij\mathbf{k}} = \frac{1}{Z} \exp w_i^\top \phi_i(\mathbf{x}), \tag{5}$$

where $\mathbf{x}$ are the realizations of all random variables in X with $j \in \mathbf{x}$ and $\mathbf{k} \subset \mathbf{x}$. A feature vector $\boldsymbol{\phi}$ and weights $\mathbf{w}$ that fulfill Eq. (5) can be specified as follows: consider the CPT describing the relationship between a node X_A and its $n-1$ parent nodes $pa(X_A)$. The CPT for these n nodes contains 2^n entries. Let $\mathbf{k} \in \{0,1\}^{n-1}$ denote one possible assignment of states to the parent nodes $pa(X_A)$. We can therefore define $p(X_A = 1 \mid pa(X_A) = \mathbf{k}) = 1 - p(X_A = 0 \mid pa(X_A) = \mathbf{k}) = 1 - p_{A,0,\mathbf{k}}$. To continue, let $p_{A,x_A,\mathbf{k}} = \frac{1}{Z} \exp w_{A,\mathbf{k}}(1 - 2x_A) = \exp w_{A,\mathbf{k}}(1-2x_A)/(\exp w_{A,\mathbf{k}} + \exp(-w_{A,\mathbf{k}}))$, which leads to the feature function $\phi_A(x) = 1 - 2x_A$. We therefore obtain the joint distribution as a product of the exponential terms which translates to a weighted linear combination of feature vector entries in the exponent and thus fulfills Eq. (5). From this formulation also follows that we need 2^{n-1} parameters to specify a CPT including n skills.

Optimization. In contrast to HMMs, the learning task for DBNs is not computationally tractable. However, [22] showed that a convex approximation admits efficient parameter learning. Note that interpretability of the parameters is not ensured, since guarantees exist only for converging to a local optimum. Recently, [12] extended the approach presented by [22] to include constraints on parameters and demonstrated that the constrained optimization increases prediction accuracy on unseen data while yielding interpretable models. Using the log-linear formulation, the algorithm presented in [12] can be directly applied to learn the parameters of a DBN model.

DBN Specification. Next, we illustrate the specification of a simple DBN. Similarly to BKT, we can interpret the parameters of a DBN in terms of a learning context. To specify the CPTs of the example DBN in Fig. 1, we employ $F = 22$ weights that can be associated with a parameter set θ. We subsequently use $\simeq$ to denote proportionality in the log domain; *i.e.*, $w \simeq p$ is equivalent to $w \propto \exp p$. Let O_3 denote the task associated with skill S_3. Then the parameters $w_{20} \simeq p(O_3 = 0 \mid S_3 = 0) = 1 - p_G$ and $w_{21} \simeq p(O_3 = 0 \mid S_3 = 1) = p_S$ represent the guess and slip probabilities. Similarly, w_{18} and w_{19} are associated with p_G and p_S as evident from Fig. 1. Furthermore, parameters $w_6 \simeq p$

$(S_{1,t} = 0 \mid S_{1,t-1} = 0) = 1 - p_L$ and $w_7 \simeq p(S_{1,t} = 0 \mid S_{1,t-1} = 1) = p_F$ are associated with learning and forgetting; the same holds true for w_8 and w_9.

Skills S_1 and S_2 are prerequisites for knowing skill S_3, *i.e.*, the probability that skill S_3 is mastered in time step t depends not only on the state of skill S_3 in the previous time step, but also on the states of S_1 and S_2 in the current time step. Therefore $w_{10} \simeq p(S_{3,t} = 0 \mid S_{3,t-1} = 0, S_{1,t} = 0, S_{2,t} = 0) = 1 - p_{L0}$, where p_{L0} denotes the probability that the student learns S_3 despite not knowing S_1 and S_2. Also, $w_{17} \simeq p(S_{3,t} = 0 \mid S_{3,t-1} = 1, S_{1,t} = 1, S_{2,t} = 1) = p_{F1}$, the probability of forgetting a previously learnt skill. Furthermore, we set $w_l \simeq 1 - p_{LM}$ if $l \in \{11, 12, 13\}$ and $w_l \simeq 1 - p_{FM}$ if $l \in \{14, 15, 16\}$, where p_{LM} denotes the probability that the student learns S_3 given that he knows at least one of the precursor skills of S_3. Moreover, p_{FM} is the probability that the student forgets the previously known skill S_3, when either S_1 or S_2 or none of them are known.

Finally, the parameters w_l with $l \in \{2, 3, 4, 5\}$ describe the dependencies between the different skills. We let $w_l \simeq 1 - p_{P0}$, if $l \in \{2, 3, 4\}$ and $w_5 \simeq p_{P1}$, where p_{P0} is the probability of knowing a skill despite having mastered only part of the prerequisite skills and p_{P1} denotes the probability of failing a skill given that all precursor skills have been mastered already. Moreover, we refer to the probability of knowing a skill a-priori via p_0. Note that w_0 and w_1 are associated with p_0. The example DBN can therefore be described by the parameter set $\theta = \{p_0, p_G, p_L, p_F, p_{L0}, p_{F1}, p_{LM}, p_{FM}, p_{P0}, p_{P1}\}$. Importantly, the method proposed in this work is independent of the exact parametrization used. Therefore, the parametrization introduced here could be easily extended.

3 Results and Discussion

We show the benefits of DBN models with higher representational power on five data sets from various learning domains. The data sets were collected with different tutoring systems and contain data from elementary school students up to university students. We compare the prediction accuracy of DBNs modeling skill topologies with the performance of traditional BKT models.

Fitting the BKT models was done using [25], applying skill-specific parameters and using gradient descent for optimization. As described in [25], we set the forget probability p_F to 0, while p_S and p_G are bounded by 0.3. In the following, we will denote this constrained BKT version as BKT_C.

We used constrained latent structured prediction [12] to learn the parameters of the DBNs. All models are parametrized according to Sec. 2.2 and we impose the constraints described in the following on the parameter set θ of the different models to ensure interpretable parameters. For our first constraint set $\mathcal{C}_1$, we let $p_D \leq 0.3$ for $D \in \{G, S, L, F, L0, F1\}$ to ensure that parameters associated with guessing, slipping, learning and forgetting remain plausible. The constraints on θ can be directly turned into constraints on $\mathbf{w}$. For the example DBN (Fig. 1), the constraints translate into the following linear constraints on the weights for $\mathcal{C}_1$: $w_i \geq 0.4236$, if $i \in \{6, 8, 10, 18, 20\}$ and $w_i \leq -0.4236$, if $i \in \{7, 9, 17, 19, 21\}$. For the second constraint set $\mathcal{C}_2$, we augment $\mathcal{C}_1$ by limiting $p_D \leq 0.3$ if $D \in$

$\{LM, FM, P0, P1\}$, yielding $w_i \geq 0.4236$, if $i \in \{2, 3, 4, 11, 12, 13\}$ and $w_i \leq -0.4236$, if $i \in \{5, 14, 15, 16\}$ for the example DBN (Fig. 1). The additional constraints ensure that parameters are consistent with the hierarchy assumptions of the model. The constraint sets $\mathcal{C}_3$ and $\mathcal{C}_4$ bound the same parameters as $\mathcal{C}_1$ and $\mathcal{C}_2$, but are more restrictive by replacing 0.3 by 0.2. Note that constraints were selected according to previous work [4]. The presented work is, however, independent of the selected constraint sets.

Prediction is performed as follows: we assume the observation at time $t = 1$ to be given and predict the outcome at time t with $t \in \{2, ..., t\}$ based on the previous $t - 1$ observations. The number of observations t for the different experiments is the minimal number of observations covering all skills of the according experiment. To assess prediction accuracy, we provide the following error measures: root mean squared error (RMSE), classification error CE (ratio of incorrectly predicted student successes and failures based on a threshold of 0.5) and the area under the roc curve (AUC). All error measures were calculated using cross-validation. Statistical significance was computed using a two-sided t-test, correcting for multiple comparisons (Bonferroni-Holm).

Note that we selected skills, where users showed low performance for our experiments, in order to make learning and prediction more challenging. In the following, we describe the DBN models for the five data sets and discuss the prediction accuracy for our models as well as for BKT_C.

Number Representation. For the first experiment, we use data collected from `Calcularis`, an intelligent tutoring system for elementary school children with math learning difficulties [10]. The data set contains log files of 1581 children with at least 5 sessions of 20 minutes per user. `Calcularis` represents student knowledge as a DBN consisting of different mathematical skills [11,13].

The graphical model used in this experiment (see Fig. 1) is an excerpt of the skill model of `Calcularis` described in [11]. Skill S_1 represents knowledge about the Arabic notation system. `Calcularis` does not contain any tasks associated with this skill. The ability to assign a number to an interval is denoted by S_2. The task associated with this skill is to guess a number in as few steps as possible. Finally, S_3 denotes the ability to indicate the position of a number on a number line. We used a maximum of $T = 100$ observations per child for learning and prediction and specified the CPTs of the graphical model with $F = 22$ weights.

Prediction errors for the constraint sets $\mathcal{C}_1$ to $\mathcal{C}_4$ as well as BKT_C are given in Tab. 1. The constrained DBN approach yields significant and large improvements in prediction accuracy compared to BKT_C. We highlight the improvement in accuracy by 11.4% ($\text{CE}_{BKT_C} = 0.3141$, $\text{CE}_{\mathcal{C}_2} = 0.2783$) and the reduction of the RMSE by 3.8% ($\text{RMSE}_{BKT_C} = 0.4550$, $\text{RMSE}_{\mathcal{C}_4} = 0.4378$). Also note the large improvement achieved in AUC ($\text{AUC}_{BKT_C} = 0.5975$, $\text{AUC}_{\mathcal{C}_2} = 0.7093$).

Subtraction. The second experiment is based on the same data set as the first experiment. This time, however, we investigate subtraction and number understanding skills. The graphical model (see Fig. 2(a)) is again an excerpt of the skill model [11] of `Calcularis`. Subtraction skills are ordered according to

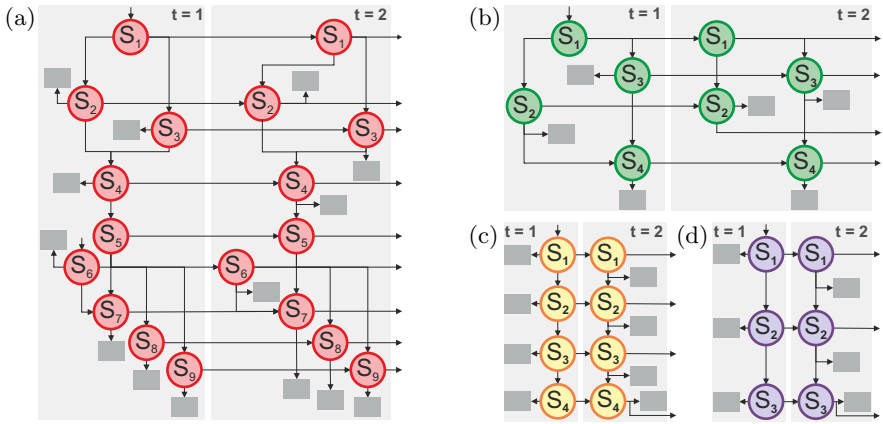


Fig. 2. Graphical models for the subtraction (a), physics (b), algebra (c) and spelling learning (d) experiments. Circular nodes represent skills, while the rectangles are the tasks associated with those skills.

their difficulty, which is determined by the magnitude of involved numbers, task complexity and the means allowed to solve a task. Skills S_1 (*e.g.*, 48-6=?), S_2 (*e.g.*, 48-9=?), S_3 (*e.g.*, 48-26=?), S_4 (*e.g.*, 48-29=?) and S_5 denote subtraction tasks in the number range 0−100. We emphasize that there are no observation nodes associated with S_1 and S_5. The number understanding skill S_6 represents knowledge about the relational aspect of number (number as a difference between other numbers) in the number range 0−1000. Finally, skills S_7 (*e.g.*, 158-3=?), S_8 (*e.g.*, 158-3=?) and S_9 (*e.g.*, 158-9=?) represent subtraction in the number range 0−1000. The difference between S_7 and S_8 lies in the means allowed to solve the task. A maximum of $T = 100$ observations per child is used for learning and prediction. Specification of the CPTs for the model requires $F = 86$ weights.

The resulting prediction accuracy for this experiment (see Tab. 1) again demonstrates that the DBN model outperforms BKT$_C$. With a reduction of the RMSE by 3.5% (RMSE$_{BKT_C} = 0.4368$, RMSE$_{\mathcal{C}_2} = 0.4215$) and an increase of the accuracy by 8.4% (CE$_{BKT_C} = 0.2818$, CE$_{\mathcal{C}_4} = 0.2580$), improvements confirm the results observed in the first experiment. Also the growth in AUC (AUC$_{BKT_C} = 0.5996$, AUC$_{\mathcal{C}_4} = 0.6916$) is again substantial.

Physics. This experiment is based on the 'USNA Physics Fall 2005' data set accessed via DataShop [15]. Data originate from 77 students of the United States Naval Academy and were collected from `Andes2`, an intelligent tutoring system for physics [3]. The tutor uses rule-based algorithms to build solution graphs that identify all possible solutions to the different tasks. Based on these graphs, a Bayesian network is constructed to assess the general physics knowledge of the student as well as the progress for the problem at hand.

We use the different modules of the data set as skills for our experiment. The graphical model is depicted in Fig. 2(b). Note that we intentionally use a simplified skill model to avoid introducing incorrect assumptions and to assess

Table 1. Prediction accuracy of the experiments, comparing BKT_C with different constraint sets for the DBNs. Numbers in bold denote a significant improvement compared to BKT_C. The best result for each error measure is marked (*).

		BKT_C	$\mathcal{C} = \mathcal{C}_1$	$\mathcal{C} = \mathcal{C}_2$	$\mathcal{C} = \mathcal{C}_3$	$\mathcal{C} = \mathcal{C}_4$
Number representation	RMSE	0.4550	**0.4469**	**0.4452**	**0.4416**	**0.4378***
	CE	0.3141	0.3279	**0.2783***	0.3079	**0.2831**
	AUC	0.5975	**0.7072**	**0.7093***	**0.7087**	**0.7049**
Subtraction	RMSE	0.4368	0.4417	**0.4215***	0.4389	**0.4216**
	CE	0.2818	0.2812	**0.2588**	**0.2757**	**0.2580***
	AUC	0.5996	**0.6157**	**0.6870**	**0.6332**	**0.6916***
Physics	RMSE	0.4530	0.4521	**0.4272**	**0.4465**	**0.4244***
	CE	0.2930	**0.2893**	**0.2677**	**0.2870**	**0.2616***
	AUC	0.5039	**0.6511**	**0.6971**	**0.6795**	**0.7007***
Algebra	RMSE	0.3379	**0.3335**	**0.3254***	**0.3321**	**0.3267**
	CE	0.1461	0.1466	**0.1392**	0.1466	**0.1379***
	AUC	0.5991	**0.6682**	**0.7004**	**0.6718**	**0.7007***
Spelling	RMSE	0.4504	0.4521	**0.4495**	**0.4492**	**0.4472***
	CE	0.2898	0.2893	0.2914	0.2882*	0.2906
	AUC	0.5029	**0.5695**	**0.5771**	**0.5735**	**0.5804***

if even non-experts can exploit skill structures using our proposed methods. The model consists of the following modules: "Vectors" (S_1), "Translational Kinematics" (S_2), "Statistics" (S_3) and "Dynamics" (S_4). These modules consist of more complex tasks for the given topic, *i.e.*, calculating total forces in a system (see example in [3]). A maximum of $T = 500$ observations per child are considered for learning and prediction and the model is described by $F = 33$ weights.

In this experiment, the benefits of the DBN model are again high (see Tab. 1): the accuracy is increased by 10.7% ($\text{CE}_{BKT_C} = 0.2930$, $\text{CE}_{\mathcal{C}_4} = 0.2616$) while the RMSE is reduced by 6.3% ($\text{RMSE}_{BKT_C} = 0.4530$, $\text{RMSE}_{\mathcal{C}_4} = 0.4244$) and the AUC grows to 0.7007 ($\text{AUC}_{BKT_C} = 0.5039$).

Algebra. For this experiment we used data from the KDD Cup 2010 Educational Data Mining Challenge (*http://pslcdatashop.web.cmu.edu/KDDCup*). The data set contains log files of 6043 students that were collected by the `Cognitive Tutor` [14], an intelligent tutoring system for mathematics learning. The student model applied in this system is based on BKT.

We use the units of the 'Bridge to Algebra' course as skills for our experiment and select four units of increasing difficulty, where students have to solve word problems involving calculations with whole numbers. The graphical model for this experiment is illustrated in Fig. 2(c). Skill S_1 (*e.g.*, $728624 - 701312$) denotes written addition and subtraction tasks without carrying/borrowing, while S_2 involves carrying/borrowing (*e.g.*, $728624 - 703303$). S_3 (*e.g.*, 33564×18) and S_4 (*e.g.*, $10810 \div 46$) represent long multiplications and divisions. Note that the skill model is again simplified for the reasons explained in the Physics experiment. We use a maximum of $T = 500$ observations per student for learning

and prediction and specify the CPTs of the model employing $F = 29$ weights. Similarly to the previous experiments, DBN significantly outperforms BKT_C (see Tab. 1). The RMSE is reduced by 3.7% ($\text{RMSE}_{BKT_C} = 0.3379$, $\text{RMSE}_{\mathcal{C}_2} = 0.3254$), while accuracy is increased by 5.6% ($\text{CE}_{BKT_C} = 0.1461$, $\text{CE}_{\mathcal{C}_4} = 0.1379$) and the AUC increases to 0.7007 ($\text{AUC}_{BKT_C} = 0.5991$). Note that DBN and BKT_C both perform better than in the other experiments as the high performance of students in the involved skills makes learning and prediction easier.

Spelling Learning. The last experiment uses data collected from `Dybuster`, an intelligent tutoring system for elementary school children with dyslexia [7]. The data set at hand contains data of 7265 German-speaking children. `Dybuster` groups the words of a language into hierarchically ordered modules with respect to their frequency of occurrence in the language corpus as well as a word difficulty measure. The latter is computed based on the word length, the number of dyslexic pitfalls and the number of silent letters contained in the word.
We use these modules as skills to build our graphical model (see Fig. 2(d)). Skills S_1, S_2 and S_3 denote the modules 3, 4 and 5 within `Dybuster`. Word examples are "warum" ("why", S_1), "Donnerstag" ("Thursday", S_2) and "Klapperschlange" ("rattlesnake", S_3). We use a maximum of $T = 200$ observations per child for the learning and prediction tasks and parametrize the model using $F = 21$ weights. While the DBN model still significantly outperforms BKT_C in this experiment (see Tab. 1), the magnitudes of improvement are small: the RMSE is reduced by 0.7% ($\text{RMSE}_{BKT_C} = 0.4504$, $\text{RMSE}_{\mathcal{C}_4} = 0.4472$), the highest AUC amounts to 0.5804 ($\text{AUC}_{BKT_C} = 0.5029$) and there is no significant improvement in CE.

Discussion. The results demonstrate that more complex DBN models outperform BKT in prediction accuracy. For hierarchical learning domains, CE can be reduced by 10%, while improvements of RMSE by 5% are feasible. The DBN models generally exhibit a significantly higher AUC than BKT, which indicates that they are better at discriminating failures from successes. As expected, adding skill topologies has a much smaller benefit for learning domains that are less hierarchical in nature (such as spelling learning). The results obtained on the physics and algebra data sets show that even simple hierarchical models improve prediction accuracy significantly. A domain expert employing a more detailed skill topology and more complex constraint sets could probably obtain an even higher accuracy on these data sets. The use of the same parameterization and constraint sets for all experiments demonstrates that basic assumptions about learning hold across different learning domains and thus the approach is easy to use.

4 Conclusion

In this work, we showed that prediction accuracy of a student model is increased by incorporating skill topologies. We evaluated the performance of our models on five data sets of different learning domains and demonstrated that the DBN models outperform the traditional BKT approach in prediction accuracy

on unseen data. To conclude, our results show that modeling skill topologies is beneficial and easy to use, as even simple hierarchies and parameterizations lead to significant improvements in prediction accuracy. In the future, we would like to analyze the influence of the skill hierarchies and the different parameters in detail. We furthermore plan to apply the individualization techniques used in BKT [18,24,25] to DBNs. Moreover, we would like to explore further modelling techniques such as dynamic decision networks.

Acknowledgements. This work was funded by the CTI-grant 11006.1.

References

1. Baker, C., Tenenbaum, J.B., Saxe, R.: Bayesian models of human action understanding. In: Proc. NIPS (2005)
2. Baschera, G.-M., Busetto, A.G., Klingler, S., Buhmann, J.M., Gross, M.: Modeling Engagement Dynamics in Spelling Learning. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 31–38. Springer, Heidelberg (2011)
3. Conati, C., Gertner, A., VanLehn, K.: Using Bayesian Networks to Manage Uncertainty in Student Modeling. UMUAI (2002)
4. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. UMUAI (1994)
5. Frank, M.C., Tenenbaum, J.B.: Three ideal observer models for rule learning in simple languages. Cognition (2010)
6. Friedman, N., Murphy, K., Russell, S.: Learning the structure of dynamic probabilistic networks. In: Proc. UAI (1998)
7. Gross, M., Vögeli, C.: A Multimedia Framework for Effective Language Training. Computer & Graphics (2007)
8. Horvitz, E., Breese, J., Heckerman, D., Hovel, D., Rommelse, K.: The Lumière Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users. In: Proc. UAI (1998)
9. Käser, T., Baschera, G.M., Busetto, A.G., Klingler, S., Solenthaler, B., Buhmann, J.M., Gross, M.: Towards a Framework for Modelling Engagement Dynamics in Multiple Learning Domains. IJAIED (2012)
10. Käser, T., Baschera, G.M., Kohn, J., Kucian, K., Richtmann, V., Grond, U., Gross, M., von Aster, M.: Design and evaluation of the computer-based training program Calcularis for enhancing numerical cognition. Front. Psychol. (2013)
11. Käser, T., Busetto, A.G., Solenthaler, B., Baschera, G.M., Kohn, J., Kucian, K., von Aster, M., Gross, M.: Modelling and Optimizing Mathematics Learning in Children. IJAIED (2013)
12. Käser, T., Schwing, A.G., Hazan, T., Gross, M.: Computational Education using Latent Structured Prediction. To appear in Proc. AISTATS (2014)
13. Käser, T., Busetto, A.G., Baschera, G.-M., Kohn, J., Kucian, K., von Aster, M., Gross, M.: Modelling and optimizing the process of learning mathematics. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 389–398. Springer, Heidelberg (2012)
14. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. IJAIED (1997)

15. Koedinger, K., Baker, R., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J.: A Data Repository for the EDM community: The PSLC DataShop. In: Handbook of Educational Data Mining. CRC Press, Boca Raton (2010)
16. Lafferty, J., McCallum, A., Pereira, F.C.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. ICML (2001)
17. Mayo, M., Mitrovic, A.: Optimising its behaviour with bayesian networks and decision theory. IJAIED (2001)
18. Pardos, Z.A., Heffernan, N.T.: Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 255–266. Springer, Heidelberg (2010)
19. Pardos, Z.A., Trivedi, S., Heffernan, N.T., Sárközy, G.N.: Clustered knowledge tracing. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 405–410. Springer, Heidelberg (2012)
20. Pavlik, P.I., Cen, H., Koedinger, K.R.: Performance Factors Analysis - A New Alternative to Knowledge Tracing. In: Proc. AIED (2009)
21. Reye, J.: Student Modelling Based on Belief Networks. IJAIED (2004)
22. Schwing, A.G., Hazan, T., Pollefeys, M., Urtasun, R.: Efficient Structured Prediction with Latent Variables for General Graphical Models. In: Proc. ICML (2012)
23. Wang, Y., Beck, J.: Class vs. Student in a Bayesian Network Student Model. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 151–160. Springer, Heidelberg (2013)
24. Wang, Y., Heffernan, N.T.: The student skill model. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 399–404. Springer, Heidelberg (2012)
25. Yudelson, M.V., Koedinger, K.R., Gordon, G.J.: Individualized Bayesian Knowledge Tracing Models. In: Proc. AIED (2013)

Identifying Effective Moves in Tutoring: On the Refinement of Dialogue Act Annotation Schemes

Alexandria Katarina Vail and Kristy Elizabeth Boyer

Department of Computer Science
North Carolina State University
Raleigh, North Carolina, USA
{akvail,keboyer}@ncsu.edu

Abstract. The rich natural language dialogue that is exchanged between tutors and students has inspired many successful lines of research on tutorial dialogue systems. Yet, today's tutorial dialogue systems do not regularly achieve the same level of student learning gain as has been observed with expert human tutors. Implementing models directly informed by, and even machine-learned from, human-human tutorial dialogue is highly promising. With this goal in mind, this paper makes two contributions to tutorial dialogue systems research. First, it presents a dialogue act annotation scheme that is designed specifically to address a common weakness within dialogue act tag sets, namely, their dominance by a single large majority dialogue act class. Second, using this new fine-grained annotation scheme, the paper describes important correlations uncovered between tutor dialogue acts and student learning gain within a corpus of tutorial dialogue for introductory computer science. These findings can inform the design of future tutorial dialogue systems by suggesting ways in which systems can adapt at a fine-grained level to student actions.

1 Introduction

It has been widely demonstrated that one-on-one tutoring is more effective than many other forms of instruction [1, 2]. This success is thought to be largely a result of the rich natural language interaction between student and tutor [3–5]. Human tutorial dialogue has therefore been studied extensively, and the strategies observed with human tutors have inspired a number of successful tutorial dialogue systems (e.g., [6–10]). However, despite the rapid progress achieved in modern tutorial dialogue systems, systems do not yet match the effectiveness of expert human tutors [1]. A promising direction for further improving tutorial dialogue systems is to identify direct associations between measured student learning gain and tutorial strategies [6, 10–12].

Tutorial strategies are realized at the level of *dialogue acts*, which characterize the intent of dialogue utterances. This paper explores the dialogue acts that human tutors make and identifies relationships between particular dialogue events

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 199–209, 2014.

and student learning gain. The analyses were conducted on a corpus of human-human textual dialogue collected through a tutoring interface for introductory computer science. This study is part of the larger JavaTutor project that is developing an intelligent tutoring system whose behaviors are machine-learned from corpora of human-human tutoring. This paper makes two novel contributions. First, it presents a tutorial dialogue act annotation scheme that addresses an important weakness of prior annotation schemes applied in numerous tutorial dialogue domains: the presence of a large majority class dialogue act that is more vague than other acts and that presents challenges for machine-learning models. Second, this paper utilizes a corpus manually tagged with this refined dialogue act tag set to explore relationships between dialogue acts and student learning gain at the end of the tutoring session. The results suggest important relationships between tutor choices and student learning.

2 Related Work

It has long been recognized that one-on-one tutoring is one of the more effective methods of instruction [13] and that the study of human-human tutorial interactions is crucial to the development of effective intelligent tutorial systems addressing this need [5]. Several dialogue acts have been previously identified as significantly correlated with learning gain [10]; in particular, specific collaborative acts between tutor and student have been studied and established as influential [14]. Historically, it was often assumed that the most frequent human tutorial acts are the effective tutorial acts, since human tutors are considered to be generally effective [11]. This might not be the best approach, as effective tutorial strategies vary from student to student and tutor to tutor [10].

Moving beyond this pure frequency approach, dialogue has been demonstrated to correlate with learning gain in a variety of ways: particular dialogue act sequence occurrences [10], adaptation to dialogue structure correlated with positive learning gain [6], and responsiveness to student uncertainty [12]. However, a frequent limitation in capturing dialogue acts for tutoring and across a variety of dialogue domains lies with crafting the annotation scheme, where it is often discovered after annotation that one dialogue act encompasses a larger portion of the corpus than any other act. For example, the Inform tag comprises 29% of an airline reservation human-human dialogue corpus [15], and the Non-Substantive Act tag, defined to be any act that was not a question, feedback, or answer, comprises 46% of an ITSPOKE physics tutorial dialogue corpus [16].

This paper expands upon prior work by defining a novel annotation scheme derived from a variety of prior schemes. With this refined annotation scheme, the analysis produced new statistical relationships not previously identified between student learning gain and the dialogue events of the tutoring sessions.

3 Tutorial Dialogue Corpus

The corpus examined here consists of computer-mediated textual human-human interactions. The sessions were conducted within an online remote tutoring

interface for Java programming. The interface, displayed in Figure 1, consists of four panes: the task description, the compilation and execution output, the student's Java source code, and the textual dialogue messages between the tutor and the student. The content of the interface was synchronized in real time between the student and the tutor; however, the tutor's interactions with the environment were constrained to the textual dialogue with the student and the ability to progress between tasks.

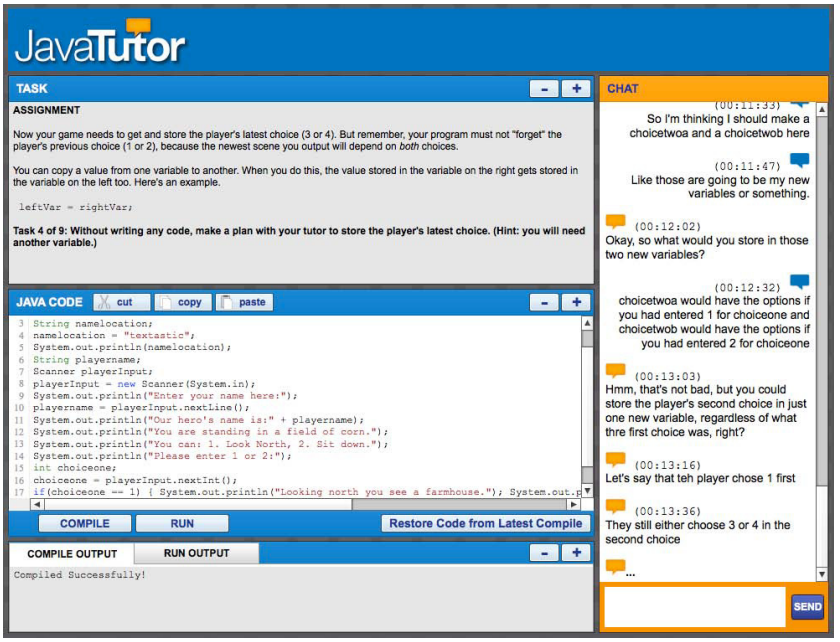


Fig. 1. The tutorial dialogue interface

The full tutorial dialogue corpus under consideration was collected in Fall 2011 and Spring 2012. Due to the time requirements of manual annotation, the current analysis examines a subset of the full corpus, sessions between 30 students paired with one of five tutors for the first of six sequential lessons [17]. This 30-session corpus consists of 4,035 utterances: 2,846 (71%) tutor utterances and 1,189 (29%) student utterances. The average number of utterances per tutorial session is 134.5 (min = 69, max = 213); tutors averaged 94.9 utterances per session (min = 46, max = 159), and students 39.6 utterances per session (min = 21, max = 65). Many utterances contained multiple dialogue acts; to address this concern, the utterances were manually partitioned at sentential and phrasal boundaries by the principal dialogue act annotator. Two sample excerpts from the corpus after annotation are displayed in Figure 2. (The annotation scheme is described in Section 4.)

To measure learning gain over the course of the session, students completed an identical pretest and posttest for each lesson. The average pretest score was 50.98% (min = 23.53%, max = 100%), and the average posttest score was 76.67%

TUTOR Let's move on. [D]

Tutor advances to next task.

TUTOR We have plenty of time. [R]

STUDENT Okay. [ACK]

Student edits code.

STUDENT Which do I put first? [QD]

TUTOR Try it. [D]

TUTOR Be sure you are satisfying the task. [D]

Student compiles, with errors.

TUTOR What you had was close. [FOE]

STUDENT Why did I need quotes for the Hello World println(), but not this one? [QI]

TUTOR Hello World was printing literal "hello world". [AWH]

TUTOR The second was printing the value inside the variable DylansCompGame. [AWH]

STUDENT Oh, alright. [ACK]

STUDENT Makes sense. [FU]

Fig. 2. Sample annotated excerpts from the Lesson 1 corpus

(min = 41.18%, max = 100%), administered immediately after completing the lesson. This learning gain (*posttest* − *pretest*) was statistically significant ($p < 0.0001$). In addition to the pretest, the students also completed a self-efficacy survey with six Likert-scale items prior to the initial tutorial session [17]. Each student's computer science self-efficacy was computed as the average of these six items. The mean self-efficacy score among the students was 3.39 out of a possible 5.00 (min = 2.33, max = 4.33), and as described later, this score is used in the current analysis along with pretest score as control variables within the predictive models of learning.

4 Dialogue Act Annotation

The new refined dialogue act annotation protocol expanded upon a prior scheme for task-oriented tutorial dialogue [17] and was further inspired by previous annotation schemes for tutorial dialogue in several domains [16, 18, 19]. The annotation scheme is presented in detail in Tables 1 and 2, along with the relative frequency of the individual tags and the Cohen's kappa achieved between two independent human annotators. Table 1 displays dialogue acts assigned to both tutor and student utterances; Table 2 displays those assigned to only tutor or only student utterances.

The present dialogue act annotation scheme expands upon a prior set, with a primary goal of further defining the vague large classes previously observed. Figure 3 displays the decomposition of the prior tagset into the current one. The previous set contained thirteen dialogue act tags, with the largest tag accounting for 33.5% of the corpus [17]. The refined annotation scheme presented here contains 31 dialogue act tags, with the largest tag accounting for 13.66% of the corpus. Despite the increased complexity of the proposed annotation scheme, two independent human annotators achieved a Cohen's kappa of $\kappa = 0.87$ on 37% of the corpus (agreement of 89.6%). The prior simpler annotation scheme yielded a Cohen's kappa of $\kappa = 0.79$ (agreement of 81.1%). Of the 31 tags, 21

Table 1. Dialogue act tags assigned to both tutor and student

Tag	Example	Freq.	κ
Explanation (**E**)	*Your code stops on line 2.*	13.66%	0.716
Greeting (**GRE**)	*Have a good day!*	3.49%	0.931
Acknowledge (**ACK**)	*Okay.*	6.93%	0.960
Correction (**CO**)	**explanation*	0.61%	0.734
Observation (**O**)	*See, we have an error.*	1.87%	0.582
Extra Domain Question (**QEX**)	*How are you today?*	1.11%	1.000
Extra Domain Answer (**AEX**)	*I'm doing well.*	1.11%	0.916
Extra Domain Other (**OEX**)	*Calculus is difficult.*	3.59%	0.797
Yes/No Answer (**AYN**)	*No, sir.*	4.20%	0.973
WH-Question Answer (**AWH**)	*Line 9.*	2.68%	0.816

tags achieved a kappa that is characterized as 'almost perfect' inter-rater reliability [20], and the excellent overall kappa achieved by the new tag set suggests that it reliably captures important differences in dialogue acts within the tutorial dialogue corpus. In addition to the tutorial dialogue acts, student task actions were annotated automatically using an edit distance approach. Each period of student coding was classified as improved, worsened, or unchanged, depending on the change in edit distance [17].

5 Relationships between Dialogue and Student Learning

The objective of the present analysis is to identify tutor dialogue act choices correlated with student learning gain. Dialogue acts were identified at the unigram (individual dialogue acts) and bigram (pairs of adjacent dialogue acts) levels [16]. Bigrams were extracted using a three-act collocational window, as demonstrated in Figure 4.

Utterances annotated with the Correction (CO) tag were removed prior to analysis, as these utterances constitute artifacts of the 'instant-messaging' nature of the corpus and reflect typing skill rather than tutoring content. Then, the relative frequencies of each dialogue act tag or bigram were computed, and simple linear correlations were calculated between these and student learning. Then, any correlations that appeared statistically significant at the $p < 0.05$ level were provided as input to a stepwise linear regression model within the SAS statistical modeling software, alongside the pretest and self-efficacy scores. Providing the pretest and self-efficacy as predictors allows the model to account for any differences in posttest scores explainable by these variables.

Several individual dialogue acts or bigrams were significantly predictive of student learning gain. These predictors and their regression coefficients, along with associated p-values within the stepwise linear regression, are listed in Table 3.

Table 2. Dialogue act tags only assigned to one role

Tag	Example	Freq.	κ
	TUTOR		
DIRECTIVE (**D**)	*Test your program.*	9.26%	0.960
INFORMATION (**I**)	*Variable names must be one word.*	7.64%	0.734
REASSURANCE (**R**)	*We have plenty of time left.*	1.01%	0.748
READY QUESTION (**QR**)	*Ready to move on?*	8.65%	1.000
QUESTIONS (**QQ**)	*Any questions?*	1.32%	0.972
FACTUAL QUESTION (**QF**)	*What line is it waiting on?*	1.21%	0.831
OPEN QUESTION (**QO**)	*How can you fix it?*	0.66%	1.000
EVALUATIVE QUESTION (**QE**)	*Does that make sense?*	0.76%	0.933
PROBING QUESTION (**QP**)	*Do you think that looks correct?*	0.40%	0.712
POSITIVE FEEDBACK (**FP**)	*Very good!*	10.72%	0.948
POSITIVE FEEDBACK (WITH ELABORATION) (**FPE**)	*That's a very good approach.*	1.97%	0.729
NEGATIVE FEEDBACK (**FN**)	*No, that's incorrect.*	0.05%	1.000
NEGATIVE FEEDBACK (WITH ELABORATION) (**FNE**)	*That's not the right syntax.*	0.25%	1.000
OTHER FEEDBACK (**FO**)	*That's an okay implementation.*	0.25%	0.800
OTHER FEEDBACK (WITH ELABORATION) (**FOE**)	*That's alright, but you need to fix line 9.*	0.61%	0.952
	STUDENT		
INFORMATION QUESTION (**QI**)	*Why does that happen?*	1.77%	0.917
CONFIRMATION QUESTION (**QC**)	*It's line 6, right?*	2.18%	0.895
DIRECTION QUESTION (**QD**)	*What do I do next?*	1.32%	1.000
READY ANSWER (**AR**)	*Yes, I'm ready.*	8.14%	0.952
UNDERSTANDING (**FU**)	*Oh, that makes sense!*	1.87%	0.847
NOT UNDERSTANDING (**FNU**)	*I don't know why that works...*	0.71%	0.665

The unigram occurrence of tutor directives were negatively correlated with learning gain, aslseen in previous studies [16, 17]. Interestingly, this was the only unigram significantly correlated with learning gain at the $p < 0.05$ level. Two other previously-identified tutorial decisions also emerged as significant: consecutive tutor directives, as seen in a previous study on the same corpus [17] and a tutor information move following a student answer, as seen in the ITSPOKE dialogue corpus [16].

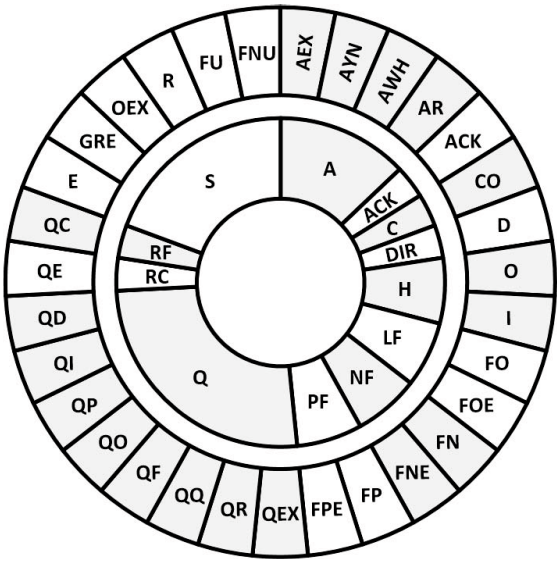


Fig. 3. Decomposition of the prior tags (inner ring) into the new tags (outer ring)

Table 3. A selection of tutor dialogue act choices significantly predictive of student learning gain

Weight	Dialogue Act and Task Sequence	Partial R^2	p
+0.3345	pretest	0.1785	< 0.0001
+0.0868	self-efficacy	0.0003	0.2156
−0.4995	(Improved Code → D (Tutor))	0.0047	0.0050
−0.5004	(FNU (Student) → E (Tutor))	0.0369	0.0049
+0.4388	(QC (Student) → PF (Tutor))	0.0265	0.0153
−0.5781	(D (Tutor))	0.2587	0.0008
−0.5758	(D (Tutor) → D (Tutor))	0.0216	0.0009
−0.4748	(E (Tutor) → QE (Tutor))	0.0231	0.0080
−0.3904	(I (Tutor) → O (Tutor))	0.1474	0.0329
+0.3784	(AWH (Student) → I (Tutor))	0.0907	0.0392
+0.1458	(Intercept)		0.0212

There were several dialogue bigrams significantly correlated with learning gain that had not been identified with a coarser annotation scheme. The bigrams, as shown in Table 3, include improved code followed by tutor directive (D), a student expression of not understanding (FNU) to a tutor explanation (E), a student confirmation question (QC) to positive feedback (PF), a tutor explanation (E) followed by an evaluative question (QE), and a tutor instruction (I) followed by an observation (O). These significant relationships are discussed in the next section.

Role	Utterance	Extracted Bigrams		
Tutor	Do you have any questions? [QQ]			
Tutor	Look over your program. [D]	QQ → D		
Student	No [AYN]	D → AYN	QQ → AYN	
Student	I believe I am understanding the concept. [FU]	AYN → FU	D → FU	QQ → FU

Fig. 4. An example of the collocational window employed to capture dialogue act bigrams at a distance

6 Discussion

This section examines the tutorial dialogue events that were found to be significantly associated with student learning. First we examine tutor directives (D (Tutor)), which are indications that the tutor is giving explicit direction to the student. Consecutive instructions of this nature (D (Tutor) → D (Tutor)) could indicate that the tutor is choosing to exert substantial control over the tutorial session, or that the student is relying heavily on tutor instructions [16, 17]. Another relationship that has been observed in other literature relates to the bigram of a tutor offering information (I) following a student response to a question (AWH), which could indicate a tutor elaborating upon the student's response beyond what he or she initially understood to be correct. This can sometimes provide the answer that the tutor originally expected of the student. This bigram has been previously identified as significant to student learning gain in tutoring for physics [16].

One interesting relationship occurs when the tutor decides to offer a directive after the student has improved the Java program (Improved Code → D (Tutor)). This tutor dialogue act is negatively predictive of learning gain. This relationship could be due to a tutor incorrectly believing that the student needs guidance, and taking control of the session before it is necessary. The directives in the current corpus were frequently an instruction to compile or run the program. This could also occur due the enforced time limit on the session; if the tutor does not believe that the student will complete the lesson before the end of the session, he may give more direct instructions to hasten the completion of the tasks.

A tutor observation after a tutor information turn (I (Tutor) → O (Tutor)) is also negatively predictive of learning gain. This bigram could potentially indicate a "lecturing" mode by the tutor, whereas leaving these tasks to the student to discover could be beneficial to her overall understanding.

Another negative association with learning emerges when tutor evaluative questions, such as *"Does that make sense?"*, follow an explanation (E (Tutor) → QE (Tutor)). One suggested interpretation of this phenomenon is that new students lack meta-cognition; that is, students may not truly know if the material 'makes sense' yet. This is possibly a novice tutor move, as experienced tutors

tend to ask more open-ended questions, judging a student's understanding by his or her demonstrated ability to use the material, rather than relying on the student's meta-cognitive abilities.

Another bigram that was negatively correlated with student learning gain was a tutor offering an explanation when the student expresses a lack of understanding (FNU (Student) → E (Tutor)). This could be explained by a tutor instinctively offering the solution to the student, instead of allowing an exploratory approach by the student before giving aid.

The only bigram found to be significantly positively correlated with learning gain was positive feedback after a confirmation question from the student (QC (Student) → PF (Tutor)). Often, interchanges with these annotations were of the form *"I think the answer is X?"*, followed by a *"Yes, very good!"*. The decision to actively support a student's uncertain answer may provide the student some level of confidence in his ability, which can positively impact further work in the session.

7 Conclusion and Future Work

Tutorial dialogue is rich and highly effective, yet the mechanisms responsible for its effectiveness are not fully understood. Identifying tutor dialogue acts that are associated with student learning gain is a promising direction for research. This paper has presented a novel dialogue act annotation scheme designed to substantially reduce the dominance of a vague majority class that has existed in many prior annotation schemes. When applied in a regression analysis to predict student learning, this new annotation scheme demonstrated its use in identifying previously undiscovered specific dialogue interactions that are predictive of outcomes.

Compelling directions for future work include identifying and comparing effective tutor choices across differing student types, e.g. low versus high self-efficacy students, or students entering from a variety of disciplines. Additionally, a crucial direction for the field is to examine how our annotation schemes support machine learning and data mining on corpora of tutorial dialogue in ways that can inform the design of or support the direct extraction of effective tutorial dialogue system behaviors. These lines of investigation will lead to greater understanding of student learning through tutoring and will inform the design of tutorial dialogue systems.

Acknowledgements. The authors wish to thank Joseph Wiggins for his contributions to the annotation stage of this study, and the members of the Center for Educational Informatics at North Carolina State University for their helpful input. This work is supported in part by the Department of Computer Science at North Carolina State University and the National Science Foundation through Grant DRL-1007962 and the STARS Alliance, CNS-1042468. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the authors, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

References

1. VanLehn, K., et al.: When Are Tutorial Dialogues More Effective Than Reading? Cog. Sci. 31(1), 3–62 (2007)
2. Bloom, B.S.: The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. Educ. Res. 13(6), 4–16 (1984)
3. Chi, M.T., et al.: Learning from human tutoring. Cog. Sci. 25(4), 471–533 (2001)
4. Lepper, M.R., et al.: Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. Computers as Cognitive Tools 1993, 75–105 (1999)
5. Graesser, A.C., et al.: Collaborative dialogue patterns in naturalistic one-to-one tutoring. Applied Cog. Psy. 9(6), 495–522 (1995)
6. Chi, M., VanLehn, K., Litman, D.: Do Micro-Level Tutorial Decisions Matter: Applying Reinforcement Learning to Induce Pedagogical Tutorial Tactics. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 224–234. Springer, Heidelberg (2010)
7. Dzikovska, M.O., Steinhauser, N.B., Moore, J.D., Campbell, G.E., Harrison, K.M., Taylor, L.S.: Content, social, and metacognitive statements: An empirical study comparing human-human and human-computer tutorial dialogue. In: Wolpers, M., Kirschner, P.A., Scheffel, M., Lindstaedt, S., Dimitrova, V. (eds.) EC-TEL 2010. LNCS, vol. 6383, pp. 93–108. Springer, Heidelberg (2010)
8. Kumar, R., Ai, H., Beuth, J.L., Rosé, C.P.: Socially Capable Conversational Tutors Can Be Effective in Collaborative Learning Situations. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 156–164. Springer, Heidelberg (2010)
9. D'Mello, S.K., et al.: A Motivationally Supportive Affect-Sensitive AutoTutor. New Perspectives on Affect and Learning Tech. 3, 113–126 (2011)
10. Chen, L., et al.: Exploring Effective Dialogue Act Sequences in One-on-one Computer Science Tutoring Dialogues. In: Tetreault, J., et al. (eds.) Proc. 6th BEA Work., Portland, USA, pp. 65–75. Assoc. for Comp. Ling (2011)
11. Stellan, Ohlsson, o.: Beyond the Code-and-count Analysis of Tutoring Dialogues. In: R, Luckin, o. (eds.) Proc. 13th Int. Conf. AIED, Los Angeles, USA, vol. 158, pp. 349–356. IOS (2007)
12. Forbes-Riley, K., Litman, D.J.: Adapting to Student Uncertainty Improves Tutoring Dialogues. In: Vania, Dimitrova, o. (eds.) Proc. 14th Int. Conf. AIED, Brighton, United Kingdom, pp. 33–40. IOS (2009)
13. Cohen, P.A., et al.: Educational Outcomes of Tutoring: A Meta-analysis of Findings. Am. Educ. Res. J. 19(2), 237–248 (1982)
14. D'Mello, S.K., et al.: Mining Collaborative Patterns in Tutorial Dialogues. J. EDM 2(1), 1–37 (2010)
15. Chu-Carroll, J.: A Statistical Model for Discourse Act Recognition in Dialogue Interactions. In: Chu-Carroll, J., Green, N. (eds.) AAAI Spring Symp.: Applying Machine Learning to Discourse Processing, Pan Alto, USA, vol. 1996, pp. 12–17. AAAI Press (1998)
16. Litman, D.J., Forbes-Riley, K.: Correlations between dialogue acts and learning in spoken tutoring dialogues. Nat. Lang. Eng. 12(2), 161–176 (2006)
17. Mitchell, C.M., et al.: Recognizing Effective and Student-Adaptive Tutor Moves in Task-Oriented Tutorial Dialogue. In: Youngblood, M.G., McCarthy, P.M. (eds.) Proc. 25th Int. FLAIRS Conf., Marco Island, Florida, pp. 450–455. AAAI Press (2009)

18. Person, N.K., et al.: The Dialog Advancer Network: A Conversation Manager for AutoTutor. In: Gauthier, G., et al. (eds.) Proc. ITS Work. Modeling Human Teaching Tactics and Strategies, Montreal, Canada, pp. 86–92. Springer (2000)
19. Core, M.G., Allen, J.F.: Coding Dialogs with the DAMSL Annotation Scheme. In: Proc. 1997 AAAI Fall Symp.: Communicative Action in Humans and Machines, Providence, USA, pp. 28–35. AAAI (1997)
20. Landis, J.R., Koch, G.G.: The Measurement of Observer Agreement for Categorical Data Data for Categorical of Observer Agreement The Measurement. Biometrics 33(1), 159–174 (1977)

When Is Tutorial Dialogue More Effective Than Step-Based Tutoring?

Min Chi[1], Pamela Jordan[2], and Kurt VanLehn[3]

[1] Computer Science Department, North Carolina State University,Raleigh NC USA
mchi@ncsu.edu
[2] Learning Research and Development Center, University of Pittsburgh, Pittsburgh, PA USA
pjordan@pitt.edu
[3] School of Computing, Informatics and Decision Science Engineering, Arizona State University, AZ USA
Kurt.Vanlehn@asu.edu

Abstract. It is often assumed that one-on-one dialogue with a tutor, which involves micro-steps, is more effective than conventional step-based tutoring. Although earlier research often has not supported this hypothesis, it may be because tutors often are not good at making micro-step decisions. In this paper, we compare a micro-step based NL-tutoring system that employs induced pedagogical policies, Cordillera, to a well-evaluated step-based ITS, Andes. Our overall conclusion is that the pairing of effective policies with a micro-step based system does significantly outperform a step-based system; however, there is no significant difference in the absence of effective policies. Moreover, while micro-step tutoring is more time-consuming, the findings still hold for five out of six learning performance measures when time on task is factored out.

Keywords: Natural Language tutoring systems, Step-based tutoring systems, Reinforcement Learning, Pedagogical Strategy.

1 Introduction

In domains like math and science solving a problem requires producing an argument, proof or derivation consisting of one or more inference steps, and each step is the result of applying a domain principle, operator or rule. For instance, solving an algebraic equation, 2x+5=21 can be done via two steps: 1) subtract the same term 5 from both sides of the equation; and 2) divide both sides by 2.

A typical Intelligent Tutoring System (ITS) is step-based [13] while human tutors or Natural Language (NL) tutoring systems, by contrast, often scaffold students via a series of micro-steps leading to the full step. A step corresponds to (writing) actions that are ordinarily done even without the tutor, whereas micro-steps correspond roughly to dialogue acts in tutorial dialogues. For example, in order to solve a physics problem, the student needs to apply several domain principles, some of which may need to be applied multiple times. Each principle

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 210–219, 2014.

1. **T:** So let's start with determining the value of KE_0.
2. **T:** Which principle will help you calculate the rock's kinetic energy at T0? Please provide the name of the principle, not an equation. **{ELICIT}**
3. **S:** Definition of kinetic energy
4. **T:** Yes, I agree. Now I will write the equation for applying the definition of kinetic energy to the rock at T0: KE0 = 1/2*m*v0^2 **{TELL}**

Fig. 1. Illustrations of Two Micro-Steps

application can be seen as a step in the ITS. In a physics tutor, for example, applying the definition of Kinetic Energy ($KE = \frac{1}{2}mv^2$) to solve for the kinetic energy of a falling rock at T_0 is a step; once a student enters a step, then the tutor gives feedback and/or hints. Human tutors, by contrast, often scaffold students via a series of micro-steps leading to the full step. In the step mentioned above, for instance, a human tutor can take the following **micro-level** steps: selecting the principle to apply; writing the corresponding equation; solving the equation; and engaging in some qualitative discussion about the principle.

Fig. 1 illustrates two micro-steps and each numbered line represents a dialogue turn. The labels **T** and **S** designate tutor and student turns respectively. In this example, the tutor and the student first select a principle (lines 2 & 3) and then write the corresponding equation (line 4). Some of the tutor turns in Fig. 1 are labeled **{ELICIT}** or **{TELL}**. This label designates a ***tutorial decision step*** wherein the tutor has to make a tutorial decision whether to ask the student for the requisite information or to tell it to the student. For example, in line 2, the tutor chooses to *elicit* the answer by asking, "Which principle will help you calculate the rock's kinetic energy at T0? Please provide the name of the principle, not an equation." If the tutor elects to tell, however, then he or she would state, "To calculate the rock's kinetic energy at T0, let's apply the definition of Kinetic Energy."

One common hypothesis as to the effectiveness of human one-on-one tutoring comes from the detailed management of "micro-steps" in tutorial dialogue[6,7] and thus suggests that micro-step based tutors are more effective than step-based tutors. In several tests of this hypothesis, however, neither human tutors nor NL tutors designed to mimic human tutors, outperformed step-based tutors once content was controlled to be the same across all conditions [5,12]. All three types of tutors were more effective than no instruction (e.g., students reading material and/or solving problems without feedback or hints). One possible conclusion is that tutoring is effective, but the micro-steps of human tutors and NL tutoring systems provide no additional value beyond conventional step-based tutors[13].

Alternatively, we argue that the lack of difference between micro-step and step-based tutors is because neither the human tutors nor the NL tutoring systems involved in those studies were good at making micro-step decisions and several studies provide some support for this claim[3,11,2]. Previously, we investigated the impact of pedagogical policies on student learning by comparing different versions of a micro-step based NL tutoring system called Cordillera [2].

We applied a general data-driven methodology, Reinforcement Learning (RL), to induce pedagogical policies directly from student interactivity logs and found that Cordillera with effective pedagogical policies, RL-induced Cordillera significantly out-performed other versions of Cordillera. However, it is still unclear whether the former is significantly better than a step-based ITS.

In this paper, we directly compare RL-induced Cordillera with a well-evaluated step-based conventional ITS, Andes [14]. Our main research question is: *Can a NL tutoring system with machine-learned pedagogical policies be more effective than a step-based ITS?* Overall, we find that RL-induced Cordillera significantly outperforms Andes. In order to investigate whether this result is indeed caused by effective RL-induced policies, we also compare Andes to two other versions of Cordillera: Hybrid-RL and Random. In the following, we will briefly describe the two types of tutoring systems and the pedagogical policies employed in them and then describe our study and finally present our results.

2 Two Types of ITSs

The Micro-Step Based Cordillera: NL Tutorial Dialogue System
The Cordillera tutorial dialogue system tutors students in both quantitative and qualitative physics in the work-energy domain and was implemented using the TuTalk tutorial dialogue system toolkit toolkit [8]. TuTalk supports dialogues in which a tutor tries to elicit the main line of reasoning from a student by a series of coherent questions. This style of dialogue was inspired by CIRCSIM-Tutor's directed lines of reasoning [5]. The Cordillera style of dialogue is system-initiative in that the system always chooses the topics discussed.

Figure 2 illustrates a sample student dialogue with Cordillera. The upper top right pane of the figure shows the problem that the student is attempting to

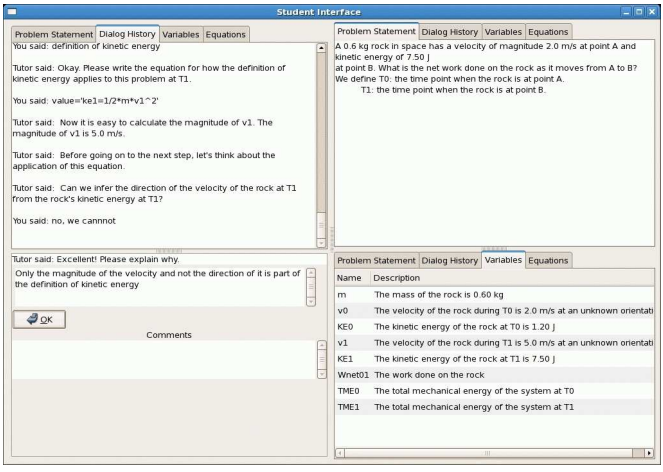


Fig. 2. An example of the Cordillera interface

solve. The top left pane shows a portion of the dialogue history, and illustrates a few questions and student responses, as well as a number of system informs; the pending tutor question is shown in the input pane at the bottom followed by the response the student is entering. Finally, the variables in the bottom right pane and the equations (hidden) were entered either by the student using a form interface (not shown) or provided by the tutor. When the tutor asks the student to compute the value for a variable, the student must transform the equation to a solvable form with the known values substituted and then the tutor will do the final calculation. In order to avoid confounds due to imperfect NL understanding, a human wizard replaced the NL understanding module. During tutoring, the wizard matched students' answers to one of the available responses but made no tutorial decisions.

The Step-Based Andes Tutoring System

Andes provides a multi-paned screen that consists of a problem-statement window, a variable window, an equation window, and a dialogue window. An example of the Andes interface, as the student would see it, is shown in Figure 3. On Andes, students construct and manipulate a solution. The interaction is open-ended, event-driven and student-initiated. Students can enter an equation that is the algebraic combination of several principle applications and Andes provides immediate feedback on each entry. Andes can also algebraically manipulate equations to calculate the value for a variable. It considers an entry correct if it is true, regardless of whether it is useful for solving the problem. When an entry is incorrect, students can either fix it independently, or ask for what's-wrong help. When they do not know what to do next, they can ask for next-step help. Both next-step and what's-wrong helps are provided via a sequence of hints

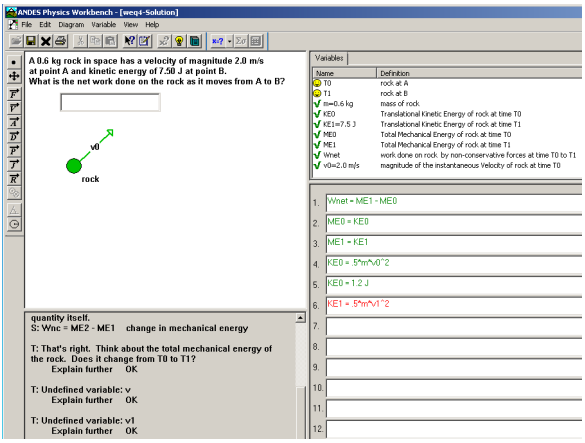


Fig. 3. An example of the Andes interface

that gradually increase in specificity. The last hint in the sequence, called the bottom-out hint, tells the student exactly what to do.

Andes provides conceptual and procedural help that is designed to encourage students to think on their own. Students can always enter any correct step and Andes does not attempt to determine their problem-solving plans. If necessary for giving a hint, it asks students what principle they are working on. If students indicate a principle that is part of a solution to the problem, Andes hints an uncompleted step from the principle application. If no acceptable principle is chosen, Andes picks an unapplied principle from the solution that they are most likely to be working on.

3 Decision Policies within Cordillera and Andes

In many tutoring systems, the system's behaviors can be viewed as a sequential decision process wherein, at each discrete step, the system is responsible for selecting the next action to take. Pedagogical strategies are defined as policies to decide the next system action when multiple are available. Each of these system decisions affects the user's successive actions and performance. Its impact on student learning cannot often be observed immediately and the effectiveness of one decision also depends on the effectiveness of subsequent decisions. Ideally, an effective tutor should craft and adapt its decisions to users' needs [1,10]. However, there is no existing well-established theory on how to make these system decisions effectively. In this work, different versions of micro-step based Cordillera employed different pedagogical policies. The step-based Andes employs hand-coded rules.

Three versions of Cordillera - Random, Hybrid-RL, and RL-induced - were involved. The only difference among the three is the policy used. Random Cordillera made tutorial decisions randomly. Hybrid-RL Cordillera used expert-guided data-driven induced rules. These rules were induced by using 18 features and a greedy-like procedure to prune the features to meet efficiency and training constraints[4]. Both the initial features and pruning procedure were suggested by human experts and the final induced policies were also checked and approved by human experts. But no significant difference was found on overall learning performance between the Hybrid-RL and random policies. For RL-induced Cordillera, the data-driven approach was greatly improved. More specifically, the RL approach involved a much larger feature set (50 features), and more advanced domain-general feature selection approaches. Human experts were not involved in directing the policy generation. As reported earlier[2], these RL-induced policies indeed helped students learn more and in a deeper way than either Hybrid-RL or random policies.

Andes, on the other hand, like most existing ITSs employs hand-coded pedagogical policies. For example, help in Andes is provided upon request because it is assumed that students know when they need help and will only process help when they desire it. A student deciding to request help can be seen as a human-like decision policy for whether to skip or not skip content.

4 Methods

Participants: A total of 163 participants used either Andes or one of the three versions of Cordillera: the Andes group comprised 33 students; the Random Cordillera Group comprised 64 students so that we could collect enough data for RL policy induction; the Hybrid-RL Cordillera Group comprised 37 students; and the RL-induced Cordillera group comprised 29 students. All participants were recruited in the same way but in different years.

Domain and Procedure: The training covered the first-year college physics work-energy domain. All participants experienced identical procedures: 1) a background survey; 2) read a textbook covering the target domain knowledge; 3) took a pretest; 4) solved the same seven training problems in the same order on either Andes or Cordillera; and 5) finally took a posttest. The pretest and posttest were identical and contained 16 quantitative items and 16 qualitative items. Both quantitative and qualitative items include multiple choice and open-ended problems.

Students' learning outcomes were measured by using three types of scores: quantitative, qualitative and overall. All tests were graded in a double-blind manner by experienced graders. In a double-blind manner, neither the students nor the graders know who belongs to which group. For comparison purpose all test scores were normalized to fall in the range of [0,1].

Except for following the policies (Random, Hybrid-RL, or RL-induced), the remaining components of Cordillera, including the interface, the training problems, and the tutorial scripts, were identical for all students. However, there are some noticeable differences for the Andes training compared to Cordillera.

Differences in the Training: The Cordillera dialogues guided students through the training problems by hinting at the next problem solving step to be completed, or telling them what it is. Hints took the form of short answer questions. In addition to guiding the student through problem solving, Cordillera also attempted to help the student increase his/her conceptual understanding of the domain by asking for justifications for the most important problem solving steps. The decision for when to ask for a justification was determined by a set of pedagogical policies. For an example of a justification requested during problem solving, see the current tutor turn in the bottom left input pane in Figure 2. There was also a post-problem discussion for each problem which sought to increase the student's conceptual qualitative understanding.

We implemented the same seven training problems in Andes and because Cordillera provided drawings and pre-defined some variables for each problem, we set-up Andes to provide the same. We added a post-problem discussion to Andes by collecting all the post-problem discussion for Cordillera into a *static* text document so that the content coverage for post-problem discussion was about the same. The post-problem discussion was delivered in a series of web pages after the experimenter verified that the student had completed the Andes problem.

Note that we did not attempt to provide identical content for the problem solving help since it reflects two different tutoring systems, but what is available is similar. For example, while the Cordillera system's micro-steps will always present the content illustrated in Fig. 1, Andes will show the following series of hints for this same step after the student makes four consecutive help requests: 1) Why don't you continue with the solution by working on the definition of kinetic energy. 2) What is the kinetic energy of the rock at T0? 3) The kinetic energy of an object is defined as one half its mass times its velocity squared. That is, $0.5 * m * v^2$. 4) Write the equation $KE0 = 0.5 * m * v0^2$. So for this illustration asking for all hints on the Andes step is equivalent to a decision to tell for all the related micro-steps in Cordillera.

While the problem solving help content is similar, there is also some conceptual qualitative discussion during Cordillera's problem solving that Andes does not offer. It is up to the student to consider the concepts involved on their own. However, as has been pointed out, novice students have a tendency to simply manipulate equations to isolate the unknown and seldom consider the conceptual knowledge involved during problem solving [9].

5 Results

Overall Training Time

A one-way ANOVA showed significant differences among the four groups on overall training time: $F(3, 154) = 53.90$, $p < 0.001$. The Andes group spent significantly less time[1] than the other three groups but there were no significant differences in time on task among the three Cordillera groups. The average training time (in minutes) across the seven training problems, was $M = 115.94$, $SD = 42.03$ for Andes, $M = 280.38$, $SD = 66.88$ for Random, $M = 294.33$, $SD = 87.51$ for Hybrid-RL, and $M = 259.99$, $SD = 59.22$ for RL-induced.

Learning Performance

Although students were recruited during different time periods, they appear balanced on incoming competence across the conditions. A one-way ANOVA showed that there were no significant differences in pretest scores among the four groups on either quantitative: $F(3, 159) = 1.18$, $p = .32$, or qualitative: $F(3, 159) = 0.06$, $p = .98$, or overall questions $F(3, 159) = 0.46$, $p = .71$.

A repeated measures analysis using test (pretest vs. posttest) as a factor and test score as the dependent measure showed that there was a main effect for test. All four groups of students scored significantly higher on the posttest than the pretest, $F(1, 32) = 19.87$, $p < 0.001$ for Andes, $F(1, 63) = 78.37$, $p < 0.001$ for Random, $F(1, 36) = 48.36$, $p < 0.001$ for Hybrid-RL, and $F(1, 28) = 238.58$, $p < 0.001$ for RL-induced.

The same results were found from pretest to posttest on both quantitative and qualitative questions as well. More specifically, on quantitative questions,

[1] Some reading times for the last problem were lost so we used the minimum average reading time for all other easier problems.

Table 1. RL-induced Cordillera vs. Andes on Various Test Scores

Test Item Set	Test Score	RL-induced Cordillera	Andes	Stat	cohen d
quant	Pre	0.35 (0.25)	0.28 (0.26)	$t(60) = 1.01, p = .28$	0.27
	Post	0.64 (0.22)	0.41 (0.30)	$t(60) = 3.29, p = 0.002$	$0.87**$
	Adj Post	0.61 (.18)	0.44 (.17)	$F(1,59) = 13.793, p < .0001$	$0.97**$
	NLG	0.49 (0.28)	0.16 (0.38)	$F(1,59) = 14.442, p < 0.0001$	$0.99**$
qual	Pre	0.46(0.12)	0.45(0.14)	$t(60) = 0.40, p = .688$	0.08
	Post	0.65 (0.14)	0.54 (0.18)	$t(60) = 2.68, p = 0.010$	$0.68**$
	Adj Post	0.65 (.14)	0.54 (.14)	$F(1,59) = 7.74, p = .007$	$0.79**$
	NLG	0.36 (0.24)	0.14 (0.34)	$F(1,59) = 8.86, p = 0.004$	$0.75**$
Overall	Pre	0.42 (0.15)	0.39 (0.16)	$t(60) = 0.87, p = .39$	0.19
	Post	0.65 (0.15)	0.50 (0.21)	$t(60) = 3.35, p = 0.001$	$0.82**$
	Adj Post	0.64 (.11)	0.51 (.12)	$F(1,59) = 16.50, p < .0001$	$1.13**$
	NLG	0.42 (0.19)	0.17 (0.28)	$F(1,59) = 15.97, p < 0.0001$	$1.04**$

$F(1,32) = 15.83$, $p < 0.001$ for Andes, $F(1,63) = 33.55$, $p < 0.001$ for Random, $F(1,36) = 58.01$, $p < 0.001$ for Hybrid-RL, and $F(1,28) = 95.79$, $p < 0.001$ for RL-induced. On qualitative questions, $F(1,32) = 7.68$, $p = 0.009$ for Andes, $F(1,63) = 40.62$, $p < 0.001$ for Random, $F(1,36) = 17.20$, $p < 0.001$ for Hybrid-RL, and $F(1,28) = 89.56$, $p < 0.001$ for RL-induced. Therefore all four conditions made significant gains from pre-test to post-test across all three sets of questions: quantitative, qualitative and overall questions. In order to investigate whether micro-step based tutors can be more effective than step-based tutors, we first investigated whether the most effective version of Cordillera would outperform Andes.

RL-Induced Cordillera vs. Andes

Table 1 compares the pre-test, post-test, adjusted post-test, and NLG scores between the RL-induced Cordillera and Andes conditions by question type. The adjusted Post-test scores were compared between the two conditions via an ANCOVA with the corresponding pre-test score as a covariate. NLG measures students' gain *irrespective of their incoming competence*: $NLG = \frac{posttest - pretest}{1 - pretest}$. Here 1 is the maximum score. The third and fourth columns in Table 1 list the means and SDs of the two groups' corresponding scores. The fifth column lists the statistical comparison and the last column lists the effect size of the comparison using Cohen's d[2]. Table 1 shows that there was no significant difference between the two conditions on pre-test scores. However, there were significant differences between them on the post-test, adjusted post-test, and NLG scores for all three question types.

We then compared the two groups' performance on six types of learning measures: $\{\text{Quantitative}, \text{Qualitative}, \text{Overall}\} \times \{\text{Posttest}, \text{NLG}\}$ using both pre-test and total training time as the covariates. On one measure, quantitative posttest,

[2] Which is defined as the mean learning gain of the experimental group minus the mean of the control group, divided by the groups' pooled standard deviation.

there was no significant difference between the two groups: $F(1, 58) = 2.34, p = 0.132$. But on the remaining five measures, RL-induced Cordillera significantly outperformed Andes: $F(1, 58) = 7.27, p = 0.009$ for qualitative posttest, $F(1, 58) = 5.94, p = 0.018$ for overall posttest, $F(1, 59) = 4.72, p = 0.034$ for quantitative NLG, $F(1, 59) = 7.34, p = 0.009$ for qualitative NLG and $F(1, 58) = 9.71, p = 0.003$ for overall NLG respectively.

In sum, our results showed that micro-step based tutors can indeed be more effective than step-based tutors as RL-induced Cordillera significantly outperformed Andes on all types of test questions. Even when time on task is factored out, the same results hold for five out of six learning measures. Next, we compared Random and Hybrid-RL Cordillera with Andes to investigate whether the micro-step tutor would still be more effective than the step-based tutor *without* effective pedagogical policies.

Random vs. Andes and Hybrid-RL Cordillera vs. Andes: There were no significant differences between the Random-Cordillera and Andes groups on any of the learning outcome measures. Since Andes students spent significantly less time than Cordillera students, we compared the two conditions' posttest scores using both pre-test score and total training time as covariates and their NLG scores using total training time as the covariate. To our surprise, we still found no significant differences between the two groups. We had expected the efficiency of the Andes group to have some impact.

Similar results were found when we compared Hybrid-RL Cordillera and Andes on all types of learning outcome measures either when time on task is factored in or out. Since Hybrid-RL Cordillera employed human-influenced pedagogical rules, these results again indicate that expert tutors' pedagogical rules may not always be effective. Again, this study suggests that fine-grained interactions at micro-steps are a potential source of pedagogical power, but human tutors may not be particularly skilled at choosing the right micro-steps.

6 Conclusions and Future Work

Although it is often believed that micro-step based NL tutoring systems should be more effective than conventional step-based ITSs, little evidence was previously found to support this. Our hypothesis is that it is because the existing micro-step based NL tutoring systems do not employ effective pedagogical strategies. Previous work applied a general data-driven RL approach to induce effective pedagogical policies directly from student logs and found them to be more effective than either random or Hybrid-RL policies. However, it was still not clear whether these RL-induced policies would make micro-step based NL tutoring systems more effective than step-based ITSs.

In this paper, we found that RL-induced Cordillera significantly outperforms Andes while neither Hybrid-RL Cordillera nor Random Cordillera were significantly different from step-based Andes. Our overall conclusion is that a micro-step based system with effective RL-induced policies can significantly outperform a step-based ITS with hand-coded policies; however, there is no significant difference between micro-step based and step-based tutoring systems in the absence of

effective policies. Note that micro-step based Cordillera is more time-consuming than Andes. However, even when time on task is factored out, the micro-step based tutoring system with effective RL-induced policies is still significantly better than the step-based tutoring systems with hand-coded policies on five out of six learning performance measures.

Future work that remains is to explore policy-induction for Andes and to conduct a comparison of step-based tutoring to micro-step tutoring when both have effective RL-induced pedagogical policies. This may improve our understanding of the grain-size (step vs. micro-step) issue.

Acknowledgments. This work was supported by NSF Award #0325054.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
2. Chi, M., VanLehn, K., Litman, D.J., Jordan, P.W.: Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. User Model. User-Adapt. Interact. 21(1-2), 137–180 (2011)
3. Chi, M.T.H., Siler, S., Jeong, H.: Can tutors monitor students' understanding accurately? Cognition and Instruction 22(3), 363–387 (2004)
4. Chi, M., Jordan, P.W., VanLehn, K., Litman, D.J.: To elicit or to tell: Does it matter? In: Dimitrova, V., Mizoguchi, R., du Boulay, B., Graesser, A.C. (eds.) AIED, pp. 197–204. IOS Press (2009)
5. Evens, M., Michael, J.: One-on-one Tutoring By Humans and Machines. Erlbaum, Mahwah (2006)
6. Graesser, A.C., Person, N., Magliano, J.: Collaborative dialog patterns in naturalistic one-on-one tutoring. Applied Cognitive Psychology 9, 359–387 (1995)
7. Graesser, A.C., VanLehn, K., Rosé, C.P., Jordan, P.W., Harter, D.: Intelligent tutoring systems with conversational dialogue. AI Magazine 22(4), 39–52 (2001)
8. Jordan, P.W., Hall, B., Ringenberg, M., Cui, Y., Rosé, C.: Tools for authoring a dialogue agent that participates in learning studies. In: Proceedings of AIED 2007, pp. 43–50 (2007)
9. Leonard, W., Dufresne, R., Mestre, J.: Using qualitative problem-solving strategies to highlight the role of conceptual knowledge in solving problems. American Journal of Physics 64(12) (1996)
10. Phobun, P., Vicheanpanya, J.: Adaptive intelligent tutoring systems for e-learning systems. Procedia - Social and Behavioral Sciences 2(2), 4064–4069 (2010), innovation and Creativity in Education
11. Putnam, R.T.: Structuring and adjusting content for students: A study of live and simulated tutoring of addition. Amer. Edu. Res. Journal 24(1), 13–48 (1987)
12. VanLehn, K., Graesser, A.C., Jackson, G.T., Jordan, P., Olney, A., Rosé, C.P.: When are tutorial dialogues more effective than reading? Cog. Sci. 31(1), 3–62 (2007)
13. VanLehn, K.: The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist 46(4), 197–221 (2011)
14. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The andes physics tutoring system: Lessons learned. IJAIED 15(3), 147–204 (2005)

Predicting Student Learning from Conversational Cues

David Adamson[1], Akash Bharadwaj[2], Ashudeep Singh[3], Colin Ashe[4], David Yaron[1], and Carolyn P. Rosé[1]

[1] Carnegie Mellon University, USA
[2] National Institute of Technology Karnataka, India
[3] Indian Institute of Technology Kanpur, India
[4] Indiana University of Pennsylvania, USA

Abstract. In the work here presented, we apply textual and sequential methods to assess the outcomes of an unconstrained multiparty dialogue. In the context of chat transcripts from a collaborative learning scenario, we demonstrate that while low-level textual features can indeed predict student success, models derived from sequential discourse act labels are also predictive, both on their own and as a supplement to textual feature sets. Further, we find that evidence from the initial stages of a collaborative activity is just as effective as using the whole.

Keywords: Computer-Supported Collaborative Learning, Discourse Analysis, Educational Data Mining.

1 Introduction

Intelligent tutoring and computer-supported collaborative learning can both provide cognitive, metacognitive, and social benefits to learners [13, 22, 27]. These systems also offer a wealth of process data to researchers and developers. This windfall can be used to analyze learning and other behavioral processes, and opens the door to automatic moment-to-moment formative assessment and support. The recent boom in massive and open online courses, with their similarly massive student-to-human-teacher ratios, has underlined both the need and the potential for such data-driven assessments and interventions. In this paper, we present multiple sources of predictive features from the chat transcripts of a collaborative learning scenario. As a baseline, we show that features based on the lexical and syntactic contents of student contributions in chat are predictive. We then supplement those features by paying attention to the sequence and structure of dialogue at the discourse level, and demonstrate that these features can anticipate student learning.

The remainder of this paper is organized as follows: In Section 2, we review relevant literature and establish a theoretical framework for our contribution. In Section 3, we describe the collaborative learning context which we analyze according to the methods presented in Section 4. We present our results in Section 5, and offer some in-depth interpretation. We end with a look forward, to future applications and extensions of this work.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 220–229, 2014.

2 Background

This paper grounds itself in the fields of Educational Data Mining and Computer Supported Collaborative Learning. In particular, we build upon prior work that has successfully employed a variety of methods for feature extraction and pattern learning to predict affective, collaborative, and learning outcomes from discourse.

Linguistic analysis methods for studying both individual learners and small groups [11] have been be used to assess cognitive and meta cognitive knowledge [10], critical thinking, knowledge construction [9] and consensus building techniques [16]. In many cases [5, 26], methods for automatically labeling these features are developed hand-in-hand with their application to a prediction task. Analysis applied to course message boards has shown it is possible to detect unresolved questions [12] in asynchronous discussions, and that patterns of interaction and participation can be used to predict final learning outcomes [21]. In the context of a single-user conversational tutor, a set of conversational features, including measures of the quality and content of student answers as derived from Latent Semantic Analysis [15], have been successfully applied to predict the moment-to-moment affect of the learner [5].

In intelligent tutoring systems with a conversational component, automated analysis methods may be employed as formative assessments, predicting student learning or collaborative performance. These predictions can be used to inform a tutor's interventions during future learning experiences, or to provide moment-by-moment facilitation in response to continuous assessment [1]. Recent work has demonstrated the power of data mining for building moment-to-moment models of student learning [2], although as this work was situated in a non-conversational tutoring system, it did not leverage linguistic features to anticipate learning. Fully automated coding and modeling methods have been used to successfully predict the outcome of a facilitated civil-dispute negotiation [26]. Models of conversational trajectory have also been developed as a source of feedback for learners and their human instructors, using a set of features describing conversational attributes derived from per-turn coding of a conversation [3, 4]. In that work, each coded move contributes to one of four underlying conversational dimensions (conformity, creativity, elaboration, and initiative), allowing concrete quantitative measures to power a qualitative analysis of group state.

Hidden Markov Models [20] trained on sequences of student-selected sentence-opener moves have been used to classify and describe groups of collaborative learners as more or less productive [24, 25]. HMMs have also been applied to surveys of participant emotion, to draw inferences about underlying affective or cognitive state [6]. However, such work has relied on participants selecting their next move or observed state from a limited set of options. More recent work has used n-grams or stretchy patterns [8] over discourse act labels to model local conversational structure and predict group task success [19]. Although this body of work illustrates the potential of sequential models for understanding student state, their suitability as a method for assessing individuals within an unconstrained multiparty discourse has not been fully explored.

3 Context and Corpus: College Chemistry Collaboration

We conduct our study data collected from a small-group chat-based collaborative task in the domain of college chemistry. The participants in this study were first-year undergraduate students in an introductory chemistry course, during a unit on intermolecular interactions. Students were randomly assigned to groups of three or four. Participation in the exercise was voluntary, and students had the option of not consenting for their data to be included in our research. Altogether, our analysis includes data from 50 consenting students from 16 different groups - with a mean of 93 messages per student, or 292 per group. Students were administered a pre-test the day before they completed the task, and completed a post-test the day after. Two test forms were randomly counter-balanced by student between pre- and post-test.

This task and chat environment have been used before to study methods for automatic discussion facilitation [1]. The 90-minute task focuses on intermolecular forces and their influence on the boiling points of liquids. The task was framed as a collaborative data analysis activity, where the students in each group were assigned to read individually about one of three classes of molecules, and the factors most likely to influence their boiling point. This division also provided intrinsic motivation for collaboration, as the task could not be completed without knowledge from each of the student experts. A conversational agent [14] facilitated the activity for each group, presenting the series of exercises to the group and prompting them to explain their reasoning to each other.

4 Methods: Predicting Learning from Conversation

We aim to capture the properties of conversation that are distinctive of more (or less) successful learners. Low-level lexical and syntactic features are examined alongside higher-order representations of discourse, and evaluated as candidates for automating future formative assessment. In order to assess individual learning, we first build a linear model, predicting student post-test score from pre-test score alone. This model accounts for 61% of variance in student performance. The impact of collaboration, if any, might be found in the remaining unaccounted-for variance. Thus, we use as our target the residual from this regression in the remainder of the analysis.

4.1 Baseline Textual Features

Especially in unstructured conversational data, the success of a machine learning algorithm is tied to the feature representation of the contents of that data. We first use **"bag-of-words"** features, which represents only the vocabulary used in a conversation (including both content words and function words). We then present a second model, based on **"complex language"** features. This model contains a superset of the bag-of-words feature set. Adjacent pairs of words (bigrams) and local syntactic part-of-speech bigrams are added as features.

In addition to a single student's language (the **Student Only** condition above), much of her learning may be tied up in her interactions with peers. We therefore introduce an additional text representation (**Whole Group**), including features for all students in each conversation as a second feature set. These new features are represented as distinct - thus any unigram may appear twice in an instance as a distinct feature, once if spoken by the student of interest, and again if spoken by any of her groupmates.

Finally, in order to evaluate our methods' suitability for mid-activity formative assessment, we also test the condition where only features from the first third of each student transcript are used for prediction (**Start Only**), stopping at the end of the first phase of the activity described in Section 3.

We train a Naive Bayes classifier to differentiate groups with a positive residual (learning more than the pre-test would suggest) from those with a negative residual. To avoid overfitting (identifying the peculiarities of individual groups, rather than overall trends in student behavior), results of our machine-learning experiments are presented from 16-fold leave-one-group-out cross-validation. In this arrangement, models are trained on 15 groups of 46 or 47 students, and tested on the remaining group of 3 or 4 students. Reported performance is averaged across groups. The model is limited to using the top 100 most predictive language features on each training fold, using χ^2 feature selection [7].

4.2 Active Learning Annotation

To represent features above the contributions of individual lines of dialogue, we refer to established frameworks for conversational analysis. In Barros et al.'s work, a set of attributes for qualitative conversational analysis is proposed [4] based on a set of six sentence-opening moves. This is similar to the scheme used by Soller [23]. We combine Barros' two types of proposal and consider just five types of "Active Learning" moves:

- Proposals (**PR**) begin a sequence and introduce a new concept or idea.
- Questions (**QU**) target proposals and question them.
- Clarifications (**CL**) are elaborations on proposals, or answers to questions.
- Agreements (**AG**) show agreement or assent between speakers in a sequence.
- Remaining contributions are Comments (**CM**); including topic statements, floor grabbing moves, pauses, etc.

In earlier works, assignment of turn labels relied on student inputs being constrained to a fixed set of sentence-openers. In our approach, the students are not thus fettered, and we instead rely on annotation of free text. To allow this flexibility, we adapted a coding manual based on the systemic functional linguistics "Negotiation" framework [17], describing the flow of information and action within a conversation. Recent work has shown that Negotiation annotation can be automated for freeform chatroom conversations [18]. With an eye toward such future automation, we adapted Mayfield's coding manual, converting Negotiation labels to Active Learning moves using heuristics. This manual

was first validated on separate pilot data. For this data, three transcripts (about 2000 turns of conversation) were coded by both annotators to check reliability, and the rest were each coded by a single annotator. Resulting reliability was high for Active Learning annotations, $\kappa = 0.75$.

From these annotations, we can now represent sequences of labeled turns as inputs to our machine learning algorithms. As a starting point (**Active Learning Trigrams**), we use sequences of three consecutive labels, extracted from the sequence of labeled turns, as a feature for our group and student tasks. In the case of per-student outcome prediction, each tag is differentiated based on who (relative to the student in question) is speaking - either the student herself, or another participant. For example, $\mathbf{PR}_s$ is a proposal issued by this student, $\mathbf{CL}_o$ is a clarification by another student, and so on. We consider this representation both on its own and as a supplement to our textual features.

As in Section 4.1, we train a Naive Bayes classifier with these features and report results from 16-fold cross-validation. As an additional experiment, we also evaluate a single classifier trained on the combined feature set of **Active Learning Trigrams** and **"complex language"** features.

4.3 Predicting Learning with Contrastive Hidden Markov Models

As a more sophisticated differentiator of conversational structure, we use Hidden Markov Models [20] to model variation between successful and unsuccessful students. HMMs are a sequential labeling algorithm, where observed behaviors are assumed to be a result of an unobserved, hidden state. In this case, states may correspond to a student's intention when contributing a new turn to the dialogue. By analyzing sequences of observed labels, HMMs can discover these unobserved states statistically.

Following Soller et al. [24], we train two HMMs with four hidden states, on sequences drawn from subsets of the corpus - one using the sequences from the four students with the highest residuals, the other using the four students with the lowest residuals. The resulting models should distinguish the sequential behaviors of unusually high- and low-performing students. We make no presumptions about the meanings of specific hidden states [6], although we expect to see meaningful patterns relevant to collaborative discourse.

As with our textual experiments, we use leave-one-group-out cross-validation, so no student transcript is evaluated on a model trained on a member of that transcript. For each held-out student in the test group, we calculate the normalized sequence likelihood of their entire transcript for each model, and use the likelihoods that the two models assign to the held-out data as features for a linear model performing binary classification. To mirror the **Start Only** conditions above, we also apply the same procedure to only the first third of the Active Learning sequences in each transcript, to assess this method's suitability for in-process formative assessment.

5 Results and Discussion

Results for the classification experiments using textual features are presented in Table 1. In general, we find that richer text features and including context from group members' posts both contribute to performance well above an individual student's vocabulary alone, and their benefit is somewhat additive. Further, in the more complex model we find that using only features from the starting section of each transcript perform statistically indistinguishably from models built on the entire transcript, suggesting that such methods may enable mid-activity formative assessments based on conversational features.

Table 1. Predicting individual learning above or below expected levels with textual features alone, based on raw accuracy (%) and Cohen's kappa. **Bold** represents a marginal improvement over baseline accuracy, $p < 0.1$.

	Student Only		**Whole Group**		**Start Only**	
Feature Set	%	κ	%	κ	%	κ
Bag-of-words	0.58	0.14	0.64	0.25	0.49	-0.01
Complex language	0.64	.025	**0.70**	**0.38**	0.68	0.38

In Table 2, we see the impact of Active Learning sequential features. Active Learning trigrams appear to offer additive benefit alongside textual features, improving our ability to predict student over- or underperformance. Using the more sophisticated contrastive HMM model, we are able to replicate this performance by only modeling states based on sequences of Active Learning tags. Table 3 lists a few features from this combination model that are highly predictive of high and low residual scores.

Table 2. Predicting individual learning above or below expected levels with sequential dialogue features. **Bold** represents marginal improvement over baseline, $p < 0.1$.

Sequence Representation	%	κ
Active Learning Trigrams	0.66	0.30
Trigrams + Textual Features	**0.72**	**0.43**
Contrastive HMMs	**0.72**	**0.44**
Contrastive HMMs (Start Only)	0.64	0.28

5.1 Qualitative Analysis of Contrastive HMMs

The output of the contrasting HMMs can be used to gain insight into the conversational habits of more (or less) successful students. Figure 1 illustrates the difference in transition patterns between student with higher and lower residual scores. Note that although the learned states were not predetermined, fairly consistent groupings emerge between models. In the model for higher scores, we see

Table 3. Representative features for high or low residual scores

Feature	Actor(s)	Class	Example
thinking	student	high	i'm thinking in between
ADV WH	other	high	so why not higher?
$\mathbf{CL}_o\mathbf{PR}_s\mathbf{QU}_s$	both	high	yep - the dipole moment is what's different chcl3 second highest, ch3cl third highest the last one has no dipole moment then?
agree ?	student	low	KCl will be in the middle ... agree?
ADJ CONJ	other	low	smaller or bigger?
$\mathbf{CL}_o\mathbf{QU}_o\mathbf{PR}_o$	other	low	i think the bp increases as we go down the table does all 3 increase down the table? i think the dipole moment is more important

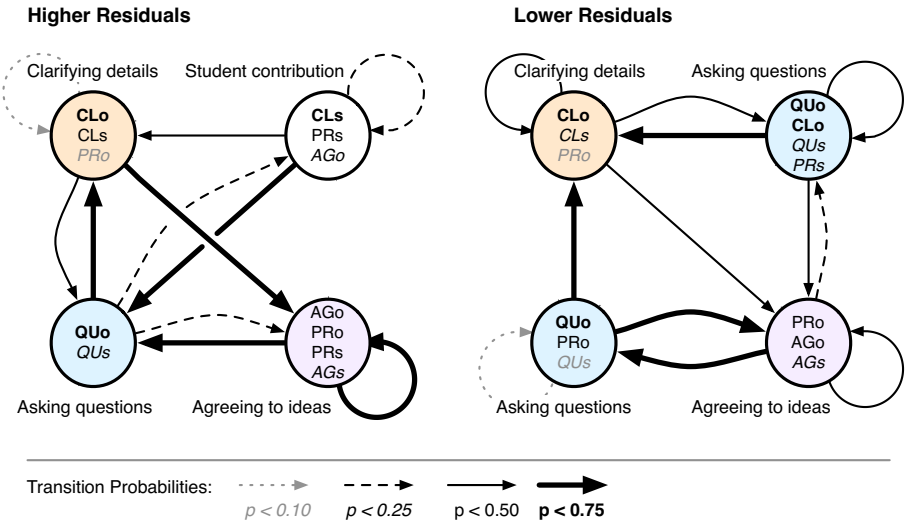


Fig. 1. Learned High and Low State Transitions

a strong flow between states that have high emission probabilities for questions and clarifying statements, and from clarification to agreement to proposals. In particular, the high-residual model favors transitions from questioning, to clarification, to agreement and new ideas, whereas there's a comparatively weak flow out of the clarification state in the low-residual model. The low-residual model also displays stronger tendencies toward loops in the clarification and questioning states. It may be that students who fit the lower-residual model find themselves in groups experiencing more confusion, but with less productive resolution. The low-residual model expects a lesser degree of student participation (as indicated by lower emission probabilities for student moves, versus moves by others). A hard-to-reach state focusing on student contributions is unique to the high-residual model, which favors reentry into the question-clarify-agree loop.

Table 4. Highly likely sequences, according to the HMMs for high and low residual scores (top and bottom). Note that comments are not included in the model.

Tag	Text
$\mathbf{PR}_o$	yea they are made up of the same molecules so i cant really tell yet
$\mathbf{QU}_s$	It's going to be in the middle right?
$\mathbf{CL}_o$	its going to be the smallest because the dipole moment is the smallest
$\mathbf{QU}_o$	so its actually smallest?
$\mathbf{CL}_s$	wait just kidding i read that wrong! Smallest.
$\mathbf{AG}_o$	ya smaller dipole=smaller boil pt

$\mathbf{PR_o}$	Polar molecules have a permanent dipole moment which is caused by differences in electronegativity between bonded atoms. One might have more electronegativity than the other causing a nonuniform electron distribution.
$\mathbf{CL_s}$	In my intro, it said dipole moments do not at all affect the boiling point
$\mathbf{PR_o}$	The table shows you it does though
$\mathbf{AG_s}$	yeah this one shows that it does
$\mathbf{CM_s}$	which is weird
$\mathbf{PR_o}$	They look like nonpolar molecules

Some highly probable sub-sequences according to each model are illustrated in Table 4, with examples from the corpus.

6 Conclusions and Future Work

The experiments presented in this paper identify successful methods for predicting learning outcomes from conversational transcripts. However, the small size of this dataset makes it difficult to draw robust conclusions of statistical significance. Future work will look to explore the predictive power of Active Learning sequences in larger-scale and more diverse collaborative learning contexts, and to pursue the potential in combining textual cues with conversational sequence information in a more sophisticated ways. Further, we hope to use such models as real-time formative assessments based on similar conversational cues to direct instruction and provide agile conversational support for collaborative learning.

Acknowledgements. This research was funded in part by NSF grants IIS-1320064 and OMA-0836012.

References

[1] Adamson, D., Dyke, G., Jang., H., Rosé, C.P.: Towards an agile approach to adapting dynamic collaboration support to student needs. International Journal of AI in Education (2013)

[2] Baker, R.S., Goldstein, A.B., Heffernan, N.T.: Detecting learning moment-by-moment. International Journal of Artificial Intelligence in Education 21(1), 5–25 (2011)
[3] Barros, B., Verdejo, M.: An approach to analyse collaboration when shared structured workspaces are used for carrying out group learning processes. In: International Conference on Artificial Intelligence in Education. Citeseer, Le Mans (1999)
[4] Barros, B., Verdejo, M.F.: Analysing student interaction processes in order to improve collaboration. The Degree Approach. International Journal of Artificial Intelligence in Education 11(3), 221–241 (2000)
[5] D'Mello, S.K., Craig, S.D., Witherspoon, A., Mcdaniel, B., Graesser, A.: Automatic detection of learner's affect from conversational cues. User Modeling and User-Adapted Interaction 18(1-2), 45–80 (2008)
[6] D'Mello, S.K., Graesser, A.: Modeling cognitive-affective dynamics with hidden markov models. In: Proceedings of the 32nd Annual Cognitive Science Society, pp. 2721–2726 (2010)
[7] Forman, G.: An extensive empirical study of feature selection metrics for text classification. The Journal of Machine Learning Research 3, 1289–1305 (2003)
[8] Gianfortoni, P., Adamson, D., Rosé, C.P.: Modeling of stylistic variation in social media with stretchy patterns. In: Proceedings of the First Workshop on Algorithms and Resources for Modelling of Dialects and Language Varieties, pp. 49–59. Association for Computational Linguistics (2011)
[9] Gunawardena, C.N., Lowe, C.A., Anderson, T.: Analysis of a global online debate and the development of an interaction analysis model for examining social construction of knowledge in computer conferencing. Journal of Educational Computing Research 17(4), 397–431 (1997)
[10] Henri, F.: Computer conferencing and content analysis. Series F: Computer and Systems Sciences (1992)
[11] Howley, I., Mayfield, E., Carolyn, P.: Linguistic analysis methods for studying small groups. In: The International Handbook of Collaborative Learning, ch. 10, Routledge (2013)
[12] Kim, J., Li, J., Kim, T.: Towards identifying unresolved discussions in student online forums. In: Proceedings of the NAACL HLT 2010 Fifth Workshop on Innovative Use of NLP for Building Educational Applications, pp. 84–91. Association for Computational Linguistics (2010)
[13] Kirschner, F., Paas, F., Kirschner, P.A.: A cognitive load approach to collaborative learning: United brains for complex tasks. Educational Psychology Review 21 (2009)
[14] Kumar, R.: Socially capable conversational agents for multi-party interactive situations. Ph.D. thesis, Carnegie Mellon University (2011)
[15] Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse Processes 25(2-3), 259–284 (1998)
[16] Leitão, S.: The potential of argument in knowledge building. Human Development 43(6), 332–360 (2000)
[17] Martin, J.R., Rose, D.: Working with discourse: Meaning beyond the clause. Continuum International Publishing Group (2003)
[18] Mayfield, E., Adamson, D., Rosé, C.P.: Hierarchical conversation structure prediction in multi-party chat. SIGDIAL 2012 (2012)
[19] Mayfield, E., Adamson, D., Rudnicky, A.I., Rosé, C.P.: Computational representational of discourse practices across populations in task-based dialogue. ICIC, Bangalore (2012)

[20] Rabiner, L., Juang, B.: An introduction to hidden markov models. IEEE ASSP Magazine 3(1), 4–16 (1986)
[21] Romero, C., López, M.I., Luna, J.M., Ventura, S.: Predicting students' final performance from participation in on-line discussion forums. Computers & Education 68, 458–472 (2013)
[22] Scardamalia, M., Bereiter, C.: Technologies for knowledge-building discourse. Communications of the ACM 36(5) (1993)
[23] Soller, A., Lesgold, A.: Analyzing Peer Dialogue from an Active Learning Perspective. In: Proceedings of the AI-ED 99 Workshop: Analysing Educational Dialogue Interaction: Towards Models that Support Learning, pp. 63–71 (1999)
[24] Soller, A., Lesgold, A.: A Computational Approach to Analyzing Online Knowledge Sharing Interaction. In: Proceedings of Artificial Intelligence in Education 2003, Sydney, Australia (2003)
[25] Soller, A., Wiebe, J., Lesgold, A.: A Machine Learning Approach to Assessing Knowledge Sharing During Collaborative Learning Activities. In: Proceedings of Computer-Support for Collaborative Learning 2002, Boulder, CO (2002)
[26] Twitchell, D.P., Jensen, M.L., Derrick, D.C., Burgoon, J.K., Nunamaker, J.F.: Negotiation outcome classification using language features. Group Decision and Negotiation 22(1), 135–151 (2013)
[27] Webb, N.M., Palinscar, A.S.: Group processes in the classroom. In: Berliner, D.C., Calfee, R.C. (eds.) Handbook of Educational Psychology, pp. 841–873. Prentice Hall, New York (1996)

Validating the Automated Assessment of Participation and of Collaboration in Chat Conversations

Mihai Dascalu[1], Ştefan Trausan-Matu[1], and Philippe Dessus[2]

[1] Politehnica University of Bucharest, Computer Science Department, Romania
mihai.dascalu@cs.pub.ro, stefan.trausan@cs.pub.ro
[2] LSE, Univ. Grenoble Alpes, France
philippe.dessus@upmf-grenoble.fr

Abstract. As Computer Supported Collaborative Learning (CSCL) gains a broader usage as a viable alternative to classic educational scenarios, the need for automated tools capable of supporting tutors in the time consuming process of analyzing conversations becomes more stringent. Moreover, in order to fully explore the benefits of such scenarios, a clear demarcation must be made between *participation* or active involvement, and *collaboration* that presumes the intertwining of ideas or points of view with other participants. Therefore, starting from a cohesion-based model of the discourse, we propose two computational models for assessing collaboration and participation. The first model is based on the cohesion graph and can be perceived as a longitudinal analysis of the ongoing conversation, thus accounting for participation from a social knowledge-building perspective. In the second approach, collaboration is regarded from a dialogical perspective as the intertwining or overlap of voices pertaining to different speakers, therefore enabling a transversal analysis of subsequent discussion slices.

Keywords: Computer Supported Collaborative Learning, Cohesion-based Discourse Analysis, Dialogism, Participation Assessment, Collaboration Evaluation.

1 Introduction

Computer Supported Collaborative Learning (CSCL) gains a broader usage in several newest educational settings, like MOOCs or collaborative serious games, as a viable alternative to classic educational scenarios. The need for automated tools capable of supporting all their actors in the time consuming process of analyzing conversations becomes more stringent. Chat conversations or forums became the place where knowledge is collaboratively built and shared [1] and there is a complex intertwining between collective and individual learning processes that is worth analyzing [2].

Shortly put, two complementary analysis approaches compete. The first one is *structural*, uses Social Network Analyses and stems from group dynamics to unveil relationships between individuals to sketch networks of collaboration [3]. The second approach is *dialogical*, has roots in discourse theories [4] and uses Natural Language

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 230–235, 2014.

Processing techniques to analyze the semantic cohesion of textual utterances (e.g., sentences or paragraphs).

After devising several systems inspired from the dialogical approach [5] and using a cohesion-based model of the discourse as underlying structure [6], we propose computational models for assessing participation and collaboration. Within our approach, *participation* is regarded as cumulative qualitative utterance scores and is modeled through the interaction graph presented in the second section. Section three introduces two computational models for assessing *collaboration*. The first one is based on the cohesion graph [7] and can be perceived as a longitudinal analysis of the ongoing conversation, thus accounting for participation from a social knowledge-building perspective. In the second model, collaboration is regarded from dialogism as the intertwining or overlap of voices pertaining to different speakers, therefore enabling a transversal analysis of subsequent discussion slices. This paper is the occasion to present in the fourth section the results of a large-scale validation by comparing the outputs of our system with human evaluations.

2 Participation Assessment

Measuring participation in virtual groups and communities on the web communicating through chats, forums or different types of social networking was performed in the *structural approach* by considering the number of emitted posts or utterances and by using several social networks metrics like centrality (number of links to other nodes), betweenness (nodes that, if eliminated would highly reduce or eliminate communication among other participants) [8] or page-rank derived formulas [9] in the interaction graph with users as nodes and posts as arcs [9]. Sometimes arcs have weights computed in different ways, from the simplest number of posts to more complicated metrics, considering the language content of the messages, like in our *dialogical* approach, which will be presented below.

The assessment of participation of each student in CSCL chats has some differences from the cases of forums or other social networking due to the small number of participants (typical examples are 3 to 7 students) and the large number of exchanged utterances. In this case, due to the fact that for chat conversations we are dealing in most cases with a complete graph, betweenness score for all nodes is 0. Centrality also is not a very significant discriminant: only participants with very low number of emitted utterances are not central.

In our approach, we are taking a perspective based on natural language processing of the *content of utterances*, considering the *topics* that were supposed to be discussed (for example, stated by the teacher in a CSCL homework) and focusing on discourse analysis. The latter's defining feature is *cohesion* and our approach is fundamentally based on it. From a computational point of view, cohesion is computed as a combination of semantic distances in ontologies, semantic similarity from Latent Semantic Analysis vector spaces, and Latent Dirichlet Analysis topic models [7]. Starting from this aggregated similarity function, a multi-layered cohesion graph is built [10] that models through cohesive links the dependencies between the key

elements of the analysis: the whole conversation, participants' utterances and sentences for longer posts. The previous links can be either *explicit* if participants marked the dependencies within the user interface, *enforced* for hierarchical links and adjacent analysis elements or *implicit*, if cohesion exceeds a threshold value [10].

In terms of participation, we start with the identification of discussion topics for each participant for pinpointing out if the needed concepts were covered. One of the most important metrics is the *utterance score* that, shortly put, represents the overall topics coverage augmented through cohesion with inter-linked analysis elements [10]. In this aim, an *interaction graph* is built with participants as nodes and the weight of links equal to the sum of utterances scores multiplied by the cohesion with the inter-linked analysis elements [10].

3 Collaboration Evaluation

In order to thoroughly assess collaboration, we have proposed two computational models. The first model [6] represents a refinement of the gain-based collaboration assessment [11] and takes full advantage of the cohesion graph [12]. The second is a novel approach that evaluates collaboration as an intertwining or overlap of voices pertaining to different speakers. The main difference between the two is that the first focuses on the ongoing conversations, therefore on its longitudinal dimension, whereas the later considers subsequent slices of the conversation, the synergy of voices, in other words the transversal dimension.

3.1 Social Knowledge-Building Model

The actual information transfer through cohesive links from the cohesion graph can be split between a personal and a social knowledge-building process [1, 13, 14] at utterance level. Firstly, a *personal dimension* emerges by considering utterances with the same speaker, therefore modeling a kind of inner voice or continuation of the discourse. Secondly, inter-changed utterances with different speakers define a *social perspective* that models collaboration as a cumulative effect. Our model is similar to some extent to the gain-based collaboration model [11] and marks a transition towards Stahl's model of collaborative knowledge-building [1] by representing a conversation thread as a multi-layered cohesion graph.

The continuation of ideas or explicitly referencing utterances of the same speaker builds an inner dialogue or personal knowledge, whereas the social perspective measures the interaction with other participants, encourages idea sharing and fosters creativity for working in groups [15], thus enabling a truly collaborative discussion. Moreover, personal knowledge building addresses individual voices (participant voices or implicit/alien voices covering the same speaker), while social knowledge building, derived from explicit dialog (that by definition is between at least two entities), sustains collaboration and highlights external voices.

3.2 Dialogical Voice Inter-animation Model

In order to achieve genuine collaboration, the conversation must contain threads of utterances integrating key concepts ('voices', in the musical polyphonic sense [16]) that inter-animate in a similar way to counterpoints in polyphonic fugues. Voices are present in utterances from multiple participants of the conversation. In order to obtain an operationalization, a shift of perspective is required from voices, computed as semantic chains of related concepts, towards an individual participant. As collaboration is centered on multiple participants, a split of each voice into multiple viewpoints pertaining to different participants is required. A viewpoint consists of a link between the concepts pertaining to a voice and a participant, through their explicit use within one's interventions in the ongoing conversation. We opted to present this split in terms of implicit (alien) voices [17]. Moreover, this split presentation of semantic chains per participant is useful for observing each speaker's coverage and distribution of dominant concepts throughout the discussion.

In addition, in order to identify the voice overlaps now pertaining to different participants, we changed from an ongoing longitudinal analysis of the discourse, presented in the previous subsection, to a transversal analysis of a context consisting of several adjacent utterances. We use a cumulated value of Pointwise Mutual Information (PMI) obtained from all possible pairs of voices pertaining to different participants (different viewpoints), within subsequent contexts of the analysis (within our implementation we used a sliding window of 5 interventions in order to model the local context of each voice occurrence). From an individual point of view, each participant's overall collaboration can be seen as the cumulated mutual information between his personal viewpoints and all other participant viewpoints. Therefore, by comparing individual voice distributions that span throughout the conversation, collaboration emerges from the overlap of voices pertaining to different participants.

4 Participation and Collaboration Validation

The validation experiments focused on the assessment of 10 chat conversations that took place in an academic environment in which Computer Science students from the 4th year undergoing the Human-Computer Interaction course at our university debated on the advantages and disadvantages of CSCL technologies. Each conversation involved 4 or 5 participants who each had to support a given technology (e.g., chat, blog, wiki, forum or Google Wave) in specific use case scenarios during the first phase of the discussion, later on proposing an integrated alternative that would encompass the previously presented advantages. The 10 conversations were manually selected from a 10 times larger corpus of chats.

Afterwards, 76 4th year undergraduate students following the same course, but from a different generation, and 34 1st year master students attending the Adaptive and Collaborative Systems course were each asked to manually annotate 3 chat conversations. We opted to distribute the evaluation of each conversation due to the high amount of time it takes to manually assess a single discussion (on average, users reported 1.5 to 4 hours for a deep understanding) [18]. In the end, we had on average

33 annotations per conversation and the overall results indicated a reliable automatic evaluation of both participation and collaboration. We validated the machine vs. human agreement by firstly computing intra-class correlations between raters for each chat (avg $ICC_{participation} = .97$; avg $ICC_{collaboration} = .90$) and, secondly, as these correlations were all very high indicating very few disagreements between raters, non-parametric correlations (avg $Rho_{participation} = .84$; avg $Rho_{collaboration} = .74$) were determined between machine vs. human mean ratings for each chat.

5 Conclusions and Future Research Directions

Starting from a dialogic model of discourse centered on cohesion, we thoroughly validated our system in terms of analyzing chat participants' involvement and collaboration, the later employing a longitudinal model based on social knowledge-building and a different transversal model based on voice inter-animation. Moreover, as the validations proved the accuracy of the models built on dialogism, we can state that the proposed methods emphasize the dialogical perspective of collaboration in CSCL environments.

In addition, the analyses performed in this paper have a very broad spectrum of applications, extending from utterance cohesion towards group cohesion rooted in collaboration. Beyond the rather simple visualization of individual and collective involvement, our developed system is also well-suited to enable students to self-regulate their learning.

Acknowledgements. We would like to thank the students of University "Politehnica" of Bucharest who participated in our experiments. This research was partially supported by the 264207 ERRIC–Empowering Romanian Research on Intelligent Information Technologies/FP7-REGPOT-2010-1 project.

References

1. Stahl, G.: Group cognition. Computer support for building collaborative knowledge. MIT Press, Cambridge (2006)
2. Cress, U.: Mass collaboration and learning. In: Luckin, R., Puntambekar, S., Goodyear, P., Grabowski, B., Underwood, J., Winters, N. (eds.) Handbook of Design in Educational Technology. Routledge, New York (2013)
3. Reffay, C., Martinez-Mones, A.: Basic concepts and techniques in social network analysis. In: Luckin, R., Puntambekar, S., Goodyear, P., Grabowski, B., Underwood, J., Winters, N. (eds.) Handbook of Design in Educational Technology, pp. 448–456. Routledge, New York (2013)
4. Bakhtin, M.M.: The dialogic imagination: Four essays. The University of Texas Press, Austin and London (1981)
5. Dascalu, M., Rebedea, T., Trausan-Matu, S., Armitt, G.: PolyCAFe: Collaboration and Utterance Assessment for Online CSCL Conversations. In: CSCL 2011, vol. 2, pp. 781–785. ISLS, Hong Kong (2011)

6. Dascalu, M., Trausan-Matu, S., Dessus, P.: Cohesion-based Analysis of CSCL Conversations: Holistic and Individual Perspectives. In: CSCL 2013, vol. 1, pp. 145–152. ISLS, Madison (2013)
7. Dascalu, M., Dessus, P., Trausan-Matu, Ş., Bianco, M., Nardy, A.: ReaderBench, an Environment for Analyzing Text Complexity and Reading Strategies. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 379–388. Springer, Heidelberg (2013)
8. Brandes, U.: A Faster Algorithm for Betweenness Centrality. Journal of Mathematical Sociology 25(2), 163–177 (2001)
9. Dascalu, M., Chioasca, E.-V., Trausan-Matu, S.: ASAP – An Advanced System for Assessing Chat Participants. In: Dochev, D., Pistore, M., Traverso, P. (eds.) AIMSA 2008. LNCS (LNAI), vol. 5253, pp. 58–68. Springer, Heidelberg (2008)
10. Dascalu, M.: Analyzing Discourse and Text Complexity for Learning and Collaborating. SCI, vol. 534. Springer, Switzerland (2014)
11. Trausan-Matu, S., Dascalu, M., Rebedea, T.: A system for the automatic analysis of Computer-Supported Collaborative Learning chats. In: ICALT 2012, pp. 95–99. IEEE, Rome (2012)
12. Trausan-Matu, S., Dascalu, M., Dessus, P.: Textual Complexity and Discourse Structure in Computer-Supported Collaborative Learning. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 352–357. Springer, Heidelberg (2012)
13. Bereiter, C.: Education and Mind in the Knowledge Age. Lawrence Erlbaum Associates, Mahwah (2002)
14. Scardamalia, M.: Collective cognitive responsibility for the advancement of knowledge. In: Smith, B., Bereiter, C. (eds.) Liberal Education in a Knowledge Society, pp. 67–98. Open Court Publishing, Chicago (2002)
15. Trausan-Matu, S.: Computer Support for Creativity in Small Groups using chats. Annals of the Academy of Romanian Scientists, Series on Science and Technology of Information 3(2), 81–90 (2010)
16. Trausan-Matu, S., Rebedea, T.: Polyphonic Inter-Animation of Voices in VMT. In: Stahl, G. (ed.) Studying Virtual Math Teams, pp. 451–473. Springer, Boston (2009)
17. Trausan-Matu, S., Stahl, G.: Polyphonic inter-animation of voices in chats. In: CSCL 2007 Workshop on Chat Analysis in Virtual Math Teams, p. 12. ISLS, New Brunwick (2007)
18. Trausan-Matu, S.: Automatic Support for the Analysis of Online Collaborative Learning Chat Conversations. In: Tsang, P., Cheung, S.K.S., Lee, V.S.K., Huang, R. (eds.) ICHL 2010. LNCS, vol. 6248, pp. 383–394. Springer, Heidelberg (2010)

Context-Based Speech Act Classification in Intelligent Tutoring Systems

Borhan Samei[1], Haiying Li[1], Fazel Keshtkar[2], Vasile Rus[1], and Arthur C. Graesser[1]

[1] University of Memphis, Institute for Intelligent Systems, TN, USA
{bsamei,hli5,vrus,graesser}@memphis.edu
[2] Southeast Missouri State University, MO, USA
fkeshtkar@semo.edu

Abstract. In intelligent tutoring systems with natural language dialogue, speech act classification, the task of detecting learners' intentions, informs the system's response mechanism. In this paper, we propose supervised machine learning models for speech act classification in the context of an online collaborative learning game environment. We explore the role of context (i.e. speech acts of previous utterances) for speech act classification. We compare speech act classification models trained and tested with contextual and non-contextual features (contents of the current utterance). The accuracy of the proposed models is high. A surprising finding is the modest role of context in automatically predicting the speech acts.

Keywords: speech act· machine learning· intelligent tutoring systems.

1 Introduction

Speech act classification is one of the indispensable components of dialogue-based intelligent tutoring systems (ITS) because speech act categories dramatically constrain the system's response [1, 2]. For example, when a student asks a question, the system should respond very differently than when the student asserts a fact or expresses being lost. Speech act classification is used for detecting students' intentions (Is the student asking a question or asserting a fact?). More precisely, speech act classification is framed as a classification task in which the goal is to detect the speech act categories of a given utterance from a predefined set of categories that together form the speech act taxonomy. The speech act taxonomy is usually predefined by researchers although attempts to automatically discover it from data are emerging [3]. We used a predefined taxonomy in the present paper [4].

The models in this paper will be incorporated in a multiparty simulation game on urban planning, called Land Science, an expansion of Urban Science [5]. The previous model of speech act classification in Land Science relied entirely on the lexical, semantic, and discourse features of the individual utterances without considering previous utterances within the context [3,4]. However, conversation progresses dependent on the previous utterances or context. For instance, after a greeting a greeting is more likely. Therefore, this study aims to investigate the role of context in speech act classification.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 236–241, 2014.

Speech act classification has theoretical roots in Austin's language as action theory [6] and subsequent work by Searle [7,8]. Different speech act taxonomies have been used in different domains of application. D'Andrade and Wish proposed seven categories of speech acts with high inter-annotator agreement among human judges: assertions, questions, requests and directives, reactions, expressive evaluations, commitment, and declaration [9].

Researchers have proposed several other taxonomies that are sensitive to various tasks and knowledge domains. Rus et al (2012) developed a data-driven method for automatically discovering speech act categories from online chats that were extracted from educational games, Urban Science and Land Science [3]. They applied utterance clustering methods based on the content of utterances and tried to find the natural groupings of the utterances in a fully automatic approach. The clusters were then deemed as speech act categories by assigning semantic names to the automatically discovered clusters.

Rasor et al (2011) proposed a machine learning approach using decision trees to automate the speech act classification in student chat interactions [10]. Olney et al. (2003) proposed a rule-based approach to classify speech acts by focusing on 16 categories of questions [11]. The Question category is important in an ITS because the tutor/mentor is expected to give answer to students' questions. Therefore, the first step is to identify questions in student utterances.

Moldovan et al. (2011) developed automated speech act classification for Land Science epistemic game [4]. The categories of their taxonomy included the same seven categories as Rus et al. [3]: *Statement, Request, Reaction, Metastatement, Greeting, ExpressiveEvaluation, and Question.* Using a supervised machine learning approach, they examined several models with feature sets containing the 2-8 leading tokens of the utterance and found that using 3 leading tokens achieves more accurate results. Based on their approach, our model uses the two leading tokens, the last token, and the length of utterance as features and we used the same taxonomy.

2 Method

Our approach to speech act classification is a supervised machine learning approach. In supervised machine learning approach, models of the tasks are proposed as sets of features. Parameters of these models are learned/trained from annotated data and the performance of the learned models is then assessed on new, test data. The parameters of the proposed models are learned using several machine learning algorithms, i.e. decision trees and naïve Bayes.

The feature set used in our models was designed based on two principles: first, it is intuitively inferred and tested that human identified the speech act of an utterance as soon as they heard the first few words [4], namely, the first leading tokens. However, the context of an utterance is assumed to improve the accuracy. Thus, another feature set included the contextual information, e.g. speech act category of the last few utterances. Our model adds context to previous models that relied merely on the contents of current student utterance [4].

Briefly, our feature set consists of content (non-contextual) features of the current utterance and contextual features (speech acts and speaker of previous utterances). The non-contextual features include the first two tokens and the last token which were represented as the actual string of characters (tokens) and the length of the utterance in words. The contextual features captured contextual information with the five prior utterances (the speech acts and actual speakers of these utterances). Our taxonomy consisted of seven categories. Table 1 shows examples extracted from the actual utterances for each category.

Table 1. Speech act taxonomy of seven categories with examples

Speech act category	*Example from dataset*
ExpressiveEvaluation	Your stakeholders will be grateful!
Greeting	Hello!
MetaStatements	oh yeah, last thing.
Statement	a physical representation of data.
Question	What should we do?
Reaction	Thank you
Request	Please check your inbox

Our training data was extracted from a dataset of mentor-student chat utterances from seven Land Science games. A total number of 26,148 chat utterances were generated by the players and the mentor. We randomly extracted chat utterances to form our training data and adjusted the training data to include an even distribution of 30 instances per speech act category.

This data set was annotated by one human expert within the context of the chats. The human expert had access to the whole dialogue and context of the conversation. This annotated data set is deemed as the reference annotation and includes 30 utterances per speech act category.

In order to examine the impact of the limited contextual information defined in our automated models (speech acts of previous five utterances), the data set was further annotated by a second human judge in two forms. First, the utterances were randomly ordered and the rater annotated them without considering the limited context. Second, each utterance was accompanied by the speech act category (not the content) of five prior utterances and rater annotated the data considering the contents of the current utterance and prior context.

In the first form of annotations, the rater showed a kappa of 0.55 in agreement with reference annotations. The agreement with reference annotations was improved to 0.75 kappa when the rater was provided with contextual information. On the other hand, the agreement of the rater with himself on the two forms of annotations (with/without context) was about 0.6 *kappa* which implies that having some sort of information about context, changes human's judgments and improves their accuracy compared to reference annotations.

Using the reference annotation data set, we applied J48 decision trees and Naïve Bayes machine learning models to create the automated speech act classifier with different feature sets of contextual and semantic information to examine the role of context. The performance of our models is presented in next section.

3 Results

Based on the human annotation, having contextual information improves the accuracy of human judgments. In fact, the more we know about context the better we can make decisions. Our feature set consists of two types of features: A set of 10 features which represent the context of the utterance by looking at the speech act category and speaker of five prior utterances (contextual features), and 4 features representing the semantic information of the individual utterances including the first two tokens, last token, and the length of the utterance (semantic features). The performance of proposed models was tested with feature sets of contextual, semantic and both.

Using the reference annotations as our training data, we created J48 decision trees and Naïve Bayes learning models using WEKA [12] and we tested our models with 10-fold cross validation. The overall performance of models was evaluated with the three feature sets (contextual, semantic, and semantic & context).

Table 2. Overall Accuracy and Kappa statistics of Naïve Bayes and J48 decision tree models with different feature sets

	J48 decision tree		***Naïve Bayes***	
Feature set	Accuracy (%)	Kappa	Accuracy (%)	Kappa
Contextual	23.80	0.11	37.14	0.26
Semantic	55.71	0.48	53.80	0.29
Contexual & Semantic	56.19	0.48	54.76	0.47

As seen in Table 2, using only contextual features provides enough clue to predict the speech act categories with an accuracy of about 37% with Naïve Bayes model. The semantic features improve the accuracy of J48 model to 55%, with 0.48 kappa. Using both kinds of features together, surprisingly, showed a low impact on the performance. Adding context to semantic feature set improved Naïve Bayes algorithm while the performance of the J48 model did not change by adding contextual features.

Overall, J48 model had better performance. To take a closer look at the role of context in our models, we examined the performance of J48 models on predicting each of the speech act categories. Table 3 shows the precision and recall on each category for models with different feature sets.

Table 3. The performance of J48 models on predicting each speech act category with different feature sets

	Contextual		**Semantic**		**Cont. & Sem.**	
Category	*Precision*	*Recall*	*Precision*	*Recall*	*Precision*	*Recall*
Expressive Evaluation	0.22	0.30	0.35	0.93	0.35	0.93
Greeting	0.36	0.43	0.73	0.63	0.73	0.63
Metastatement	0.30	0.36	0.64	0.60	0.60	0.56
Question	0.20	0.20	0.76	0.63	0.63	0.70
Reaction	0.08	0.06	0.50	0.13	0.13	0.21
Request	0.18	0.13	0.70	0.46	0.50	0.60
Statement	0.23	0.16	0.62	0.50	0.53	0.58

As shown in Table 3, adding contextual features to the semantic feature set improves the recall on some categories, such as Question, Reaction, Request, and Statement, whereas the precision on the categories gets lower by adding context. Overall adding context to the feature set had a modest impact on the performance of models.

4 Conclusion

In this paper, we examined the role of context (i.e., prior speech act categories and speakers, but not the actual content) in the performance of automated speech act classification. Contextual features seem to not have a significant impact on the overall performance of models; however adding context improves the performance on certain categories.

The results presented in previous sections showed that having some sort of contextual information has a positive impact on the accuracy of speech act classification for both human and computer. The models presented in this paper can be improved with having a larger training data and adjusting the features sets. The taxonomy also can be modified to multi-layer structure which enables the use of multiple feature sets to maximize the accuracy on certain categories.

For future work, we plan to test our model on different and new data sets once available. The models can be applied to different domains to explore the possible improvements. We will also investigate different types and representations of contextual features which can be used in the System to improve the accuracy.

Acknowledgements. This work was supported by the National Science Foundation BCS 0904909 and DRK-12-0918409. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of these funding agencies, cooperating institutions, or other individuals.

References

1. Graesser, A.C., D'Mello, S.K., Hu, X., Cai, Z., Olney, A., Morgan, B.: AutoTutor. In: McCarthy, P., Boonthum-Denecke, C. (eds.) Applied Natural Language Processing: Identification, Investigation, and Resolution, pp. 169–187. IGI Global, Harshey (2012)
2. Rus, V., D'Mello, S., Hu, X., Graesser, A.C.: Recent Advances in Intelligent Systems with Conversational Dialogue. AI Magazine 34, 42–54 (2013)
3. Rus, V., Graesser, A.C., Moldovan, C., Niraula, N.: Automated Discovery of Speech Act Categories in Educational Games. In: Yacef, K., Zaïane, O., Hershkovitz, H., Yudelson, M., Stamper, J. (eds.) Proceedings of the 5th International Conference on Educational Data Mining, pp. 25–32 (2012)
4. Moldovan, C., Rus, V., Graesser, A.C.: Automated Speech Act Classification For Online Chat. In: The 22nd Midwest Artificial Intelligence and Cognitive Science Conference, pp. 23–29 (2011)
5. Shaffer, D.W., Gee, J.P.: Epistemic Games as Education for Innovation. BJEP Monograph Series II, Number 5-Learning through Digital Technologies 1(1), 71–82 (2007)

6. Austin, J.L.: How To Do Things With Words. Harvard University Press, Cambridge (1962)
7. Searle, J.R.: Speech acts: An essay in the philosophy of language, vol. 626. Cambridge University press, Cambridge (1969)
8. Searle, J.R.: A Taxonomy of Illocutionary Acts. In: Gunderson, K. (ed.) Language, Mind and Knowledge, Minneapolis, pp. 344–369 (1975)
9. D'Andrade, R.G., Wish, M.: Speech Act Theory in Quantitative Research on Interpersonal Behavior. Discourse Processes 8(2), 229–258 (1985)
10. Rasor, T., Olney, A., D'Mello, S.K.: Student Speech Act Classification Using Machine Learning. In: Proceedings of the International Florida Artificial Intelligence Research Society Conference. AAAI Press (2011)
11. Olney, A., Louwerse, M.M., Mathews, E.C., Marineau, J., Mitchell, H.H., Graesser, A.C.: Utterance classification in AutoTutor. Building Educational Applications using Natural Language Processing. In: Burstein, J., Leacock, C. (eds.) Proceedings of the Human Language Technology - North American Chapter of the Association for Computational Linguistics Conference Workshop, pp. 1–8. Association for Computational Linguistics, Philadelphia (2003)
12. Hall, M., Frank, E., Holmes, G., Pfahringer, B.: The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1) (2009)

Macro-adaptation in Conversational Intelligent Tutoring Matters

Vasile Rus, Dan Stefanescu, William Baggett, Nobal Niraula,
Don Franceschetti, and Arthur C. Graesser

The University of Memphis, Memphis, TN, 38152, USA
vrus@memphis.edu

Abstract. We present in this paper the findings of a study on the role of macro-adaptation in conversational intelligent tutoring. Macro-adaptivity refers to a system's capability to select appropriate instructional tasks for the learner to work on. Micro-adaptivity refers to a system's capability to adapt its scaffolding while the learner is working on a particular task. We compared an intelligent tutoring system that offers both macro- and micro-adaptivity (fully-adaptive) with an intelligent tutoring system that offers only micro-adaptivity. Experimental data analysis revealed that learning gains were significantly higher for students randomly assigned to the fully-adaptive intelligent tutor condition compared to the micro-adaptive-only condition.

Keywords: macro-adaptation, intelligent tutoring systems, assessment.

1 Introduction

We address in this paper the role of macro-adaptivity in ITSs. We study the role of macro-adaptivity in the context of conversational or dialogue-based ITSs (Rus et al.; 2013). These ITSs interact with the students primarily through conversation although other elements, such as images associated with instructional tasks, may accompany the dialogue. Our target domain is conceptual Newtonian Physics and our target population is college students taking an introductory course in Physics, (e.g. nursing, engineering students, or even Physics majors).

Currrent state-of-the-art ITSs are quite effective. An extensive review of tutoring research by VanLehn (2011) showed that the effectiveness of computer tutors (d = 0.78) is as high as the effectiveness of human tutors. Furthermore, it was found that the effectiveness of human tutoring is not as high as it was originally believed (effect size d = 2.0) but much lower (d = 0.79). Relevant questions arise from these findings. Where does the effectiveness come from and how can it be further increased? The conventional wisdom of the last decade or so has speculated that as interactivity of tutoring increases, the effectiveness of tutoring should keep increasing. However, VanLehn (2011) reported that as interactivity of tutoring increases, the effectiveness of human and computer tutors plateaus.

There are several aspects of state-of-the-art conversational ITSs that may explain their plateau in effectiveness. First, they do not emphasize macro-adaptation through selection of learner-specific content and tasks, which is needed when students begin a

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 242–247, 2014.

tutoring session with different backgrounds. Second, while tutorial strategies are somehow understood, that is not necessarily the case for tutorial tactics that control tutors' actions at micro-level, e.g. decisions about step in a solution to a problem (VanLehn, Jordan, & Litman, 2007). Third, existing conversational ITSs emphasize mostly cognitive aspects. Other aspects of learning, such as affect and motivation, are less considered. Researchers have started to address at least two of the above three aspects that could lead to further increases in ITSs effectiveness: tutorial tactics (VanLehn, Jordan, & Litman, 2007) and affect (Lehman et al., 2011). We investigate in this paper the role of the less studied aspect, i.e. macro-adaptivity. Therefore, our research complements existing efforts towards better effectiveness of ITSs.

It should be noted that the role of macro-adaptation was noted early on (Brusilovsky, 1992). Attempts to handle macro-adaptivity have been made but their exact impact on learning gains has not been pursued to the best of our knowledge. For instance, while the intelligent tutor ANDES (VanLehn et al., 2005) relies on a student model which could be used for macro-adaptation, it was never used for this purpose (Conati, Gertner, & VanLehn. 2002; VanLehn et al., 2005). In fact, there is one ITS that focuses exclusively on macro-adaptation. Indeed, the mathematics tutor ALEKS offers macro-adaptation only. Once a task has been selected for a learner, the learner sees an identical worked-out solution to the task as any other student that was assigned the same task. That is, within a task all students see same information following a one-size-fits-all approach (no micro-adaptivity). Interestingly, a recent study showed that ALEKS can offer significant learning gains comparable to other ITSs (Sabo, Atkinson, Barrus, Joseph, Perez, 2013). This result emphasizes the importance of macro-adaptation in intelligent tutoring.

Our work here offers further support for the important role of macro-adaptation in tutoring. In particular, we offer a glimpse at the important role of macro-adaptation in conversational ITSs. To achieve our goal, we compared a fully-adaptive conversational ITS that offers both macro- and micro-adaptivity, i.e. a fully-adaptive system, with a micro-adaptive-only ITS. In the fully-adaptive ITS, instructional tasks for a particular student were selected based on the knowledge level of the student. We defined four distinct knowledge levels based on a global analysis of the performance on the pre-test of our subject sample. Each individual student was then placed at a corresponding knowledge level based on his performance on the pre-test. The selection of instructional tasks for each knowledge level was based on the idea that tasks should target concepts that students in a knowledge level are just beginning to understand ("green shoots", i.e. concepts ready to emerge) while students at the immediately higher (and even higher) knowledge levels already show proficiency (to them, these look like full-grown concepts).

2 Data-Driven Macro-adaptation

The basis of our data-driven macro-adaptation is a multiple-choice test that participants were given prior to undergoing training. The pre-test consists of 24 multiple-choice questions from Force Concept Inventory (FCI; Hestenes, Wells, & Swackhamer, 1992), 8 multiple-choice questions from Alonzo and Steedle (2009; (A&S), and 7 multiple-choice questions of our own (total=39 questions). Students

took the pre-test 2-3 weeks before the actual training in order to mitigate tiring effects during the actual training session and for logistical reasons. The training session consisted of about 1 hour of training with one of our ITSs, followed by 30 minutes of post-test taking (post-test was identical to the pre-test taken weeks before).

Once the student responses (*n=49*) on the pre-test were available, we selected critical concepts that students were struggling with based on Item Characteristic Functions (Wang and Bao, 2010) and defined knowledge levels based on this analysis. There is an Item Characteristic Function for each pre-test question which indicates the probability of answering the question correctly for various levels of student proficiency. In our case, instead of using directly student proficiency levels as given by, for instance, an Item Response Theory (IRT) analysis, we relied on the overall pre-test score. Due to the small *n*, an IRT analysis would have not been possible in our case. The use of the overall pre-test score as an approximation of proficiency level is reliable as explained next. Wang and Bao (2010) conducted an IRT analysis of FCI and confirmed the correctness of the unidimensional assumption needed for IRT analysis, i.e. a factor analysis revealed that existence of a dominant factor explaining college students' abilities to answer FCI questions. Furthermore, they showed a correlation of 0.994 between the overall FCI score (#correctly-answered/total-questions) and IRT proficiency levels.

In order to facilitate the selection of targeted concepts for training, we divided the space of proficiency levels into four knowledge levels: low knowledge, medium-low knowledge, high-medium knowledge, and high-knowledge. These knowledge levels offer a more fine distinction among students than the typical binary categorization (low vs. high knowledge) but less than the finest-grain categorization based on actual proficiency levels derived based on an IRT analysis (or its approximation through the overall pre-test score). Grouping the 39 proficiency levels into four groups (low, medium-low, medium-high, high) was regarded as a good compromise between cost (authoring effort) and performance (effectiveness). Using this method, the following four proficiency/knowledge levels were obtained based on the average pre-test score (13.95/39) and standard deviation (3.97): low knowledge (score≤10; n=7), medium-low knowledge (11≤score≤14; n=17), medium-high knowledge (15≤score≤18; n=14), and high knowledge (score≥19; n=11). For instance, students in the medium-low knowledge level had scores within one standard deviation below the average. Of the 49 students who were present for pre-test, 30 participated in training.

Once the knowledge levels were assigned, we proceeded with identifying the concepts that should be targeted during training for each level group. The basic idea was to use the pre-test as a source of identifying concepts that are "green-shoots" (ready to emerge) for students at particular knowledge level. We have two criteria for identifying promising "green shoots" for a particular knowledge-level: students at that level begin to show some understanding (e.g., 10-30% of students at that level answer correctly questions related to a concept) and students at higher levels master it (e.g., >80% of the students show profficiency). Both criteria are important because there may be misleading "green shoots." Misleading "green shoots" are concepts that seem to emerge at one knowledge level (k; i.e., 10-30% of students answer correctly questions related to a concept) and are still in an emerging state (instead of becoming fully-grown concepts) for students at the higher-up level (k+1). We conclude that such

green-shoots are not yet ready for "full-growth" for students at level k because students at the immediately higher level (k+1) are still struggling with such concepts.

Once we detected the ready-for-growth "green shoots" for a knowledge-level, appropriate instructional tasks were developed aiming at exposing students to the emerging concepts. There is one exception for the highest knowledge level for which there is no immediately higher level. That is, the second criterion of selecting concepts already mastered by students at the immediately higher knowledge level cannot be applied. In this case, we simply selected concepts with the highest learning potential.

3 Experiment and Results

As already mentioned, students attending a college-level conceptual Physics course were recruited for this experiment. This was an introductory course opened to all college students. The course provided the pre-requisite kind of training that seems to be important for experiments of the type we are describing here. Subjects were randomly assigned to one of the two training conditions: micro-adaptive-only vs. fully-adaptive.

Condition 1 (Micro-adaptive Only). In this condition students interacted with a dialogue-based ITS that used a fixed, predefined set of instructional tasks for all students. That is, there was a one-size fits all approach in terms of adapting instructional tasks to students. The set of predefined tasks included two tasks associated with each of the four knowledge levels defined for the other condition (uniform selection of tasks from all four knowledge levels) plus one additional task selected at random for a total of nine tasks (the number of tasks is the same in both conditions). Once working on a task (problem solving), students were scaffolded as needed through hints in the form of increasingly informative questions. That is, there was micro-adaptation.

Condition 2 (Fully-Adaptive: Macro- and micro-adaptive): In this condition students interacted with the fully adaptive system. The system would categorize students to different levels of understanding based on their pre-test score and then select appropriate tasks that were deemed most conducive of learning at that level of understanding. Tasks were selected for each knowledge level using the data-driven method presented earlier. A total of nine tasks were selected for each knowledge level. Once a task was selected for the students to work on, the micro-adaptation within the task was identical to the micro-adaptation in the micro-adaptive only condition.
The distribution of students into the four knowledge levels was: (Low=2, MediumLow=5, MediumHigh=5, High=2) for the Fully-Adaptive condition and (Low=5, MediumLow=3, MediumHigh=7, High=1) for the Micro-Adaptive condition.

Procedure. After signing a consent form, students took a pre-test under supervision. Students were all present in the same room and were given the pre-test at the same time (on paper). After they took the pre-test (39 multiple choice questions), students were given the opportunity to sign up for free tutoring sessions. Students who chose to participate were given extra credit in the course. Students participated in training

sessions in a lab in small groups. Each student individually interacted with the tutoring system over the Internet from a personal computer. Each training session was about 1.5-hour long and consisted of approximately 1-hour of training (9 Physics problems) followed by a 0.5-hour for a post-test. There was a time span of about 3 weeks between the time students took the pre-test and the time they participated in training (and the post-test). Pre-test and post-test were identical.

Results. A number of 30 students participated in the training experiment with 16 of them in the micro-adaptive-only condition and 14 of them in the fully-adaptive condition. There was no significant difference in pre-test scores (percentage correct on the test) between the two conditions (t[28]=-.343, p=.734). A mixed ANOVA analysis was conducted with a pre-post-test within-subjects variable and the condition as a between-subjects variable. The ANOVA revealed a significant test*condition interaction (F(1,28)=6.793; p=0.015; see Figure 2). Adjusted post-test scores were compared between conditions by running an ANCOVA with the pre-test scores as covariate. A significant difference was found (F(1,27)=11.974; p=.002). A pre-post test comparison, revealed that the fully-adaptive condition had an effect size of (Cohen's) d= 0.786, r=0.366 (computed using means and pooled standard deviations). This is as good as human tutors. VanLehn (2011) reported an average human tutor effect of d= 0.79 (across many domains).

4 Conclusions

The positive results of our study in favor of macro-adaptivity indicate that improvements in this area hold the promise of increasing the effectiveness of tutoring systems beyond the interaction plateau if coupled with advanced tutorial tactics that boost micro-adaptation.

One weakness of our method stems from the IRT-style analysis based on which we defined our knowledge levels. A standard IRT analysis treats each wrong answer, i.e. distractor in a multiple-choice question, on equal footing. There is plenty of evidence that students of different proficiency levels react differently to different distractors (Dedic, Rosenfield, & Lasry, 2010). We will address this issue in order to further improve the level of macro-adaptivity by exploring recent advances proposed by the science education research community, e.g. learning progressions (Rus et al., 2013), and using polytomous IRT analysis.

Acknowledgments. This research was supported by the Institute for Education Sciences (IES) under award R305A100875 to Dr. Vasile Rus. All opinions and findings presented here are solely the authors'.

References

1. Alonzo, A.C., Steedle, J.T.: Developing and assessing a force and motion learning progression. Science Education 93, 389–421 (2009)
2. Brusilovsky, P.L.: A Framework for Intelligent Knowledge Sequencing and Task Sequencing. In: Frasson, C., Gauthier, G., McCalla, G.I. (eds.) ITS 1992. LNCS, vol. 608, pp. 499–506. Springer, Heidelberg (1992)

3. Dedic, H., Rosenfield, S., Lasry, N.: Are All Wrong FCI Answers Equivalent? In: Proceedings of the Physics Education Research Conference, Portland, Oregon, July 21-22 (2010)
4. Hestenes, D., Wells, M., Swackhamer, G.: Force concept inventory. Phys. Teach. 30, 141–158 (1992)
5. Evens, M., Michael, J.: One-on-One Tutoring by Humans and Computers. Lawrence Erlbaum Associates, Inc. (2006)
6. Graesser, A.C., VanLehn, K., Rose, C.P., Jordan, P., Harter, D.: Intelligent tutoring systems with conversational dialogue. AI Magazine 22(4), 39–41 (2001)
7. Lehman, B., D'Mello, S.K., Strain, A.C., Gross, M., Dobbins, A., Wallace, P., Millis, K., Graesser, A.C.: Inducing and tracking confusion with contradictions during critical thinking and scientific reasoning. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 171–178. Springer, Heidelberg (2011)
8. Rus, V., D'Mello, S., Hu, X., Graesser, A.C.: Recent Advances in Conversational Intelligent Tutoring Systems. AI Magazine 34(3), 42–54 (2013)
9. Sabo, K.E., Atkinson, R.K., Barrus, A.L., Joseph, S.S., Perez, R.S.: Searching for the two sigma advantage: Evaluating algebra intelligent tutors. Computers in Human Behavior 29(4), 1833–1840 (2013)
10. VanLehn, K.: The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education 16, 227–265 (2006)
11. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. International Journal of Artificial Intelligence and Education 15(3) (2005)
12. VanLehn, K., Jordan, P., Litman, D.: Developing pedagogically effective tutorial dialogue tactics: Experiments and a testbed. In: Proceedings of SLaTE Workshop on Speech and Language Technology in Education (ISCA Tutorial and Research Workshop) (2007)
13. VanLehn, K.: The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. Educational Psychologist 46(4), 197–221 (2011)
14. Wang, J., Bao, L.: Analyzing Force Concept Inventory with Item Response Theory. Am. J. Phys. 78(10), 1064–1070 (2010)

An Evaluation of Self-explanation in a Programming Tutor

Amruth N. Kumar

Ramapo College of New Jersey, Mahwah, USA
amruth@ramapo.edu

Abstract. A controlled study was conducted *in-natura* to evaluate the effectiveness of presenting passive self-explanation questions in a problem-solving tutor on code-tracing. Data was collected from multiple institutions over three semesters using a tutor on selection statements: fall 2012-fall 2013. ANOVA and ANCOVA were used to analyze the collected data. After accounting for the additional time provided to test group students to answer self-explanation questions, test group was found to fare no better than control group on the number of concepts practiced, the pre-post change in score or the number of practice problems solved per practiced concept. It is speculated that this lack of difference might be attributable to self-efficacy issues, and that the features of tutors found to be effective *in-vivo* might need self-efficacy supports to also be effective *in-natura*.

Keywords: Self-explanation, Programming tutor, Evaluation, Self-efficacy.

1 Introduction

Self-explanation, i.e., the constructive task of explaining to oneself has been studied in depth. One early study found that students develop a deeper understanding of declarative knowledge from expository text when self-explanation is elicited [6]. Another study found that self-explanation improves problem-solving skills when students are prompted to spontaneously generate self-explanations while studying worked-out examples [7]. An early meta-study found that when given by rather than given to the student, elaborate explanation is positively related to individual achievement [16]. These studies of self-explanation informed the current work, in which, *problem-solving* tutors that traditionally gave elaborate explanations to the student were extended by having them *elicit* self-explanation from the student when they present step-by-step explanation of the correct answer, as is done in *worked-out* examples (e.g., [3]).

Self-explanation has been facilitated through typed text (e.g., [13]) or verbal protocols that are manually coded for analysis (e.g., [10]). Many studies have used natural language for self-explanation, and various approaches have been used to categorize and analyze natural language self-explanations (e.g., [12]). Since self-explanations in natural language can be ambiguous, drop-down menus (e.g., [2,15]) have been used as a more objective and unambiguous mechanism for eliciting self-explanation in tutors.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 248–253, 2014.

Inasmuch as the student selects rather than constructs self-explanation, drop-down menus are a *passive* self-explanation mechanism. Nevertheless, one study found that active self-explanation did not lead to better overall learning than passive self-explanation [1].

The effectiveness of providing passive self-explanation questions in a problem-solving tutor on computer programming was studied in the current work. As compared to earlier studies which were conducted *in-vivo* or *in-ovo,* this study was conducted *in-natura.* Some of the conditions of the *in-natura* setup that differentiate this study from earlier studies conducted in the classroom [2,14,15] or laboratory [5] are:

- Students used the web-based tutor usually on their own time, and not in a structured class environment where the activity would have displaced some other structured classroom activity. So, students spent their own discretionary time to use the tutor. This might motivate students to minimize the time they spend working with the tutor.
- Students usually used the tutor unsupervised. So, the seriousness with which they engaged in the tutoring activity was internally rather than externally driven, i.e., the primary driver of their engagement was their self-efficacy [4].
- Students usually used the tutor as an optional supplement to their course, or as an assignment in the course. When they used it as an assignment, often, they received credit for completing the tutoring activity, not for the score they received during the tutoring activity. This might incentivize completion over excellence.

Given these incentives and constraints, the purpose of this study was to evaluate whether and how much self-explanation questions would affect the learning of students *in-natura*.

2 Evaluation

A tutor on selection statements (`if` and `if-else`) was used for this study. The tutor covers 9-12 concepts depending on the programming language (Java/C++/C#). The tutor presents code-tracing problems on these concepts, i.e., in each problem, it presents a program containing a selection statement and asks the student to identify its output. If the student submits an incorrect solution, the tutor presents feedback including step-by-step explanation of the correct execution of the program in the fashion of a fully worked-out example.

Self-explanation questions are presented embedded in the step-by-step explanation presented after the student submits an incorrect solution. Each self-explanation question is a blank in the step-by-step explanation that the student must fill by selecting the correct answer from a drop-down menu. The questions deal with the semantics of the program, e.g., the value of a variable, the line to which control is transferred during execution, etc. The questions are independent of each other, but answering them requires the student to closely read the step-by-step explanation/worked out example and understand the behavior of the program in question. As such, they prompt the

learner to generate missing content information, as recommended for the design of self-explanation questions [9].

So as not to overwhelm the student, the tutor limits the number of self-explanation questions per problem to three. The student is allowed as many attempts as needed, but must answer the current question correctly before proceeding to the next question, and must answer all the questions correctly before proceeding to the next problem. For controlled evaluation, a version of the tutor was used that did not present any self-explanation questions. This version of the tutor allowed the learner to advance to the next problem as soon as it displayed step-by-step explanation of the current problem.

2.1 Protocol

The tutor was configured to administer pre-test-practice-post-test protocol:

- **Pre-test:** Students solved a pre-test that contained one problem per concept. If they solved it partially or incorrectly, they received feedback, including explanation of the correct answer, as could be found in a worked-out example. During this explanation, test group was required to answer three self-explanation questions whereas control group was not presented any self-explanation questions.
- **Adaptive practice:** Students solved problems on only those concepts on which they had solved pre-test problems incorrectly. On each such concept, they solved problems until their average score on the concept exceeded a minimum percentage (usually 60%). After each problem, they received feedback that explained the correct answer. Again, during this feedback, test group was required to answer three self-explanation questions whereas control group was not presented any self-explanation questions.
- **Post-test:** Students solved problems on only those concepts on which their average score exceeded the pre-set minimum during practice session.

The entire session was limited to 30 minutes for control group; test group was allowed 40 minutes to account for the time needed to answer self-explanation questions.

The concepts on which a student solved the problem incorrectly during pre-test, scored the minimum percentage correctness during practice and solved a post-test problem are called *practiced concepts.* Each practiced concept on which a pre-post increase in score is observed is also a *learned concept.* For analysis purposes, the number of problems solved, the score per problem and the time spent per problem during pre-test, practice and post-test were considered on all the concepts as well as only the practiced concepts. Note that since self-explanation was elicited only when a student solved a problem incorrectly, test group students were guaranteed to have answered self-explanation questions during pre-test on all the *practiced concepts*.

2.2 Data Collection and Analysis

Controlled evaluation of selection tutor was conducted *in-natura* over three semesters: fall 2012-fall2013. The selection tutor was made available over the web. Adopting schools were randomly assigned to control or test group each semester. When a

student used the tutor multiple times, data from only the first time the student solved all the pre-test problems was considered for analysis. Since self-explanation questions were presented only when a student incorrectly solved a problem, all the students who had scored 100% on the pre-test were eliminated. After this elimination, 395 students remained in the control group and 335 students in the test group.

The mean number of concepts practiced by control group was 1.62, and by test group was 1.78. However, since control group was allowed 30 minutes to practice with the tutor and test group was allowed 40 minutes, univariate analysis of the number of concepts practiced was conducted with self-explanation as the fixed factor and total time spent as the covariate. The difference between the two groups was found to be significant [$F(2,597) = 62.207$, $p < 0.001$]: accounting for the extra time allowed, control group practiced 1.72 ± 0.11 concepts whereas test group practiced 1.662 ± 0.12 concepts. Therefore, test group practiced significantly fewer concepts than control group.

No significant difference was found in the average score per pre-test problem between control and test groups [$F(1,729) = 1.018$, $p = 0.313$]. So, the two groups were equivalent. Test group spent significantly more time per pre-test problem than control group [$F(1,729) = 23.024$, $p < 0.001$]: 88.39 ± 5.4 seconds for test group versus 70.82 ± 4.9 seconds for control group. This was to be expected since test group had to answer self-explanation questions every time they incorrectly solved a pre-test problem.

Since the number of practice problems solved depended inversely on the pre-test score and directly on the total time allowed, univariate analysis of the number of practice problems solved was conducted with self-explanation as the fixed factor and both pre-test average score and total time as co-variates. A significant difference was found between the two groups [$F(3,663) = 169.166$, $p < 0.001$]: control group solved 7.565 ± 0.56 problems whereas test group solved 6.657 ± 0.62 problems.

No significant difference was found in the average score per practice problem between the two groups. As could be expected, test group spent more time per practice problem than control group since it had to answer self-explanation questions [$F(1,663) = 77.429$, $p < 0.001$]: 100.615 ± 6.26 seconds for test group versus 63.343 ± 5.71 seconds for control group.

Univariate analysis of the number of post-test problems solved was conducted with self-explanation as the fixed factor and total time as a co-variate. Test group solved significantly fewer post-test problems than control group [$F(2,601) = 53.051$, $p < 0.001$]: 1.796 ± 0.14 problems for test group versus 1.844 ± 0.13 problems for control group. Test group also scored significantly fewer points per problem than control group [$F(1,601) = 5.908$, $p = 0.015$]: 0.911 ± 0.02 points for test group versus 0.946 ± 0.02 points for control group. As could be expected, test group spent significantly more time per post-test problem than control group [$F(1,597) = 3.961$, $p = 0.047$]: 60.136 ± 5.94 seconds for test group versus 52.157 ± 5.38 seconds for control group.

Finally, no significant difference was found between the two groups on the pre-post change on practiced concepts. However, test group solved significantly more problems per practiced concept than control group [$F(1,597) = 3.91$, $p = 0.048$]: 0.95 ± 0.02 problems per concept for test group versus 0.92 ± 0.02 problems per concept for control group.

2.3 Discussion

In summary, control and test groups were found to be equivalent based on average score per pre-test problem. Yet, after accounting for the additional time provided to the test group, test group practiced with significantly fewer concepts than control group. Test group solved significantly fewer practice and post-test problems and scored significantly less per post-test problem than control group. The change in learning as measured by the pre-post change in score on practiced concepts was no different between the two groups. However, the rate of learning, as measured by the number of problems solved per practiced concept, was significantly faster for control group.

These results indicate that provision of passive self-explanation did not lead to greater learning *in-natura*. The factors listed earlier differentiating the current study, i.e., that students used the tutor on their own time, unsupervised, and often only for completion credit might explain why results found in *in-vivo* experiments could not be duplicated *in-natura* - the incentive for students is not so much on learning as on completing the task at hand in as little time as possible. But, these factors are more the norm than the exception for the use of tutors, especially in higher education – once a tutor is deployed, the author of the tutor has no control over the conditions under which the tutor will be used by students, unless the tutor is made an integral part of a structured curriculum (e.g., as in Math tutors from carnegielearning.com). Providing self-efficacy supports within the tutor might counter these factors.

The study of self-explanation is not an isolated event – in the past, in a study of reflection in problem-solving tutors, no additional learning gain was found to accrue from the provision of reflection activity after each problem [11]. So, it is speculated that the features of tutors that were found to be effective *in-vivo* might need self-efficacy supports to also be effective *in-natura*. What these self-efficacy supports are, will be the subject of future work.

Acknowledgments. Partial support for this work was provided by the National Science Foundation under grant DUE-0817187**.**

References

1. Aleven, V., Ogan, A., Popescu, O., Torrey, C., Koedinger, K.: Evaluating the effectiveness of a tutorial dialogue system for self-explanation. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 443–454. Springer, Heidelberg (2004)
2. Aleven, V.A., Koedinger, K.R.: An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. Cognitive Science 26(2), 147–179 (2002)
3. Atkinson, R.K., Derry, S.J., Renkl, A., Wortham, D.: Learning From Examples: Instructional Principles from the Worked Examples Research. Review of Educational Research 70, 181–214 (2000)
4. Bandura, A.: Self-efficacy: Towards a unifying theiry of behavioral change. Psychological Review 84(2), 191–215 (1977)

5. Butcher, K.R.: Learning from text with diagrams: Promoting mental model development and inference generation. Journal of Educational Psychology 98(1), 182–197 (2006)
6. Chi, M.T., De Leeuw, N., Chiu, M.H., LaVancher, C.: Eliciting self-explanations improves understanding. Cognitive Science 18(3), 439–477 (1994)
7. Chi, M.T., Bassok, M., Lewis, M.W., Reimann, P., Glaser, R.: Self-explanations: How students study and use examples in learning to solve problems. Cognitive Science 13(2), 145–182 (1989)
8. Davis, R.: Diagnostic Reasoning Based on Structure and Behavior. Artificial Intelligence 24, 347–410 (1984)
9. Hausmann, R.G.M., VanLehn, K.: The Effect of Self-Explaining on Robust Learning. International Journal of Artificial Intelligence in Education 20(4), 303–332 (2011)
10. Hausmann, R.G., Nokes, T.J., VanLehn, K., van de Sande, B.: Collaborative dialog while studying worked-out examples. In: Proceedings of the Artificial Intelligence in Education 2009 Conference (July 2009)
11. Kumar, A.N.: Promoting Reflection and its Effect on Learning in a Programming Tutor. In: Proceedings of 22nd International FLAIRS Conference on Artificial Intelligence (FLAIRS 2009) Special Track on Intelligent Tutoring Systems, Sanibel Island, FL, May 19-21, pp. 454–459 (2009)
12. Lehman, B., Mills, C., D'Mello, S., Graesser, A.: Automatic evaluation of learner self-explanations and erroneous responses for dialogue-based ITSs. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 541–550. Springer, Heidelberg (2012)
13. McNamara, D.S., Boonthum, C., Kurby, C.A., Magliano, J., Pillarisetti, S., Bellissens, C.: Interactive paraphrase training: The development and testing of an iSTART module. In: Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modeling, pp. 181–188. IOS Press (July 2009)
14. McNamara, D.S., Levinstein, I.B., Boonthum, C.: iSTART: Interactive strategy training for active reading and thinking. Behavioral Research Methods, Instruments, and Computers 36, 222–233 (2004)
15. Rau, M.A., Aleven, V., Rummel, N.: Intelligent tutoring systems with multiple representations and self-explanation prompts support learning of fractions. In: Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, pp. 441–448. IOS Press (July 2009)
16. Webb, N.M.: Peer interaction and learning in small groups. International Journal of Educational Research 13(1), 21–39 (1989)

Identifying Thesis and Conclusion Statements in Student Essays to Scaffold Peer Review

Mohammad Hassan Falakmasir, Kevin D. Ashley,
Christian D. Schunn, and Diane J. Litman

Learning Research and Development Center,
Intelligent Systems Program, University of Pittsburgh
{mhf11,ashley,schunn,dlitman}@pitt.edu

Abstract. Peer-reviewing is a recommended instructional technique to encourage good writing. Peer reviewers, however, may fail to identify key elements of an essay, such as thesis and conclusion statements, especially in high school writing. Our system identifies thesis and conclusion statements, or their absence, in students' essays in order to scaffold reviewer reflection. We showed that computational linguistics and interactive machine learning have the potential to facilitate peer-review processes.

Keywords: Peer-review, high school writing instruction, discourse analysis, natural language processing, interactive machine learning.

1 Introduction

Writing is essential to communication, learning, and problem solving. However, poor achievement in high school writing is a major deficiency in the US educational system [1]. There appears to be no single best approach to teaching writing; however, some practices have been shown to be more effective than others.

One of these practices, peer-review of writing assignments, is a commonly recommended technique to improve writing skills, especially in large class settings. Peer-review not only provides students with feedback, it also gives them the opportunity to read essays of other students and improve their reflective and metacognitive skills. Several studies have found that providing feedback leads to improvement in the reviewer's writing [2], especially when the students provide constructive feedback [3] and put effort into the process [4].

While web-based peer-review systems solve logistical challenges of the review process, such as distribution of documents, providing rubrics and review criteria, and supporting successive drafts, they are still far from optimal [5]. In particular, reviewers may not focus on the core aspects of the text being evaluated [6]. In argumentative writing, a thesis statement plays a pivotal role: it communicates the author's position and opinion about the essay prompt; it anchors the framework of the essay, serving as a hook for tying the reasons and evidence presented and anticipates critiques and counterarguments [7]. The thesis statement thus has a major influence in assessing writing skills [8]. A conclusion reiterates the main idea and summarizes the

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 254–259, 2014.

entire argument in an essay. It may contain new information, such as self-reflections on the writer's position [7]. Since thesis and conclusion statements both play a critical role in the overall argument and share similar linguistic elements, in this paper we focus on automatically identifying these two core aspects.

Advances in computational linguistics enable systems to automatically and quickly analyze large text corpora. Shermis et al. [9] reviewed the features of the three most successful Automated Essay Evaluation (AEE) systems. These systems can analyze certain pedagogically significant aspects of essays as reliably as expert human graders. In particular, Burstein and Marcu [10] presented a machine learning model for detecting thesis and conclusion sentences in students' essays. Later they extended their model into a discourse analysis system as a part of ETS Criterion® software for online essay evaluation [11]. Their model uses lexical, syntactic, and rhetorical features and a complex classification framework to label different discourse elements of the essays like introductory material, thesis statement, topic sentences, and conclusion. Writing Pal (W-Pal) [12], an Intelligent Tutoring System, uses another AEE methodology to offer writing strategy instruction, game-based essay writing practice, and formative feedback to high school writers. It uses the Coh-Metrix AEE [13] to analyze student essays and provide formative feedback.

We hypothesize that AEE techniques can also improve computer-supported peer-review by calling reviewers' attention to particular features of an essay (e.g. thesis or conclusion statements) that deserve comment. Our AEE model is designed to be used as a part of the SWoRD peer-review system [14]. To the best of our knowledge, no one has used AEE techniques to support intelligent scaffolding of peer-reviews. We believe that our system has the potential to combine the strengths of both web-based peer review and automated essay evaluation systems. With an ability to identify thesis statements, the system will scaffold reviewers' consideration of these issues posing such questions as:

- *SWoRD thinks [quoted text] is [pseudonym]'s thesis statement. Do you agree?*
- *SWoRD cannot find a thesis statement for [pseudonym]'s paper. Can you?*
- *Tell [pseudonym] to add a thesis statement. What thesis statement would you recommend?*

Since the papers we assess are mainly the first drafts of high school essays that often lacking in both style and structure, the peer-review context gives us a unique opportunity to evaluate and improve our model in practice. We are planning to use the model in an interactive machine learning [15] framework. Since we use the results of our model to scaffold peer-review, the model's outputs will be evaluated first by the author of the paper, and then by a number of peer-reviewers. We can use these author and peer evaluations as feedback to improve the model, thus reducing the need for post hoc time-consuming manual text annotation. This exclusive advantage will enable the system to assess its performance in action and improve toward the desired behavior.

2 Methodology

It is important that reviewers attend to thesis statements: how well they are articulated and supported, and whether alternative interpretations/viewpoints are considered [16, 17]. We find that many peer reviewers, however, do not attend to thesis statements and focus instead on minor claims or lower level writing issues, even with review

prompts that specifically asked reviewers to comment on the logic of the argument. When students did use the term *thesis* in their reviews, the comments were not always sufficiently specific.

In this study we address two questions: 1) Can computational linguistic methods detect presence/absence of thesis and conclusion sentences in student essays in order to guide peer reviewers (i.e., at the essay level)? 2) How well does the model distinguish candidate thesis or conclusion statements from other sentences (i.e., the sentence level)? We evaluate our model both at the essay level and sentence level and compare the performance with a positional baseline and manually annotated essays.

We used 432 essays from 8 writing assignments in 2 high school courses on cultural literacy and world literature. We divided the essays into two sets, one for training and development purposes including 6 assignment prompts with 326 essays and the other for test purposes including 2 assignment prompts and 106 essays. We used the training set to build our model and extract the most predictive linguistic features of thesis and conclusion statements in student essays. Then we tested the performance of our model on the unseen test set.

Six human judges annotated our essays, with an instruction manual based on the scoring guidelines and sample responses of AP English Language and Composition courses. Each essay was coded by at least two human judges, who were asked to identify sentences that were candidate thesis or conclusion statements and to rate the candidate sentences from 1 to 3 (i.e., 1: vague or incomplete, 2: simple but acceptable, 3: sophisticated), based on criteria in the instruction manual. Table 1 shows the distribution and example sentences in each category.

Table 1. Distribution and example sentences from different ratings categories

Rating (%)	Example	Reason
1- Incomplete (%15)	There are contributing factors of our violent society but there are some possible solutions.	Too vague
2- Simple (%39)	As a result of gender roles in Africa, life for women is extremely challenging.	Does not mention the challenges.
3- Sophisticated (%46)	Including music programs in schools is beneficial because music improves students' academic, social and emotional lives.	Provides different reasons for the central claim.

We used Cohen's Kappa [18] to evaluate the agreement between judges on both sentence level (whether a sentence is a thesis/conclusion statement) and essay level (absence/presence of thesis). Kappa was calculated for all 8 writing assignments. If Kappa fell below 0.6, we asked the judges to review the instruction manual and redo the coding until their agreement was acceptable.

We used an iterative process to find the most predictive features for identifying thesis and conclusion sentences in essays. Starting with 42 basic computational linguistic features inspired by [11], such as positional, syntactic, and cue term features, we used several feature selection algorithms to select the most predictive features. We tried different combinations of the predictive features and also added some semantic and rhetorical structure features to improve the model's accuracy. Finally, we picked 19 features in three categories that were most predictive.

Positional Features: We used 3 positional features: paragraph number, sentence number in the paragraph, and type of paragraph (first, body, and last paragraph). We also used the same positional baseline as [11] in order to compare our results with their model. The positional baseline predicts all sentences in the first paragraph as a *thesis statement* and all sentences within the last paragraph as *conclusion sentences*.

Sentence Level Features: We used a number of sentence level features based on the syntactic, semantic, and dependency parsing of the sentence. Based on our feature selection process, prepositional and gerund phrases are highly predictive of thesis and conclusion sentences. The number of adjectives and adverbs within the sentence is also highly correlated with a sentence being a thesis or conclusion statement. A set of frequent words was also predictive for thesis and conclusion sentences (e.g., "although", "even though", "because", "due to", "led to", "caused"), and we used the number of occurrences of these words in a sentence as a feature in our model.

Essay Level Features: We used 4 essay level features: number of keywords among the most frequent words of the essay, number of words overlapping with the assignment prompt, and a sentence importance score based on Rhetorical Structure Theory (RST) adapted from [19]. Table 2 shows the top 5 most predictive features for each category based on the Gini Coefficient [20] attribute selection method. This method considers the prior distribution of the classes and looks for the largest class in the training set (in our case sentences that are not the thesis) and tries to isolate it from other classes, which is suitable based on the nature of our classification task.

Table 2. Top 5 most predictive features for each category based on Gini Coefficent

Ranking	Thesis	Conclusion
1	Last Sentence	Last Paragraph
2	First Paragraph	Keyword Overlap
3	Common Words	Common Words
4	Keyword Overlap	Number of Adjectives
5	Number of Noun Phrases	Number of Noun Phrases

3 Results and Discussion

After a data cleaning and pre-processing step, we created feature vectors for all of the sentences in the training set essays. Our target class had 3 labels: "thesis", "conclusion", and "other". We considered sentences rated 2 and 3 as thesis and conclusion statements and put the ones rated 1 (incomplete) into the "other" category. We evaluated our model on two levels: sentence level and essay level, and compared its performance against the positional baseline and human annotated data.

We used 3 classifiers in RapidMiner [21] in order to develop the sentence level models: Naïve Bayes, Decision Tree, and Support Vector Machine (SVM). We used 10-fold essay stratified cross validation in order to evaluate our models on sentence level. In order to evaluate the models on essay level, we aggregated the results of the sentence level model in order to predict whether an essay contains a thesis/conclusion statement or not. Table 3 shows the performance of the 3 classifiers based on average Precision (P), Recall (R), and F-measure (F) among all 10 rounds of cross-validation. We use F, the harmonic mean of P and R, as our main performance evaluation metric.

Table 3. Average performance of 3 models and the positional baseline on development set

	Thesis			Conclusion			Essay		
Classifier	P	R	F	P	R	F	P	R	F
Positional Baseline	0.53	0.89	0.50	0.51	0.89	0.46	0.61	0.78	0.54
Naïve Bayes	0.62	0.76	0.68	0.57	0.72	0.62	0.71	0.66	0.67
Decision Tree	0.75	0.68	0.71	0.62	0.43	0.51	0.75	0.71	0.73
SVM	0.85	0.66	0.74	0.67	0.41	0.51	0.69	0.64	0.66

In order to indicate how well the models generalize to new essays, we evaluated our models on an unseen test set. Table 4 shows the performance of 3 models.

Table 4. Average performance of 3 models and the positional baseline on unseen test set

	Thesis			Conclusion			Essay		
Classifier	P	R	F	P	R	F	P	R	F
Positional Baseline	0.58	0.88	0.57	0.58	0.84	0.55	0.58	0.84	0.55
Naïve Bayes	0.70	0.79	0.74	0.65	0.69	0.67	0.63	0.65	0.64
Decision Tree	0.82	0.84	0.83	0.49	0.75	0.59	0.75	0.73	0.74
SVM	0.82	0.65	0.72	0.60	0.54	0.56	0.62	0.58	0.60

The results show that all three models outperform the positional baseline. While the SVM classifier had the best precision on both development and test set at the sentence level, the Decision Tree classifier achieved higher recall and better overall performance at the essay level. Since we are not using the same training and test set as in [11], it is not valid to compare the exact value reported for P, R, and F. However, because we use the same positional baseline, and the results of the baseline can be considered as a rough estimate of the quality of the essays, we can compare the systems in terms of improvement over the baseline. In the thesis detection category, their highest reported improvement (regarding F) over the positional baseline is 0.22 while our best improvement is 0.24 on the development set and 0.26 on the unseen test set. In the conclusion detection category, their highest reported improvement is 0.23 while our best improvement is 0.16 development set and 0.12 on the unseen test set. In general, we have low performance in the conclusion category because the essays in our training set are first drafts of writing assignments and the students tend to spread the summary of their arguments across multiple sentences and our current model only works on the sentence level.

In conclusion, our study shows that even with a relatively small corpus of essays, a computational linguistic model can identify core aspects of students' essays. Our first priority was to detect the presence of thesis or conclusion statements within the student essays to provide instant feedback to authors upon submission. The second priority was to identify the particular sentences, to direct reviewers' attention so that they focus some comments on how well the author has framed and supported his/her argument.

Our next step is to embed our model into the SWoRD peer review system and evaluate its impact on the quality of student reviews. The peer-review nature of SWoRD gives us a unique opportunity benefit from both author and peer feedbacks in order to evaluate and refine our model while being used. We also plan to extend the model to detect other core elements of student essays such as topic sentences and supporting materials in order to provide feedback and scaffolding.

Acknowledgements. This research is supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120370 to the University of Pittsburgh. The opinions expressed are those of the authors and do not represent the views of the Institute or the U.S. Department of Education.

References

1. National Center for Education Statistics, The Nation's Report Card: Writing, Institute of Education Sciences, US Department of Education, Washington, D.C. (2012)
2. Sadler, P., Good, E.: The impact of self-and peer-grading on student learning. Educational Assessment 11(1), 1–31 (2006)
3. Wooley, R., Was, C., Schunn, C., Dalton, D.: The effects of feedback elaboration on the giver of feedback. Paper presented at the 30th Annual Meeting of the Cognitive Science Society (2008)
4. Cho, K., Schunn, C.: Developing writing skills through students giving instructional explanations. In: Stein, Kucan (eds.) Instructional Explanations in the Disciplines. Springer, NY (2010)
5. Goldin, I.M., Ashley, K., Schunn, C.D.: Redesigning Educational Peer Review Interactions Using Computer Tools: An Introduction. Journal of Writing Research 4(2), 111–119 (2012)
6. Hansen, J., Liu, J.: Guiding principles for effective peer response. ELT J. 59(1), 31–38 (2005)
7. Durst, R.: Cognitive and Linguistic Demands of Analytic Writing. Research in the Teaching of English 21(4), 347–376 (1987)
8. National Assessment of Educational Progress, Writing Framework for the, National Assessment of Educational Progress (2011)
9. Shermis, M.D., Burstein, J., Higgins, D., Zechner, K.: Automated essay scoring: Writing assessment and instruction. International Encyclopedia of Education 4, 20–26 (2010)
10. Burstein, J., Marcu, D.: A machine learning approach for identification thesis and conclusion statements in student essays. Computers and the Humanities 37(4), 455–467 (2003)
11. Burstein, J., Marcu, D., Knight, K.: Finding the WRITE stuff: Automatic identification of discourse structure in student essays. IEEE Intelligent Systems 18(1), 32–39 (2003)
12. Roscoe, R.D., McNamara, D.S.: Writing pal: Feasibility of an intelligent writing strategy tutor in the high school classroom. Journal of Educational Psychology 105(4), 1010 (2013)
13. Crossley, S.A., McNamara, D.S.: Understanding expert ratings of essay quality: Coh-Metrix analyses of first and second language writing. International Journal of Continuing Engineering Education and Life Long Learning 21(2), 170–191 (2011)
14. Cho, K., Schunn, C.D.: Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. Computers & Education 48(3), 409–426 (2007)
15. Fails, J.A., Olsen Jr, D.R.: Interactive machine learning. In: Proceedings 8th International Conference on Intelligent User Interfaces, pp. 39–45 (2003)
16. De La Paz, S., Graham, S.: Explicit teaching strategies, skills and knowledge: Writing instruction in middle school classrooms. Journal of Educational Psychology 94(4), 687–698 (2002)
17. Durst, R., Laine, C., Schultz, L.M., Vilter, W.: Appealing Texts The Persuasive Writing of High School Students. Written Communication 7(2), 232–255 (1990)
18. Fleiss, J.L., Cohen, J., Everitt, B.S.: Large sample standard errors of kappa and weighted kappa. Psychological Bulletin 72(5), 323 (1969)
19. Marcu, D.: Discourse trees are good indicators of importance in text. Advances in Automatic Text Summarization, 123–136 (1999)
20. Kakwani, N.: On a class of poverty measures. Econometrica: Journal of the Econometric Society, 437–446 (1980)
21. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 935–940 (2006)

Can Diagrams Predict Essay Grades?

Collin F. Lynch[1], Kevin D. Ashley[1], and Min Chi[2]

[1] ISP, LRDC, and School of Law University of Pittsburgh, Pittsburgh, Pennsylvania
collinl@cs.pitt.edu, ashley@pitt.edu
http://www.cs.pitt.edu/∼collinl/, http://www.lrdc.pitt.edu/Ashley/
[2] North Carolina State University, Raleigh North Carolina
mchi@ncsu.edu http://www.csc.ncsu.edu/people/mchi

Abstract. Diagrammatic models of argument have grown in prominence in recent years. While they have been applied in a number of tutoring contexts, it has not yet been shown that student-produced diagrams can be used to effectively grade students or predict their future performance. We show that manually-assigned diagram grades and automatic structural features of argument diagrams can be used to predict students' future essay grades, thus supporting the use of argument diagrams for instruction. We also show that the automatic features are competitive with expert human grading despite the fact that semantic content was ignored in automatic processing.

Keywords: Argument Diagrams, Essay Grading, Argumentation, Educational Datamining, Writing, Automatic Grading.

1 Introduction

Argumentation is an essential skill, particularly in scientific domains where students must articulate and defend clear, testable, hypotheses and frame or recharacterize research problems in order to solve them. Argumentation is difficult for novices who often fail to comprehend arguments or formulate coherent new ones. Students' argumentation skills are often masked by their speaking and writing abilities, or lack thereof, which can limit the effectiveness of expert assessments and peer review. Despite this, argumentation is not always taught explicitly, even in domains such as law where its importance is widely acknowledged. Argumentation is also a challenging domain for AI as real-world arguments are open-ended, typically presented orally or as written text, rely on domain-specific conventions, and are often largely implicit. Thus argumentation presents unique and important challenges for Intelligent Tutoring Systems (ITSs).

Diagrammatic models of argument have been growing in prominence in recent years as theoretical models, practical tools, and educational interventions. The models make argument schema explicit, reifying the essential components and the structured relationships between them as a graph. This reification both makes the structure *salient* and imposes productive *constraints* on novices [11]. This unfamiliar structure, however, can be unfamiliar and challenging to master,

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 260–265, 2014.

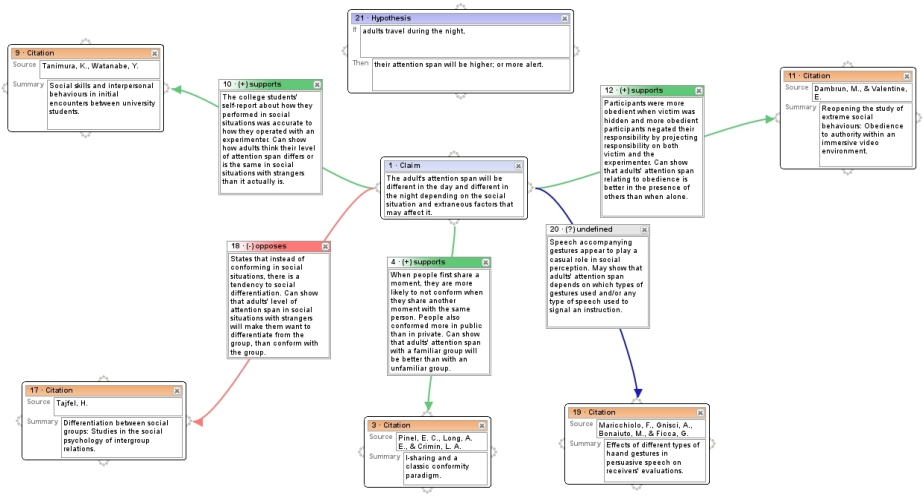


Fig. 1. A segment of a student-produced LASAD diagram representing an introductory argument. It contains a central *claim* node surrounded by *citation* nodes. The isolated node is a *hypothesis* that has not been integrated into the argument.

thus imposing additional cognitive load which can, in turn, inhibit performance [10]. Equally importantly, argument diagrams are amenable to computer processing. Making the structure of the argument explicit enables programmatic assessment and feedback. Argument diagrams have been used in a variety of domains including science [11], law [7], and philosophy [2]. A sample argument diagram of the type used in this study is shown in Figure 1.

While argument diagrams have shown some success in tutoring contexts their overall performance has been mixed (see [9]) and important open questions remain. In particular, it has not yet been shown that student-produced argument diagrams are *empirically-valid*. That is, we have not yet shown that the diagrams can be graded and that the features of those diagrams can be used to predict subsequent performance on natural argumentation tasks such as essay writing. Some prior studies (e.g. [1]) have included qualitative analyses of existing diagrams but that has not been connected to subsequent student performance. In more recent work we have shown that some *a-priori* features of student diagrams (e.g. incorrect arcs) can be used to predict students' argument comprehension [6]. That work, however, focused solely on note-taking diagrams where students were annotating a shared example and did not consider their ability to make novel arguments. In subsequent work we showed that general features of student-produced arguments (e.g. size, length of summative text) could be used to predict subsequent assignment grades. Those grades, however, reflected criteria such as students' presentation and the depth of their background research as well as argument quality. Nor did the study involve grading the diagrams themselves. Thus while argument diagrams have been used in ITSs, they have been promoted chiefly as pragmatic or *effective* interventions that improve student

performance, not *diagnostic* ones. Much like a cricket player cross-training with a soccer game, the practice is helpful but doesn't show off your bowling.

This question of diagnosticity is important, however, both for theoretical and practical reasons. If one of the primary benefits of argument diagramming is the reification of argument structures then the diagram should reflect natural practice. If, however they are not diagnostic, then explicit scaffolding is not a useful explanation. Similarly, if the diagrams are not diagnostic then it will be difficult to convince often skeptical domain experts to use them in place of traditional representations. Moreover, if the diagram structure is not diagnostic it is not clear that the skills of argument diagramming are actually *transferable* to more traditional domains. Our goal in the present study is to address these questions by testing whether or not student-produced argument diagrams can be used to predict subsequent essay grades. We will test the following hypotheses:

H_a. Manual diagram grades can be used to predict subsequent essay grades.
H_b. Automatic diagram features can be used to predict subsequent essay grades.
H_c. Feature-based predictions can be competitive with manual grade predictions.

2 Methods

We tested these hypotheses by means of a grading and machine learning study conducted with an exploratory dataset. The data consisted of a set of paired diagrams and essays collected from a course on psychological research methods (RM) held at the University of Pittsburgh in 2011. The diagrams were produced using LASAD and were graded using a set of parallel grading rubrics. We also defined a set of *a-priori* diagram rules that flagged pedagogically-relevant features. We then applied greedy linear regression to induce a set of predictive models connecting diagram features and grades to the essay grades.

LASAD is an online diagramming toolkit that supports complex diagram ontologies including node and arc types, subfields, and optional text links [3]. The ontology used here has 8 types: (nodes) *hypothesis*, *claim*, *citation*, and *current-study*; (arcs) *supporting*, *opposing*, *undefined*, and *comparison*. All contained flexible text fields for semantic information such as explanations of the relationships or citation information. A sample diagram is shown in Figure 1. While LASAD has an optional help system (see [8]) it was not used here.

RM is a threshold course that covers ethics, study design, and analysis. It is subdivided into 9 lab sections. Students in each section are required to complete 2 empirical research projects. Each section collaborates on the general study design and data collection. Students author their research reports independently or in teams of 2-3. The reports follow a clear pattern. The students are instructed to present their overall argument in the *introduction* section stating their general research question, hypothesis, claims, and citing relevant work. The subsequent sections are expected to support this basic structure. In non-study years the students are given lectures on hypothesis formation and selection of relevant citations but are not always given explicit instruction in argument formation. That is done implicitly through readings and discussion.

The study was integrated into the first writing assignment. Students were given an introductory lecture on argumentation, argument diagrams, and LASAD. They were then tasked with reading 1-3 published research papers and diagramming the arguments found using LASAD. They then used LASAD to diagram their own argument before writing their essays. Diagramming began in class and continued as a homework assignment with students submitting the final diagram and essay for grading. Further details may be found in [4].

The diagrams and essays were graded by an independent grader using a pair of parallel grading rubrics, one for diagrams and the other for essays. The grader had served as a TA in the course in 2012 where LASAD was used again. The rubrics each contained 14 questions, 11 of which focused on specific features of the arguments such as the use of citations and the quality of the hypothesis. The rest focused on the *gestalt* features of coherence, persuasiveness, and overall quality. 13 were graded on a scale of -2 to 2 in $\frac{1}{2}$ point increments. *G/E.14 (Arg-Quality)* was graded on a scale of -5 to 5 in $\frac{1}{2}$ increments given its broader scope. These scores were normalized to the range of 0 to 1 for analysis.

We tested the inter-grader reliability of the rubrics in a separate study [4]. In that study we found that suitably-trained graders can achieve statistically-significant or marginally-significant agreement on all of the diagram grades and most of the essay grades. In the present study we focused on the 5 features for which both criteria had statistically-significant agreement. 4 of these were specific criteria: (*E.01 (RQ-Quality)*) the quality of the research question; (*E.04 (Hyp-Testable)*) whether or not the hypothesis is *testable*; (*E.07 (Cite-Reasons)*) whether or not the author explains the relevance of the cited works; and (*E.10 (Hyp-Open)*) whether or not the author defends the novelty of the research hypothesis. The remaining question, *E.14 (Arg-Quality)*, addressed *gestalt* quality.

In other diagram-based systems such as LARGO [7], students are provided with automated advice driven by *a-priori* rules that detect violations of an ideal argument model or assignment-specific constraints. In this study we defined a set of 77 diagram features that we use for basic evaluation. 34 of these features were simple general features of the type examined in [5] such as the order and size of the diagram. The remaining 43 features were complex features that detect important components of the argument, such as pairs of counterarguments, and violations of argument constraints, such as claims without supporting citations.

We developed five predictive models for each essay question: $M_{baseline}$ is a static model that guesses the most common grade. M_{direct} is a simple linear model of the form $E_i = \alpha_i + \beta_i G_i + \epsilon$ that predicts each essay grade from the corresponding graph grade. M_{grade}, $M_{feature}$, and $M_{combined}$ are linear models that predict the essay grade based upon a subset of the diagram grades, diagram features, or both. These were induced via a two-pass process that first eliminates multicollinear features and then iteratively constructs predictive models based upon the Root Mean Squared Error (RMSE). RMSE is an empirical measure of model error calculated under cross-validation. RMSE gives the absolute value of the expected error of each prediction. The candidate models were selected using a greedy hill-climbing approach. They were trained using least-squares regression with RMSE scores

calculated using 10-fold cross-validation with balanced random assignment. The final RMSEs below were calculated via leave-one-out cross-validation. For more details on the algorithm see [4].

3 Results

We collected and graded 105 unique diagram-essay pairs. 74 were authored by a team, 31 by individuals. The model performance is shown in Table 1. On every question $M_{combined}$ outperformed $M_{feature}$ which outperformed M_{grade}. M_{grade} met or beat M_{direct} which beat $M_{baseline}$. On question *E.10*, for example, the baseline RMSE was 0.463, or 1.8 points out of 5. M_{direct} and M_{grade} beat $M_{baseline}$ by 0.12, while $M_{combined}$ beat it by 0.152 or more than $\frac{1}{2}$ a point out of 5. On question *E.14* $M_{combined} < (M_{grade} \approx M_{feature}) < M_{direct} < M_{baseline}$ with $M_{combined}$ beating the baseline by 0.043 or almost $\frac{1}{2}$ a point out of a range of 11. Therefore both the expert grades (M_{direct} & M_{grade}) and diagram features ($M_{feature}$) were better predictors of students' subsequent essay grades than the baseline model $M_{baseline}$ while the combined models ($M_{combined}$) beat the others on every question.

4 Analysis and Conclusions

Proponents of argument diagrams, including ourselves, have long argued that they can be used for both *effective* and *diagnostic* tutorial interventions. Our goal in this study was to determine whether or not student-produced argument diagrams can be used to predict subsequent essay grades. In this work we showed: that manual diagram grades (M_{direct} & M_{grade}) were better predictors of the essay grades than the baseline model ($M_{baseline}$) thus validating hypothesis H_a; that models based upon diagram features ($M_{feature}$) also beat $M_{baseline}$ thus validating H_b; and that the grade and feature-based models were competitive ($M_{feature} \leq M_{grade}$) thus validating H_c. This is surprising given that the human grader was able to evaluate the semantic content of the diagram fields while the automatic models did not. Therefore argument diagrams can be used for diagnostic educational interventions and this form of empirical modeling can be applied fruitfully even where natural language understanding is unavailable.

Table 1. RMSE scores for the five predictive models for the essay grades. The scores were calculated using leave-one-out cross-validation.

Question	$M_{baseline}$	M_{direct}	M_{grade}	$M_{feature}$	$M_{combined}$
E.01 (RQ-Quality)	0.344	0.311	0.311	0.29	0.284
E.04 (Hyp-Testable)	0.237	0.232	0.232	0.212	0.202
E.07 (Cite-Reasons)	0.27	0.248	0.245	0.243	0.223
E.10 (Hyp-Open)	0.463	0.339	0.334	0.316	0.311
E.14 (Arg-Quality)	0.245	0.214	0.206	0.207	0.202

Interestingly, while M_{grade} and $M_{feature}$ were competitive, $M_{combined}$ dominated on every problem. Therefore either the semantic content was not used by the grader, contrary to instructions, or it conveyed different information than the diagram structure but conferred no substantive advantage. We plan to address this in future work and to test both the generality of these models and their use in ITSs to support individuals, peers, and instructors. In LARGO, for example, help is provided upon request and students are free to ignore it. Given these results, we plan to test whether help in argumentation should be compulsory for lower-performing students and then faded over time. We also plan to test whether diagnostic models such as these can be used to improve peer review and expert instruction by helping to rank students by skill level, to match appropriate mentors, and to flag students in need of expert guidance.

Acknowledgments. This work was Supported by National Science Foundation Award #1122504, "DIP: Teaching Writing and Argumentation with AI-Supported Diagramming and Peer Review," Kevin D. Ashley PI, Chris Schunn & Diane Litman, co-PIs.

References

1. Chryssafidou, E., Sharples, M.: Computer-supported planning of essay argument structure. In: Proc. of the 5th International Conference of Argumentation (2002)
2. Harrell, M., Wetzel, D.: Improving first-year writing using argument diagramming. In: Knauff, M., Sebanz, N., Pauen, M., Wachsmuth, I. (eds.) Proc. of the 35th Annual Conf. of the Cognitive Science Society, pp. 2488–2493
3. Loll, F., Pinkwart, N.: Lasad: Flexible representations for computer-based collaborative argumentation. Int. J. Hum.-Comput. Stud. 71(1), 91–109 (2013)
4. Lynch, C.F.: The Diagnosticity of Argument Diagrams Univ. of Pittsburgh (2014)
5. Lynch, C.F., Ashley, K.D., Falakmassir, M.H.: Comparing argument diagrams. In: Schäfer, B. (ed.) JURIX 2012: The 25th Annual Conference, vol. 250, pp. 81–90. IOS Press, University of Amsterdam (2012)
6. Lynch, C.F., Ashley, K.D., Pinkwart, N., Aleven, V.: Argument graph classification with genetic programming and c4.5. In: de Baker, R.S.J., Barnes, T., Beck, J.E. (eds.) EDM, pp. 137–146 (2008), `www.educationaldatamining.org`
7. Pinkwart, N., Ashley, K.D., Lynch, C.F., Aleven, V.: Evaluating an intelligent tutoring system for making legal arguments with hypotheticals. International Journal of Artificial Intelligence in Education 19(4), 401–424 (2009)
8. Scheuer, O., Niebuhr, S., Dragon, T., McLaren, B.M., Pinkwart, N.: Adaptive support for graphical argumentation - the LASAD approach. IEEE Learning Technology Newsletter 14(1), 8–11 (2012)
9. Scheuer, O., Loll, F., Pinkwart, N., McLaren, B.: Computer-supported argumentation: A review of the state of the art. International Journal of Computer-Supported Collaborative Learning 5, 43–102 (2010)
10. Shum, S.J.B., MacLean, A., Bellotti, V.M.E., Hammond, N.V.: Graphical argumentation and design cognition. HCI 12(3), 267–300 (1997)
11. Suthers, D.D.: Empirical studies of the value of conceptually explicit notations in collaborative learning. In: Okada, A., Buckingham Shum, S., Sherborne, T. (eds.) Knowledge Cartography, pp. 1–23. Springer (2008)

Toward Automatic Inference of Causal Structure in Student Essays

Peter Hastings[1,⋆], Simon Hughes[1], Anne Britt[2], Dylan Blaum[2], and Patty Wallace[2]

[1] DePaul University, Chicago, Illinois
[2] Northern Illinois University, DeKalb, Illinois

Abstract. With an increasing focus on science and technology in education comes an awareness that students must be able to understand and integrate scientific explanations from multiple sources. As part of a larger project aimed at deepening our understanding of student processes for integrating multiple sources of information, we are developing machine learning and natural language processing techniques for evaluating students' argumentative essays. In previous work, we have focused on identifying conceptual elements of the essays. In this paper, we present a method for inferring the causal structure of student essays. We used a standard parser to derive grammatical dependencies of the essay and converted them to logic statements. Then a simple inference mechanism was used to identify concepts linked to syntactic connectors by these dependencies. The results suggest that we will soon be able to provide explicit feedback that enables teachers and students to improve comprehension.

Keywords: Reading, Argumentation, Natural language processing, Machine learning.

1 Introduction

Recent science and literacy standards are increasing the demand for students to use multiple sources of information to understand explanations for phenomena and to use data to support these explanations. Thus, there is critical need for methods of evaluating students' explanations and argumentative support based on scientifically important criteria (e.g., coherence, completeness, and accuracy).

A scientific explanation, also called a causal chain, is a statement that makes clear how one or more factors lead to an outcome. For example, in Figure 1 below, the to-be-explained outcome is an "increase in recent average global temperatures", and there are two separate initiating factors (fossil fuel consumption and deforestation). It is expected that students need practice to become more

⋆ The assessment project described in this article is funded, in part, by the Institute for Education Sciences, U.S. Department of Education (Grant R305G050091 and Grant R305F100007). The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 266–271, 2014.

facile using an explanation schema to guide both writing and reading. It would be very helpful for teachers to have a tool that supports student practice with feedback to help them develop this explanation schema. As a first step, we are examining whether we can automatically identify the causal structure of student essays in two different scientific domains.

This paper describes previous research done on this task, and then presents more fully the educational context of the current work. Then we describe our ongoing research in using machine learning to identify the conceptual elements of essays, and our initial efforts toward inferring causal structure.

2 Previous Research

Although causal explanations have long been a focus for science education [15,3, for example], very little research has been done to automatically identify causal connections in student essays, but there has been some research with other types of texts. In 1987, Cohen [4] laid out a theoretical framework encompassing the many different challenges that need to be solved to fully understand argumentative discourse. Thirty years later, a SemEval-2007 workshop focused on sentences known to have one of seven different types of relations, including causation [7]. Accuracy in distinguishing betweeen the seven types ranged from 50 to 76%.

More recently, Rink et al. developed a system focused on identifying the presence or absence of a causal relationship within a sentence [12]. They used a graph representation of the sentence and trained a machine learning technique on 700 sentences (30% with a causal relation) to distinguish graphs with and without causal connections. Their best F_1 score was 0.39. This was on news texts rather than student essays but clearly demonstrates the difficulty of the task.

3 Educational Context

To deepen our understanding of students' comprehension processes, we created two document sets describing the causes of two scientific phenomena: global warming and coral bleaching. Each document set was based on a causal model of the scientific phenomenon and used information from reputable websites (e.g. the United States Geological Survey). Each document contributed only a partial causal chain. Students were given a document set and asked to write an essay explaining the phenomenon using specific information from the documents to support their conclusions and ideas. A total of 183 middle (84%) and high school (16%) students wrote essays on the global warming document set, and 105 middle (73%) and high school (27%) students wrote essays on the coral bleaching document set.

As mentioned above, Figure 1 gives a graphical representation of the space of causal connections that students might make from the documents to the conclusion in the global warming domain. Thin black arrows indicate explicit connections made in the documents. Dotted lines indicate implicit connections — inferences that students might make between concepts. Red lines represent

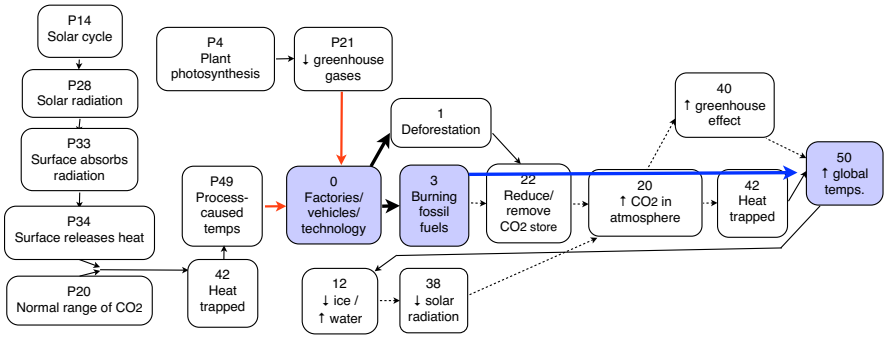


Fig. 1. Causal model with feedback for Global Warming

"counters", for example, "Normal temperature shifts happen *but* our use of cars and factories changes things." This graph also provides an example of how automatic assessment of the essays might be used to provide feedback to students. The thick black arrows mark explicit connections that were identified in the student's essay. The thick blue arrow shows where the student made a causal connection to the conclusion, but skipped some intermediate causal links. For explicit feedback, the student could be shown the graph to provide an indication of what was found and what was missing from her essay. For less guiding feedback, the student could be told that she has identified some links in the causal chain, but has omitted others.

Humans evalated the essays to identify which causes (nodes in Figure 1) were explicitly linked to the target effect (here, increase in global temperatures). Inter-rater reliability was high ($\kappa = 0.85$), and the method was useful in discriminating essays that provided coherent and complete answers. There was a difference in annotation for the two sets of essays. The global warming essays were annotated at the sentence level. Each sentence was associated with a set of codes indicating the concepts and causal connections found in that sentence. The coral bleaching essays were annotated later with a more sophisticated tool (brat.nlplab.org), identifying which specific words in the essay were associated with each concept and connection.

4 Concept Identification

In previous work, we evaluated several different techniques for identifying conceptual material (i.e., the nodes in the graphs) in student essays, including simple pattern matching, latent semantic analysis (LSA), and support vector machines (SVMs) [8,9,10]. In general, we have found that the machine learning approaches do best at identifying high-level claims and specific details about the claims. Student sentences associated with these items tend to bear a striking similarity to

the original texts that they came from.[1] The machine learning techniques have had a much more difficult time identifying conceptual material related to inferences between documents. Examples of these items are rather infrequent in the students' essays (explaining why we need a system like this). They also combine information (and, therefore, words) from different documents and are thus less similar to the original sources [9]. We have recently begun evaluating a new machine learning approach, Deep Learning [13,1,5], which uses multilayer neural networks, but details of this approach are omitted due to space limitations.

5 Inferring Causal Connections

Once the conceptual content of an essay has been identified, the next step required for automatic structure evaluation is to find where the essay makes explicit connections between the concepts. For this step, it is clear that a "bag-of-words" approach would be severely handicapped because it would not be able to take advantage of the critical information provided by the linguistic structure of the text. To capture this structure, we applied the Stanford Compositional Vector Grammar parser [14] from Stanford CoreNLP (v.3.3.1) to tokenize and parse the essays and identify coreference relations [11].

We were particularly interested in taking advantage of the dependencies that the parser identifies in the text [6]. Dependencies are textual relations that are extracted from the parse tree, connecting different components. For example, the sentence, "The fat dog was chased by a cat," produces (among others) dependencies indicating that "fat" is an adjectival modifier for "dog", "dog" is the passive nominal subject of "chased", "cat" is the agent of "chased", and "chased" is the root of the sentence.

To enable inference of causal connections, we transformed the dependencies into clauses in Prolog, because Prolog seems especially well suited for specifying complex constraints. To evaluate the identification of connections independently from the identification of concepts, we started from the human annotations of concepts and connectors,[2] which were also converted into Prolog clauses.

A total of five Prolog rules were used to do the inference. Three of them handle different forms of representation. One of the two main inference rules searches for dependencies between connectors and *causes*, looking at three dependency types. The other rule looks for dependencies between connectors and *results*, looking at 7 types of dependencies.

[1] In fact, 25 – 30% of the student essay sentences had an LSA cosine greater that 0.75 with some sentence from the relevant document set. Ironically, this facilitates our job of classifying the student sentences. The effect on student learning, however, is subtle (analysis forthcoming).

[2] We do not include connectors as concept codes, but they are a critical part of identifying causal relations. Fortunately, students use fairly standard connectors. In coral bleaching, for example, of 134 coded connectors, 32 were "because (of)" and 15 had some variation of "cause". The rest, though less frequent, followed standard conventions.

Using this minimal inference mechanism, we calculated Recall, Precision, and F_1 scores, based on the whether the *inferred* causal connections matched the *annotated* ones. On the coral bleaching essays, the scores were: $Recall = 0.26$, $Precision = 0.59$, and $F_1 = 0.36$. On the 30 (out of 183) global warming essays we have fully annotated, this method achieved $Recall = 0.37$, $Precision = 0.53$, and $F_1 = 0.44$. At this early stage, the results are very encouraging. This technique outperformed the most similar previous research on inferring causal connections (although we did have the advantage of pre-identified concepts and connectors). Also, given that the Precision scores are high relative to the Recall scores, more sophisticated inference rules should be able to find items that our simple rules missed without producing too many false alarms.

6 Conclusions and Future Work

Clearly the work presented here is in its early stages, but the results so far have been extremely encouraging. Even though we have artificially boosted our results by starting with human-annotated concept codings, our very simple mechanism for identifying causal relations has already outperformed previous approaches. We are pursuing several different directions that should bring us closer to our ultimate goal of fully automatic causal relation identification so that we can provide reliable feedback to teachers and students.

With respect to concept classification, greedy sequence classification [2] could be used where a sequential classifier is trained to incorporate the tag it predicted for the previous word when tagging the next word. Neural Network Language Models (NNLMs) have recently become very popular due to their ability to learn a distributed representation for words at the same time as creating a language model to predict the likelihood of a sequence of words [1]. However little work has been done to investigate their use in creating sequential classifiers. An NNLM could be used to create a sequential classifier that predicts the concept tag for the central word in a word window instead of predicting the likelihood of the central word.

Another critical component for identifying causal relations is anaphora resolution. Students often use pronouns to refer to previously mentioned concepts. In the coral bleaching domain, 10% of the identified causal relations involved a pronominal reference. Because we included the human annotations for references in our evaluation, we were able to correctly identify a comparable percentage of causal relations with and without anaphora. As mentioned above, the Stanford CoreNLP parser returns coreference relations in addition to the dependencies. If these are reliable for student essays, they should allow us to successfully automate identification of relations across sentences.

The current inference rules for identifying causal relations are quite simple. It is quite likely that the hit-rate of these rules can be significantly improved by adding more dependencies, although it may well be that additional constraints are necessary to avoid over-generalization. We will also explore the use of machine learning techniques like Rink et al. used to automatically derive new inference

rules [12]. Finally, we are collecting additional student essay data in these and in new scientific explanation domains. This will support cross-domain validation of our techniques, to ensure that they can produce robust results.

References

1. Bengio, Y., Ducharme, R., Vincent, P., Janvin, C.: A neural probabilistic language model. Journal of Machine Learning Research 3, 1137–1155 (2003)
2. Bird, S., Klein, E., Loper, E.: Natural Language processing with Python Analyzing Text with the Natural Language Toolkit. O'Reilly (2009)
3. Chi, M., Roscoe, R., Slotta, J., Roy, M., Chase, C.: Misconceived causal explanations for emergent processes. Cognitive Science 36, 1–61 (2012)
4. Cohen, R.: Analyzing the structure of argumentative discourse. Computational Linguistics 13(1-2), 11–24 (1987)
5. Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Cohen, W., McCallum, A., Roweis, S. (eds.) ICML, vol. 307, pp. 160–167. ACM (2008)
6. de Marneffe, M., Manning, C.: The Stanford typed dependencies representation. In: COLING 2008 Workshop on Cross-framework and Cross-domain Parser Evaluation (2008), `http://nlp.stanford.edu/pubs/dependencies-coling08.pdf`
7. Girju, R., Nakov, P., Nastase, V., Szpakowicz, S., Turney, P., Yuret, D.: Semeval-2007 task 04: Classification of semantic relations between nominals. In: Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval 2007), p. 1318 (2007), `http://acl.ldc.upenn.edu/W/W07/W07-2003.pdf`
8. Hastings, P., Hughes, S., Magliano, J., Goldman, S., Lawless, K.: Text categorization for assessing multiple documents integration, or John Henry visits a data mine. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 115–122. Springer, Heidelberg (2011)
9. Hastings, P., Hughes, S., Magliano, J., Goldman, S., Lawless, K.: Assessing the use of multiple sources in student essays. Behavior Research Methods 44(3), 622–633 (2012)
10. Hughes, S., Hastings, P., Magliano, J., Goldman, S., Lawless, K.: Automated approaches for detecting integration in student essays. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 274–279. Springer, Heidelberg (2012)
11. Recasens, M., de Marneffe, M.C., Potts, C.: The life and death of discourse entities: Identifying singleton mentions. In: HLT-NAACL, pp. 627–633. The Association for Computational Linguistics (2013)
12. Rink, B., Bejan, C.A., Harabagiu, S.M.: Learning textual graph patterns to detect causal event relations. In: Guesgen, H.W., Murray, R.C. (eds.) FLAIRS Conference. AAAI Press (2010)
13. Socher, R., Pennington, J., Huang, E., Ng, A., Manning, C.: Semi-supervised recursive autoencoders for predicting sentiment distributions. In: EMNLP, pp. 151–161. ACL (2011)
14. Socher, R., Bauer, J., Manning, C.D., Ng, A.Y.: Parsing with compositional vector grammars. In: ACL (1), pp. 455–465. Association for Computer Linguistics (2013)
15. White, B., Frederiksen, J.: Causal model progressions as a foundation for intelligent learning environments. Artificial Intelligence 42, 99–157 (1990)

Classroom Evaluation of a Scaffolding Intervention for Improving Peer Review Localization

Huy Nguyen, Wenting Xiong, and Diane Litman

Department of Computer Science,
University of Pittsburgh, Pittsburgh, PA 15260

Abstract. A peer review system that automatically evaluates student feedback comments was deployed in a university research methods course. The course required students to create an argument diagram to justify a hypothesis, then use this diagram to write a paper introduction. Diagram and paper first drafts were both reviewed by peers. During peer review, the system automatically analyzed the quality of student comments with respect to localization (i.e. pinpointing the source of the comment in the diagram or paper). Two localization models (one for diagram and one for paper reviews) triggered a system scaffolding intervention to improve review quality whenever the review was predicted to have a ratio of localized comments less than a threshold. Reviewers could then choose to revise their comments or ignore the scaffolding. Our analysis of data from system logs demonstrates that diagram and paper localization models have high prediction accuracy, and that a larger portion of student feedback comments are successfully localized after scaffolded revision.

Keywords: Peer review, review localization, scaffolding, evaluation.

1 Introduction

While peer review is a promising approach for helping students improve their writing, peer feedback can be of mixed quality. For example, prior work [6,5] has shown that feedback is more likely to be implemented in a revision when the review is *localized*, that is, pinpoints the location of the problem mentioned in the feedback (as shown in the examples in Fig. 1). As a first step towards helping students improve the quality of their feedback, natural language processing and machine learning have been used to build models for automatically detecting whether peer reviews contain localization and other desirable feedback properties [2,11,8,7]. To date, however, such models have typically been evaluated only intrinsically (i.e. with respect to predicting gold standard manual annotations), rather than extrinsically with respect to a real-world task (e.g. being incorporated into a peer review system to improve review quality). In addition, while intrinsic evaluations have shown that a predictive model can yield high accuracy when trained and tested on data from the same peer-review assignment, how the model performs on unseen data sets has not yet been examined. To address these

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 272–282, 2014.

issues, we have conducted both an intrinsic and extrinsic evaluation of review localization in a classroom setting. First, we followed our previous work [11,7] to implement models for predicting localization in comments of paper and diagram reviews, and integrated them into SWoRD [3], a web-enabled peer review system.[1] Next, we designed and implemented a system scaffolding intervention to improve students' use of localization when they provide feedback to each other. In our intervention, scaffolding is triggered whenever a review is predicted to have a ratio of localized comments less than a threshold. Students (as reviewers) can then choose to either revise their review comments or ignore the scaffolding. Finally, we deployed this system in a classroom setting, and evaluated its success from several perspectives. Our results show that for both diagram and paper reviews 1) the localization models predict the absence of localization in reviews with high accuracy, 2) the system scaffolding intervention helps reviewers to revise their feedback to increase localization, and 3) reviewers continue to add localization even after the scaffolding is removed.

2 Related Work

In instructional science, Gielen et al. [1] investigated effects of different peer feedback characteristics and showed that the presence of feedback justification significantly improved writing performance. Nelson and Schunn [6] found that localization in reviews of papers was significantly related to problem understanding, which in turn was significantly related to feedback implementation. Lippman et al. [5] similarly showed that localization was related to the implementation of peer feedback on argument diagrams.

Based on findings such as the above, research in computer science has used natural language processing and supervised machine learning to automatically detect when a free text feedback comment exhibits a desirable quality. Xiong and Litman [11] developed models for predicting localization in peer reviews of written papers, using features derived from a dependency parse tree. Nguyen and Litman [7] developed a localization model tailored to reviews of diagrams rather than papers, by considering common words between review comments and the target diagram.

Similar methods have been used to predict feedback helpfulness label (Yes v. No) [2], helpfulness rating [12], and other measures of review quality [8]. Particularly, we found in our prior work [12] that the percentage of localized comments contributes to improving performance of modeling review helpfulness. In this paper, instead of developing new prediction models, we focus on integrating existing models of review localization into a working peer review system, and evaluating model performance in a classroom deployment.

[1] While it is possible to modify a reviewing interface to have reviewers directly comment upon a paper, such an interface encourages primarily feedback on low-level text issues, and is not good for repeated errors or issues with larger sections of text. Therefore, we focus on encouraging localization in end-comments.

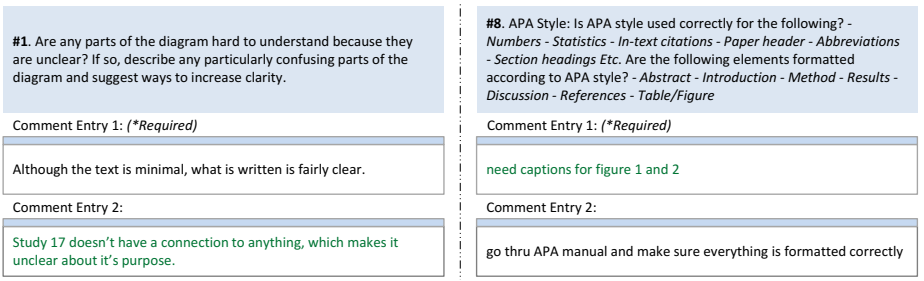


Fig. 1. Examples of localized (in green) and not localized (in black) comments in a diagram review (left) and a paper review (right). Localization cues in the green comments are *"Study 17"* (left) and *"figure 1 and 2"* (right).

Regarding system scaffolding to increase feedback quality, the design of our intervention incorporates techniques from prior work in intelligent tutoring systems. Razzaq and Heffernan [9] compared two approaches for giving hints during tutoring: proactively when students make errors, versus on-demand when students ask for a hint. They found no difference in learning gains for students who did not ask for many hints. Because our students are not trained on feedback localization we do not expect them to know when they need a hint, and thus choose to trigger our scaffolding intervention proactively whenever a student review lacks sufficient localization. In a different context, Kumar [4] showed that when error-flagging was provided during tests on introductory programming concepts, student scores improved. To implement error-flagging, correct student answers were displayed in green and incorrect answers were displayed in red; in addition, no reasons why the answers were incorrect were provided. In our system we will similarly display localized versus not-localized feedback predictions using different colors, to help students identify the problematic comments.

3 Adding Localization Scaffolding to Peer Review

A typical peer review exercise using SWoRD involves three main phases: 1) student authors create first drafts[2], 2) peer reviewers provide feedback[3] on the drafts, and 3) authors revise their drafts to address the feedback. The original version of SWoRD only facilitates the document management and review assignment aspects of peer review. To further enhance the utility of SWoRD, in this paper we add artificial intelligence to the system by integrating the detection and scaffolding of localization into phase 2, using prior models from the literature to predict paper [11] and diagram [7] review localization, respectively.

In our enhanced version of SWoRD, whenever an argument diagram or paper review is submitted, the corresponding review localization model is first used

[2] A draft can be a paper, a diagram, a presentation, etc. depending on the assignment.
[3] Feedback is in the form of written comments along with numerical ratings.

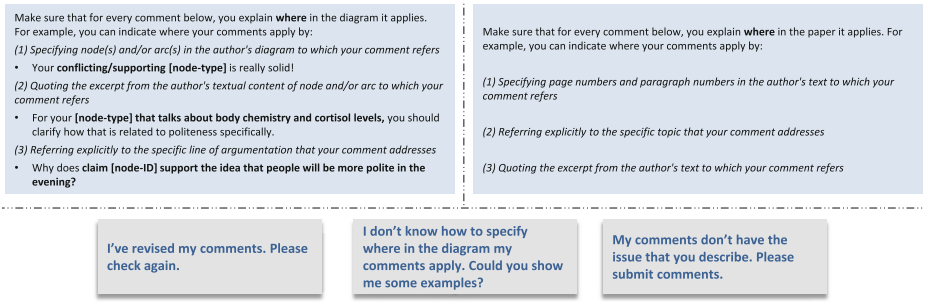


Fig. 2. Scaffolding messages for revising reviews of diagrams (top left) and papers (top right), along with the three responses available to reviewers (bottom)

to predict whether every review comment is localized or not. Fig. 1 shows examples of localized and not-localized comments from both a diagram (left) and paper (right) review, in which comments predicted as localized are highlighted in green. Then, if the submitted review is predicted to have a ratio of localized comments less than a threshold of 0.5[4], the scaffolding intervention will be triggered: the system displays an on-screen message which suggests review revision and provides advice for doing so (see the top of Fig. 2 for diagrams (left) and papers (right)). Finally, the reviewer can choose to revise the review and resubmit, view some model comments, or submit the review without revision (implying disagreement) as indicated by the three buttons at the bottom of Fig. 2. Every revised review then goes through the same localization prediction process.

4 The Peer Review Corpus

Our corpus consists of comments from *diagram* and *paper* reviews, collected from undergraduate Research Method course in psychology at University of Pittsburgh, 2013. In this class, students were asked to first create graphical argument *diagrams* using LASAD [10] to justify given hypotheses. Student argument diagrams were then distributed via SWoRD to 4 randomly assigned peers for reviewing. Student authors could revise their argument diagrams based on peer feedback, then used the diagrams to write the introduction of associated *papers*. Similarly to the diagram review step, student papers were randomly assigned to 4 peer reviewers (potentially different than the diagram reviewers). Finally, after receiving reviews of their papers, authors could revise their papers before final submission. Diagram and paper reviews both consisted of multiple feedback comments written in response to rubric prompts (e.g. #1 and #8 in the top boxes in Fig. 1). Reviewers were required to provide feedback for 5 argument diagram prompts and 8 paper prompts. Each prompt required reviewers to provide one to three comments. The system allows reviewers to edit and resubmit

[4] The threshold was tuned based on data from prior classes.

Table 1. Peer review data statistics. All re-submissions are counted.

	Diagram review	Paper review
Reviewers/Authors	181/185	167/183
Submitted reviews	788	720
Intervened submissions	173	51

Table 2. Localization annotation results

	Diagram review	Paper review
`Localized` comments	449	347
`NOT Localized` comments	718	336

their reviews at any time before the deadline with the same review scaffolding procedure. Table 1 summarizes the dataset.

To support the evaluations described below, we collected all diagram and paper review submissions which triggered a system intervention, as well as their subsequent resubmissions (if any), and then manually coded the collected reviews (both submissions and resubmissions) for the presence of localization in each comment. In addition, since reviewers may edit their submitted reviews without any system intervention, we also collected and coded localization for all reviews where re-submission occurred after a non-scaffolded submission. By pairing each comment with its revision, we aim to evaluate how the system scaffolding impacted reviewer revisions.

Following the localization annotation scheme of [5], a comment is coded as `Localized` if it contains at least one text span indicating where in the target diagram or paper the comment is applied. The comment is coded as `NOT Localized` otherwise. Two annotators independently coded comments of diagram reviews and achieved inter-rater Kappa of 0.8. The two annotators then resolved label disagreements to obtain the final labels used for our evaluations. Another annotator who had Kappa of 0.8 when coding prior paper review data was chosen to code the paper review comments obtained during our experiment. Table 2 summarizes the annotated data used in our analyses.

5 Review Localization Prediction Performance

Our first analysis aims to evaluate both the accuracy of predicting localization at the comment level, and the accuracy of using these predictions to intervene at the review submission level, for both diagram and paper reviews.

At the comment level, we evaluate how well the two review localization models predict the presence of localization compared to the manual annotations. We also compare the models' performance to their corresponding majority-class

Table 3. Localization prediction performance at the comment level

	Diagram review			Paper review		
	Accuracy	F-measure	Kappa	Accuracy	F-measure	Kappa
Baseline	61.5%	0.47	0	50.8%	0.34	0
Model	81.7%	0.82	0.62	72.8%	0.73	0.46

Table 4. Intervention prediction performance at the review submission level

	Diagram review	Paper review
Total scaffolding interventions	173	51
Incorrectly triggered scaffolding interventions	1	0

baselines[5]. Table 3 shows that both localization models substantially outperform their respective baselines. In addition, when comparing these results with the originally reported results for these models (accuracy and Kappa figures of 83.8% and .56 for diagrams [7], and 77.4% and .55 for papers [11], respectively), we see that performance is only slightly degraded in our cross-domain evaluation setting. Our current evaluation setting is more difficult because the localization models were trained prior to our corpus collection while each of the models in the original publications were trained and tested on a single dataset using cross-validation.

At the review submission level, we consider an intervention to be correct when at least one of the comments in a submission is labeled as `NOT Localized`, as reviewers should only think the system incorrectly intervened when all of the comments in a submitted review were indeed localized. As shown in Table 4, the diagram review localization model yielded only one incorrect intervention, while the system never incorrectly intervened when scaffolding a paper review.

In sum, our results show that in a real classroom setting, our models accurately predict localization in the review comments of both diagrams and papers. These comment-level predictions, in turn, are the basis of a system scaffolding intervention that is accurately triggered from a reviewer's perspective.

6 Reviewer Responses to the System Intervention

In this section, we first analyze whether reviewers actually revise their comments in response to the system scaffolding intervention. For those reviews that are indeed revised, we then analyze whether the number of localized comments in fact increases after review revision, and whether revision behavior varies depending on whether the review revision was scaffolded versus unscaffolded.

Reviewer Response Types. A reviewer can respond to the system's scaffolding intervention in one of three ways (recall the buttons shown in Fig. 2):

[5] The majority class is `NOT Localized` for diagram and `Localized` for paper review.

Table 5. Percentage of different types of reviewer responses to first interventions

Response type	`Revise`		`Disagree`		(`View Example`)	
Diagram review	54	48%	59	52%	(5)	(4%)
Paper review	13	30%	30	70%	(1)	(2%)

Table 6. Histogram of responses by true localization ratios in diagram reviews and paper reviews. NA means the bin has no data.

Ratio bin	[0,.1)	[.1,.2)	[.2,.3)	[.3,.4)	[.4,.5)	[.5,.6)	[.6,.7)	[.7,.8)	[.8,.9)	[.9,1)	1
	Diagram reviews										
Tot. responses	12	8	32	5	28	9	16	1	1	NA	1
%`Disagree`	75.0	37.5	50	20	50	77.7	43.7	0	100	NA	100
	Paper reviews										
Tot. responses	3	4	3	4	5	7	12	5	NA	NA	NA
%`Disagree`	100	50	66.7	75	60	85.7	66.7	60	NA	NA	NA

- `Revise`: the reviewer resubmits her review after revising it.
- `View Example`: the reviewer views examples of localized comments, then goes back to the system intervention interface.
- `Disagree`: the reviewer submits her review without revision.

For this paper, we consider only reviewer responses after the system's first intervention for a review.[6] Table 5 shows the percentage of different response types to these first interventions. In addition, as `View Example` is not an action that completes the review activity, the response must be followed by either a `Revise` or a `Disagree`. The number of `Revise` and `Disagree` responses thus include the responses that happened after `View Example`. As shown in Table 5, despite the system's high level of intervention accuracy (recall Table 4), reviewers disagreed more than they agreed with the system's scaffolding feedback, for both diagram and paper reviews. To investigate whether student reviewers were disagreeing with the system for good reasons (e.g., while not perfect, their review was already highly localized), Table 6 reports the percentage of the total number of responses (revisions plus disagreements) that were disagreements, with respect to different bins of true localization ratios. Pearson correlation tests between the percentage of `Disagree` responses (scaled to [0,1]) and the true localization ratio show no significant correlations (p-value's of 0.38 and 0.5 for diagram and paper review data, respectively). Student disagreement thus does not seem to be related to how well the original review had localized comments.

[6] Our data shows that first interventions account for 65% and 84% of total diagram and paper review interventions, respectively, and that reviewers were more reluctant to edit their comments in resubmissions. Based on these findings, the current version of SWoRD has been revised to intervene only once.

Table 7. Comment change patterns by intervention scopes

Change pattern	Scope=In		Scope=Out		Scope=No	
	Number of comments of diagram reviews					
NOT Localized → Localized	26	30.2%	**7**	**87.5%**	3	12.5%
Localized → Localized	26	30.2%	1	12.5%	**16**	**66.7%**
NOT Localized → NOT Localized	**33**	**38.4%**	0	0%	5	20.8%
Localized → NOT Localized	1	1.2%	0	0%	0	0%
	Number of comments of paper reviews					
NOT Localized → Localized	8	20%	**2**	**50%**	5	9.1%
Localized → Localized	13	32.5%	1	25%	**29**	**52.7%**
NOT Localized → NOT Localized	**19**	**47.5%**	1	25%	20	36.4%
Localized → NOT Localized	0	0%	0	0%	1	1.8%

Review Revision. Next we evaluate the effectiveness of the system scaffolding intervention by looking at the human-coded localization annotations for edited comments of different types, where the types are defined in terms of the prior system scaffolding interventions that a reviewer received. A reviewing session starts when the reviewer creates/opens a review and ends when the reviewer submits the review by either passing the localization threshold or disagreeing with the system (by clicking on the rightmost button in Fig. 2). We define three intervention scopes with respect to reviewer edits during a reviewing session:

- `Scope=In`: the reviewer received a system intervention in the current reviewing session.
- `Scope=Out`: the reviewer did not receive a system intervention when submitting a review for the current diagram/paper, but encountered a system intervention for a prior review of that type.
- `Scope=No`: the reviewer of a diagram/paper never received a system intervention for either the current or prior reviews of a diagram/paper.

For each intervention scope, we collect all comments that were edited in the revision and compare each comment's true localization label to the true label of its previous version. Table 7 reports the number of comment pairs according to the four possible ways in which a comment could be changed after editing, with respect to localization. The pattern of most interest is `NOT Localized` → `Localized`, as this was the type of successful edit that the scaffolding intervention was designed to promote. At the other extreme, the least desirable pattern is `Localized` → `NOT Localized`, as this type of comment editing decreased feedback quality with respect to localization.

First, consider the first rows for both the diagram and paper reviews in Table 7, which correspond to the most desirable edit pattern. Comparing columns shows that the percentages of `NOT Localized` → `Localized` in `Scope=In` and `Scope=Out` are larger than that of `Scope=No`, for both diagram and paper re-

views. Moreover, in `Scope=Out` this pattern contributes the largest portion of edits in both diagram and paper review revisions. Such evidence indirectly suggests that the system scaffolding intervention does help reviewers to localize their previously unlocalized comments, and the impact of the intervention still remains in later reviewing sessions after the scaffolding is removed.

The second pattern in the table, `Localized` → `Localized`, has the largest percentage in `Scope=No`. We hypothesize that reviewers who were never scaffolded might be revising their reviews for some reason other than feedback localization which they already did well. However, this pattern also contributes the second largest percentages for the other two scopes. Perhaps reviewers might also be attempting to add more localization signals than that were used in their original comments. In future work we plan to revisit our localization coding (which currently has a binary rather than ordinal value) to determine whether reviewer editing adds further localization, or addresses a different issue.

Our third observation is that for `Scope=In`, the pattern `NOT Localized` → `NOT Localized` accounts for the largest number of edit results in both diagram and paper reviews. This suggests that there is still room for improvement in our scaffolding of review localization. That is, even when reviewers attempted to respond to the system intervention by revising their comments and asking the system to evaluate them again, students still had difficulty in making the comments localized. Potential reasons might be that our current scaffolding messages could be made clearer, or that for some review dimensions giving localized comments is difficult. Investigating these issues will be part of our future work.

Finally, the least desirable pattern of `Localized` → `NOT Localized` occurred only twice in all of the edits. We investigated these instances and found that students apparently deleted their comments by mistake. The rareness of this pattern suggests that our highlighting of localized comments in green helped student reviewers not to remove localization from their localized comments.

7 Conclusions and Future Work

In this paper, we first integrated two review localization models for diagram and paper reviews in a web-based peer review system, then implemented a scaffolding intervention to improve the quality of peer reviews that lacked localization. Furthermore, we deployed the system in a university classroom and evaluated the system in terms of the prediction performance of the two localization models (in a cross-domain fashion), the system scaffolding intervention triggered by these models, and the effect of scaffolding on reviewer revision behavior, using data from the class. Our comment-level results showed that both localization models outperformed majority class baselines, with absolute performance levels approaching prior laboratory results [11,7]. Our review submission-level results demonstrated that the two localization models could also accurately trigger system interventions, yielding only one wrong intervention for a diagram review.

Analyzing reviewer responses to the system intervention, we found that for reviewers who revised their reviews after the system scaffolding intervention,

the number of comments with localization increased after editing. Moreover, the scaffolding intervention appeared to improve localization even in later, non-scaffolded review sessions. However, the results also demonstrated that our current approach could be further improved, as there were both a large number of unsuccessful attempts to localize comments, and a large number of disagreements with the system's suggestion to increase localization.

For future work, we plan to improve our interface to better help students localize their review comments. In addition to using color to distinguish localized and non-localized comments, we plan to highlight the localized text spans in already localized comments (e.g. "Study 17" in the left of Fig. 1). We also plan to do further annotation to examine not only whether, but how strongly, a comment is localized. Finally, we plan to ask reviewers why they are disagreeing with the system, as our initial analyses did not show any relationship with localization.

Acknowledgments. This research is supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120370 to the University of Pittsburgh. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute or the U.S. Department of Education. Our work is also supported by NFS 1122504. We are grateful to our colleagues for sharing the data. We thank M. Lipschultz, K. Ashley, C. Schunn and other members of the ArgumentPeer and ITSPOKE groups as well as the anonymous reviewers for their valuable feedback.

References

1. Gielen, S., Peeters, E., Dochy, F., Onghena, P., Struyven, K.: Improving the effectiveness of peer feedback for learning. Learning and Instruction 20(4), 304–315 (2010)
2. Cho, K.: Machine Classification of Peer Comments in Physics. In: Proceedings of 1st international conference on Educational Data Mining (EDM), pp. 192–196 (2008)
3. Cho, K., Schunn, C.D.: Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. Computers & Education 48(3), 409–426 (2007)
4. Kumar, A.N.: Error-Flagging support for testing and its effect on adaptation. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 359–368. Springer, Heidelberg (2010)
5. Lippman, J., Elfenbein, M., Diabes, M., Luchau, C., Lynch, C., Ashley, K.D., Schunn, C.D.: To Revise or Not To Revise: What Influences Undergrad Authors to Implement Peer Critiques of Their Argument Diagrams? In: International Society for the Psychology of Science and Technology 2012 Conference, Poster (2012)
6. Nelson, M.M., Schunn, C.D.: The nature of feedback: How different types of peer feedback affect writing performance. Instructional Science 37(4), 375–401 (2009)
7. Nguyen, H.V., Litman, D.J.: Identifying Localization in Peer Reviews of Argument Diagrams. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS, vol. 7926, pp. 91–100. Springer, Heidelberg (2013)
8. Ramachandran, L., Gehringer, E.F.: Automated assessment of review quality using latent semantic analysis. In: 11th IEEE International Conference on Advanced Learning Technologies (ICALT), pp. 136–138 (2011)

9. Razzaq, L., Heffernan, N.T.: Hints: is it better to give or wait to be asked? In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 349–358. Springer, Heidelberg (2010)
10. Scheuer, O., Loll, F., Pinkwart, N., McLaren, B.M.: Computer-supported argumentation: A review of the state of the art. International Journal of Computer-Supported Collaborative Learning 5(1), 43–102 (2010)
11. Xiong, W., Litman, D.: Identifying problem localization in peer-review feedback. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part II. LNCS, vol. 6095, pp. 429–431. Springer, Heidelberg (2010)
12. Xiong, W., Litman, D.: Automatically Predicting Peer-Review Helpfulness. In: Proceedings of 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT), pp. 502–507 (2011)

Comprehension SEEDING: Comprehension through *S*elf *E*xplanation, *E*nhanced Discussion, and *IN*quiry *G*eneration

Frank Paiva[1], James Glenn[1], Karen Mazidi[1], Robert Talbot[2], Ruth Wylie[3], Michelene T.H. Chi[3], Erik Dutilly[4], Brandon Helding[5], Mingyu Lin[1], Susan Trickett[5], and Rodney D. Nielsen[1]

[1] University of North Texas
{frankpaiva,jamesglenn2,karenmazidi,mingyulin}@my.unt.edu,
rodney.nielsen@unt.edu
[2] University of Colorado Denver
robert.talbot@ucdenver.edu
[3] Arizona State University
{ruth.wylie,michelene.chi}@asu.edu
[4] University of Colorado Boulder
erik.dutilly@colorado.edu
[5] Boulder Language Technologies
b.a.helding@gmail.com, sbtrickett@bltek.com

Abstract. In this paper we introduce the Comprehension SEEDING system and describe the system components designed to enhance classroom discussion by providing real-time formative feedback to teachers. Using SEEDING, teachers ask free-response questions. As students are constructing their responses using digital devices, SEEDING allows teachers to assess a student's understanding. Once SEEDING collects student responses, the system automatically groups them based on semantic similarity. Teachers can use this information to address student misconceptions and engage the classroom from a more informed perspective. This paper describes the SEEDING system and how it can be used to aid teachers and improve classroom discussion.

1 Introduction

Teachers ask students questions in the classroom both to assess their understanding and also to facilitate learning. Students learn as a result of engaging with the material and participating in shared discourse (Larson, 2000). Although this can potentially be a reasonable way to generate classroom discussion, effective classroom engagement is difficult to achieve this way because teachers can only involve one student at a time. This may cause other students to become disengaged from the discussion. To address this problem, classroom response technologies such as clickers, have been shown to improve student learning and engagement by allowing all students to answer, while providing the teacher with real-time formative feedback.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 283–293, 2014.

Clickers are a classroom response system in which each student has a handheld remote control by which they respond to questions that are projected onto a screen in the classroom. Previous work on clickers has shown that they can be beneficial for enhancing student learning and engagement (Duncan, 2006; Fies & Marshall, 2006; Herreid, 2006; Keller et al., 2007; Penuel, Boscardin, Masyn, & Crawford, 2006; Siau, Nah, Siau, Sheng, & Nah, 2006). However, there are limitations that could explain why small-scale efficacy tests for the use of the technology have seen mixed results (Bunce et al., 2006; Carnaghan & Webb, 2007; Duggan et al., 2007). In order for teachers to take advantage of clickers and any automated response tallying, teachers are limited to asking multiple choice questions. Although multiple choice questions are helpful when assessing basic factual knowledge, it can be difficult to assess deep knowledge in a closed-response question format (Campbell, 1999; McNeill et al. 2009). The effectiveness of clickers is limited to the quality of the multiple-choice questions that teachers pose, and it is difficult and time consuming to construct questions with good distractors. Even with meaningful distractors, multiple-choice questions only require students to *recognize*, rather than *generate* the correct response. According to the Interactive, Constructive, Active, Passive (ICAP) framework (Chi, 2009), constructing answers to free-response questions is a more cognitively engaging task than simply selecting answers to multiple-choice questions and should result in deeper learning.

One of SEEDING's goals is to improve on the engagement advantages afforded by clickers, while largely eliminating their weaknesses. Specifically, SEEDING is a new classroom learning technology that: allows teachers to pose free-response questions, results in all students constructing responses, provides teachers real-time formative feedback, and aims to encourage deeper questions in the classroom.

2 Comprehension SEEDING

SEEDING is grounded in results from three key areas of cognitive and learning sciences research, 1) student self explanation, 2) formative assessment with classroom engagement and discourse, and 3) educational question asking practices. The Comprehension SEEDING system is divided into three analogous distinct but related components that work together to create an enhanced learning environment for both teachers and students. These three components, self-explanation (SE), enhanced discussion (ED), and inquiry generation (ING), are summarized in this section and detailed in the sections that follow, while highlighting their theoretical advantages.

The Comprehension SEEDING system allows teachers to pose free-response questions. Students answer these questions via digital devices (each of the students in our current study, approximately 1250 in total, is using a Google Nexus 7, but classrooms outside the study have used laptops, netbooks, various tablets, android phones, iPhones, and other digital devices). While students compose their responses, the system provides a real-time analysis of the student responses.

Once SEEDING receives most of the student responses, it automatically groups them in up to four clusters. Teachers have the option to view and share each student response with the class. However, showing individual responses can be time consuming and may address misconceptions only held by a few students. Using the clusters, teachers can quickly determine the current overall status of the classroom's understanding of the question posed.

SEEDING allows each student and teacher to interact with the current presented material. To achieve this level of individual interaction, the system needs to address the different requirements of the teacher versus the students and allow each student to use the system simultaneously in the classroom. Our approach consists of a web-based solution that in the present study, runs on Nexus 7 tablets for the students and typically runs on a desktop or laptop for the teacher.

SEEDING operates differently based on the user's role (e.g, student, teacher). Teachers using the system use their classroom computer which connects to a projector. This provides the teacher with two windows, a control dashboard and a classroom display. The first control window, gives teachers the ability to control, manage, view, assess, and teach the classroom. The second window allows the teacher to share student responses, vocabulary words, and images with the classroom. Unlike the teacher windows, we have provided a minimal interface for the students – they can log in, receive questions & vocabulary words, construct their responses, and logout. As an alternative to using the on-screen keyboard, students' tablets are complimented with a physical keyboard to reduce student response time during classroom sessions

3 Self Explanation

Self Explanation. Given a question, Comprehension SEEDING allows students to reflect on their knowledge of the concepts involved and construct a free-response answer, shown in Figure 1. It is important to note that this approach is not focused on solely getting individual responses nor is it focused on incorporating more technology into the classroom. This approach engages students in a complex cognitive task that causes the student to self-reflect as they compose their response.

These cognitive tasks can be thought of as a form of self-explanation, which has been shown in numerous studies to increase student learning gains (Chi, 2009). Importantly, SEEDING enables all students in the class to engage in this cognitive task, rather than just one student at a time, as is typically the case when a teacher asks a question in the classroom. We hypothesize that students using SEEDING to self-explain or articulate their beliefs about a subject will achieve learning gains similar to those seen in typical self-explanation scenarios.

Vocabulary List. Second language (SL) learners and students with low prior domain knowledge often struggle to articulate their explanations because they can't recall the right words. To aid these students in their self-explanation, SEEDING generates a vocabulary list. This list includes key content words extracted from the question's reference answer as well as various foils to mitigate

the possibility of providing too strong of cues to the answer. Key content words are determined by their mutual information with the other questions and reference answers that the teacher saved in the same folder as the question being asked. The distractor words include key content words from those same related questions and their reference answers, WordNet's (e.g., WordNet is a freely available, machine-readable, lexical database for English available at: http://http://wordnet.princeton.edu) antonyms of the other words in the vocabulary list, and WordNet coordinate terms.

All the words in the vocabulary list are lemmatized, to extract the root. Repeated lemmas and words in the question, which the student can already see, are removed from the list. Only the most relevant distractors, those whose mutual information with the reference answer was the highest, are kept. Through teacher use, we empirically determined that ten words was the best number to keep. Finally, SEEDING presents the alphabetized list to the teacher, who is free to add or remove words from the vocabulary box and to send the list to any individual or to all logged in students. Ultimately, SEEDING aims to cognitively engage all students in self-explanation as they are constructing their responses and the vocabulary list can help by keeping SL learners and students with low prior knowledge engaged in the self-explanation process.

4 Enhanced Discussion

As students respond to a question, SEEDING performs analysis and provides teachers real-time feedback on the students' understanding. This is accomplished with system components such as a word cloud, clustering, and immediate presentation of individual student responses. The word cloud is updated in real-time to reveal the concepts students are focusing on in their responses. Clustering provides the teacher with representative responses from up to four primary groups of similar student responses, The presentation of individual student responses allows the teacher to check in on struggling students. Teachers can utilize all of this real-time feedback to evaluate whether or not the classroom understanding is headed in the direction they intend and decide what course corrections are necessary to clear up any issues or misconceptions.

Word Cloud. As students are constructing their free-response answers, SEEDING presents the teacher with a word cloud. A word cloud is a presentation of words that populates itself with frequently used content words. In this case, the word cloud is populated with words extracted from all of the student responses. A word is only presented to the teacher if it is used by more than one student. The more students that use a content word, the larger it will appear in the word cloud. The word cloud allows teachers to begin to assess the class' understanding before students submit their final responses.

Clustering. After students have submitted their responses to the teacher, SEEDING automatically clusters the responses in up to four groups based on

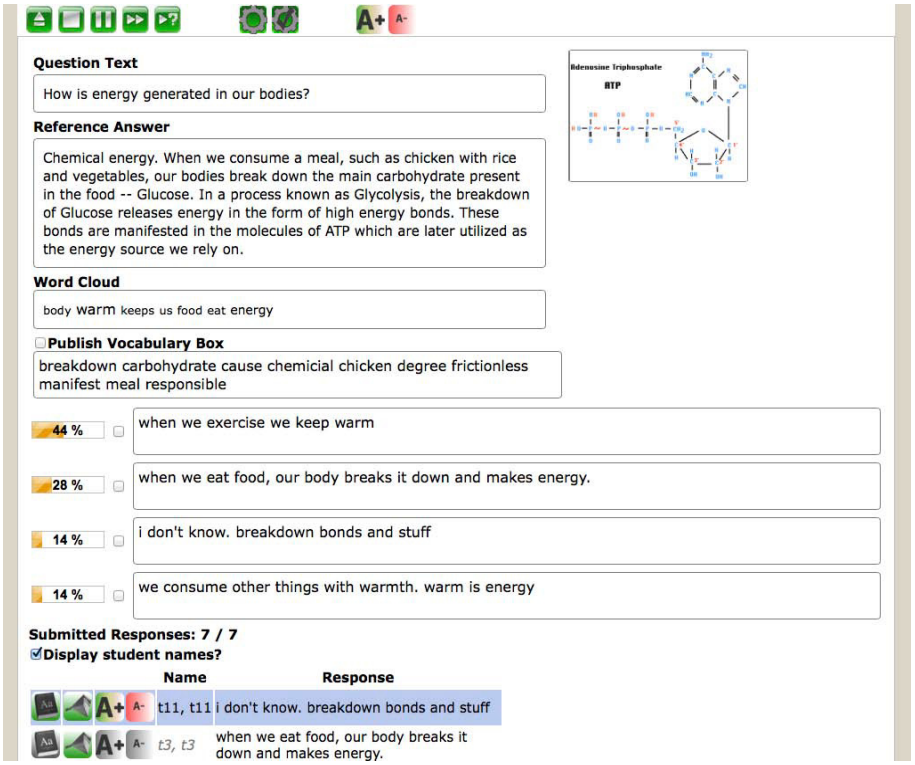


Fig. 1. Teacher control dashboard. Teachers view the word cloud, cluster representatives, and student responses.

semantic similarity. SEEDING will then present the teacher with a representative response for each cluster along with the percentage of student responses belonging to that particular cluster as shown below in figure 1. A cluster's representative is the student response that is the most representative of all of the responses in that cluster. The teacher has the option to share any or all of the cluster representatives with the class. Clustering and representative processing is hypothesized to facilitate meaningful classroom discussion because the teacher is presented with a sample of responses that represents the diverse views of the classroom. In addition, the teacher could address misconceptions in cluster representatives, ask the students to edit and resubmit their responses, and re-cluster the student responses.

To cluster student responses, we need an understanding of each student's response and its entailment relationship to the question's reference answer. We do not simply want to label responses as correct or incorrect. Instead if a response is not correct, we want to identify where the student's response is different from the reference answer and in what way it is different. To achieve this level of semantic analysis, SEEDING decomposes the question, its reference answer, and all the responses into their fine-grained semantic facets following (Nielsen et. al,

2009). An analysis of all of these semantic facets is used to generate the feature vectors used by the clustering algorithm, as discussed below.

Feature vectors are comprised of four sets of features, each of which is assigned a total weighting or importance. The sum of the weights over the four sets of features is 1.0. The first set of features is based on the subset of semantic facets found in the reference answer that are not also found in the question. These features were given a weight of 0.45. The second set of features, which has a weight of 0.225, is based on the remaining facets found in the reference answer (i.e., those facets that also existed in the question). The third set of features, with a weight of 0.1, is based on the facets found only in the question. The final set of features, comprising the remaining weight of 0.225, is based on any additional facets that occur in multiple student responses. In future work, the weights of each set of features will be learned based on training data. In the present work, facets from the reference answer were given most of the weight (just over 2/3 of the total weight), since those are the primary semantics of interest. Since it is easy for a student to just repeat words from the question, related facets were given less weight. Student responses are converted into feature vectors according to which facets in these four groups is entail by the response. These vectors are then used in the clustering process.

SEEDING automatically initiates the clustering when the percentage of students that have responded surpasses a threshold.[1] However, teachers have the option to cluster the responses much earlier, if desired, and are free to re-cluster the responses at any time, if they want to account for more complete information. Each time the teacher clusters responses, the system recomputes the feature vectors for any student response that has changed.

At the core of SEEDING's clustering is the k-means algorithm, shown in the equation below. Given a set of student responses, the goal is to find the assignment of responses, x_j, to k clusters, $\mathrm{S} = \{S_1, S_2, ...S_k\}$, that minimizes the sum of the squared distances between the response vectors, x_j, and their associated (nearest) cluster centroid, μ_i.

Once all student responses have been converted into feature vectors. Four randomly selected student response vectors are assigned as the initial cluster centroids. We iterate over each student response vector, calculate its distance from each cluster centroid, and assign the response to the cluster whose centroid is closest. After each iteration, the cluster's centroid is recalculated by averaging the response vectors assigned to it. These two steps, assigning responses to the closest cluster and recomputing the cluster centroids, are repeated for 10 iterations or until convergence, when the clusters stop changing.

Following the clustering, representative responses are selected for each cluster. These representatives are presented to the teacher, who can use them to lead a classroom discussion focused on the main beliefs expressed by students. For each cluster, the response whose vector is determined to be closest to the cluster's centroid is selected as the cluster representative.

[1] In the present work, teacher feedback indicated that 50% was a reasonable threshold to present the teacher with cluster representatives.

These cluster representatives provide the teacher with a good sense of the student conceptions in the classroom. The teacher projects the representative responses onto the classroom display and engages the students in a discussion based on the various beliefs exemplified. Unlike clickers, which only allow teachers to guess a priori when writing the distractors what the misconceptions might be, SEEDING's Enhanced Discussion can directly target the beliefs held by the teacher's students. Unlike typical classroom discussions, which engage and address the perspective of only a single student at a time, SEEDING's dialogue is grounded by the diverse beliefs held in the teacher's classroom.

5 INquiry Generation

The question generation component of the SEEDING project is designed to expand the classroom discussion to a view of the topic as explored in the wider world, and to inspire teachers to think of science as a verb, not a noun. That is, science is not a static body of factual knowledge but a process of exploration, discovery, and peer review. The question generation component itself is being introduced in phases which represent different approaches to question generation. Phase I involves questions from the QtA Questioning the Author (Beck, 2001) framework, which has also been included in teacher training. Phase II utilizes questions extracted from the web. Phase III requires the development of a knowledge base, from which conceptual questions can be generated.

The Phase I QtA component takes all student responses as input, as well as the teacher question and reference answer. Common ideas are identified in the student responses by means of word frequency counts. Meanwhile, the teacher question is analyzed to see if a concept can be extracted. For each noun in the teacher question, mutual information is calculated between these nouns and the question category extracted from within the SEEDING system. The highest scoring noun is selected as the concept, with preceding nouns and prepending adjectives, as in *kinetic energy*. There are over 100 QtA question stems which are divided into subsets for random selection based on whether the teacher question referenced a lab or experiment, whether a teacher question concept or student common idea was identified, or one of the remaining question stems. Sample stems include:

- Can you think of another experiment we could do which would teach us more about *concept*? If you were explaining *concept* to a younger person, what other knowledge would they need to understand your explanation?
- Many of you mentioned *common idea*. Does anyone disagree?
- After reading the responses on the screen, what would you change about your response, and why? If you would not change your response, why is yours better?

The questions extracted from the web in Phase II utilize the teacher question and reference answer in the web search. These texts are tokenized and tagged by the Stanford taggers, and stop words are removed. Words from this group with

the desired parts of speech (nouns, verbs, adjectives) are extracted as keywords. These keywords are sent to a Google custom search engine to retrieve relevant urls. A web crawler then traverses these urls, and the links from those pages, to extract all questions from the pages it crawls. Questions are rated according to the frequency of the keywords, and the top ranking questions are sent to be displayed. For example, the teacher question How is work turned into mechanical energy? results in the keywords: work, turned, mechanical, and energy. The top retrieved questions are:

- What devices convert mechanical energy to heat energy?
- How can mechanical energy be converted to heat energy?

Note that these questions extend the discussion beyond the original teacher question to more application and conceptual questions. The urls from which the questions were retrieved are also provided to the teacher.

6 Discussion

As of the spring 2014 phase, over 1200 students are using SEEDING in their classrooms. We collect feedback from the teachers and make changes to the system accordingly. As a result, new ways to enhance the classroom learning environment are still being developed.

Evaluation in Progress. To evaluate the effectiveness of the Comprehension Seeding system compared to traditional and clicker classrooms, we are conducting a yearlong pilot study within sixth grade science classrooms. We are analyzing the effect of the SEEDING system use on student learning, in addition to learning more about SEEDING adoption, use and integration into teacher practices. With respect to teacher adoption and use, we have collected a substantial amount of data from the teachers starting with the participatory design process and following all the way through system deployment and use. This data consists of informal interviews with teachers, short surveys, frequent email follow-ups, and discussions during researcher and support team visits. To date, the teachers have been very forthcoming with their system design needs, desires, issues, and potential barriers to use. This information has contributed greatly to our ability to make the system and interface "teacher friendly." We also collected a very substantial amount of observation and system log data related to teachers' use of the system in practice. This data helps us to make sense of how the teachers are integrating the system into their practice. As a specific example, we would hope that the teachers use the system to gather class-level formative feedback that will help them lead a rich follow-up discussion. Observation and logs can tell us if teachers are asking follow-up questions to the initiating questions, how long those questions are open for student responses, and whether or not the teacher pauses the question during student response (perhaps to discuss or clarify). In this way we are able to identify any specific pedagogical needs that

the teacher may have in order to fully integrate the system into their classroom practice.

Teachers' (and students') feedback on the system has been overwhelmingly positive. The teachers' especially appreciate the fact that all students can individually respond to a question, and that student responses can be displayed for class discussion. Students enjoy expressing their own thoughts, and become very excited when their responses are displayed as one of the cluster representatives.

We are in the process of collecting student assessment data to investigate the effect of the system on student learning. We have structured a within-teacher research design in order to control for teacher effects. Any given teacher in our research is teaching one or more class sections using the system, and other sections using clickers (multiple choice only) or no technology support. We have designed our own assessments of students' deep learning in four science units: Atoms & Elements, Particulate Model of Matter, Force & Motion, and Energy. These assessments consist of both open ended and multiple choice items that span a range of cognitive depth. Each class section (SEEDING, clicker, or no technology support) responds to each unit test pre and post instruction for that unit. The students also respond to a year long pre and post test which encompass all of these topics. This data collection and the scoring of the student responses is ongoing.

Rather than collecting this assessment data with paper tests, we added a component to the SEEDING system specifically for this purpose. Using SEEDING, teachers specify what class and exactly how long an assessment should be. Once a teacher begins an assessment, students are redirected from the traditional interface and taken to an assessment page. This page allows students to submit answers through free-response, multiple choice, and canvas, where using a stylus, students can draw their responses to a question. While students are in assessments, they are free to navigate through all the questions in the assessment, edit their responses, or erase their drawings. Once the time for an assessment ends or the teacher decides to terminate the assessment, the students exit the assessment.

Vocabulary List. We plan to do future research that will lead to populating the vocabulary box with words more meaningful to SL learners. We are exploring using a large corpus as a filter to non-science related words. We do this by calculating co-occurrence relationships between science words. In addition, we are exploring extracting hypernyms from content words to provide a broader perspective of the given word.

Facet Cloud. To provide teachers with even more real-time information about student understanding as they construct their responses, we will explore a facet cloud. Similar to the word cloud, the facet-cloud will give teachers an indication of how many students expressed each semantic facet. This will allow teachers to see the semantic relationships students make as they type out their responses. For example, if a teacher asks *Is a proton positive or negatively charged?* as students are responding, the facet-cloud could present facets such as: (proton, neutral), (atom, positive), etc. Teachers can use this feedback to guide the classroom discussion accordingly.

7 Conclusion

It is expected that combining the scientifically-grounded educational support technology and methods in Comprehension Seeding will result in learning gains that could exceed the one sigma gain found in the best current tutoring systems as well as the more modest gains associated with effective implementation of clicker systems. From a cost-benefit perspective, Comprehension SEEDING has the potential to inexpensively provide a practical, focused, nearly individualized, adaptive, scientifically based solution. Furthermore, this solution is not tied to one specific inquiry-based pedagogy or to science education, but rather has the potential for significant positive impact across many areas in education. We are currently conducting a study involving approximately 1250 students to assess the impact of Comprehension SEEDING in the classroom.

Acknowledgements. The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120808 to UNT. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

References

1. Beck, I., Margaret, M.: Inviting students into the pursuit of meaning. Educational Psychology Review 13(3), 225–241 (2001)
2. Bunce, D.M., VandenPlas, J.R., Havanki, K.: Comparing the effectiveness of student achievement of a student response system versus online WebCT quizzes. Journal of Chemisty Education 83(3), 488–493 (2006)
3. Carnaghan, C., Webb, A.: Investigating the Effects of Group Response Systems on Student Satisfaction, Learning, and Engagement in Accounting Education. Issues in Accounting Education 22(3), 391–409 (2007)
4. Chi, M.T.H.: Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities. Topics in Cognitive Science 1(1), 73–105 (2009)
5. Duggan, P.M., Palmer, E., Devitt, P.: Electronic voting to encourage interactive lectures: A randomised trial. BMC Medical Education 7(25) (2007)
6. Duncan, D.: Clickers: A New Teaching Aid with Exceptional Promise. Astronomy Education Review 5(1), 70 (2006)
7. Fies, C., Marshall, J.: Classroom Response Systems: A Review of the Literature. Journal of Science Education and Technology 15(1), 101–109 (2006)
8. Herreid, C.F.: Clicker Cases: Introducing Case Study Teaching Into Large Classrooms (2006)
9. Keller, C., Finkelstein, N., Perkins, K., Pollock, S., Turpen, C., Dubson, M., Hsu, L., et al.: Research-based Practices For Effective Clicker Use. In: AIP Conference Proceedings, pp. 128–131 (2007)
10. Larson, B.: Classroom discussion: A method of instruction and a curriculum outcome. Teaching and Teacher Education 16(5-6), 661–677 (2000)

11. Nielsen, R.D., Ward, W., Martin, J.H.: Automatic Generation of Fine-Grained Representations of Learner Response Semantics. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 173–183. Springer, Heidelberg (2008)
12. Penuel, W.R., Boscardin, C.K., Masyn, K., Crawford, V.M.: Teaching with student response systems in elementary and secondary education settings: A survey study. Educational Technology Research and Development 55(4), 315–346 (2006)
13. Siau, K., Nah, F.F., Siau, K., Sheng, H., Nah, F.F.: Use of a Classroom Response System to Enhance Classroom Interactivity (2006)

Pedagogical Evaluation of Automatically Generated Questions

Karen Mazidi and Rodney D. Nielsen

HiLT Lab., University of North Texas, Denton TX
KarenMazidi@my.unt.edu, Rodney.Nielsen@unt.edu

Abstract. Automatic Question Generation from text is a critical component of educational technology applications such as Intelligent Tutoring Systems. We describe an automatic question generator that uses semantic-based templates. We evaluate the system along with two comparable systems for both linguistic quality and pedagogical value of generated questions and find that our system outperforms prior work.

Keywords: question generation, syntactic, semantic, pedagogy.

1 Introduction

This work evaluates three automatic question generation systems which have a common aim: to assist students in remembering and understanding what they have read. Roediger and Pyc [12] describe studies which show that students who are more frequently asked questions retain significantly more than those who are not. Beck et al. [3] demonstrate that reading comprehension can be boosted with questions that are generated automatically. In creating question generation systems for educational technology applications, a crucial design consideration concerns what kinds of questions should be generated. Graesser, Rus and Cai [7] explore many facets of this consideration, including question taxonomies, purpose of questions, and assumptions behind questions. Another consideration is whether the questions should be answerable from the text. This consideration is addressed by Graesser et al. [6] in the context of information sources: whether the answer comes from the text, student knowledge, or other sources.

Question generation approaches are often classified on a syntactic-semantic continuum. In a syntactic approach, the sentence structure is rearranged and altered to turn declarative sentences into questions. Syntactic examples include early work from Wolfe [13] through recent work from Heilman and Smith [8]. Another syntactic approach, the Ceist system [14], manipulates syntax trees, but the rules are stored externally in templates. Syntactic approaches tend to outnumber semantic approaches as seen in the Question Generation Shared Task and Evaluation Challenge 2010 [4] which received only one paragraph-level, semantic entry[11]. Argawal, Shah and Mannem [1] continue the paragraph-level approach using discourse cues to generate questions of types: why, when, give an example, and yes/no. Another recent semantic approach is Lindberg et al. [10]

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 294–299, 2014.

Table 1. Classification of question generation approaches

	Internal Rules	External Rules
Syntactic Constituents	Heilman and Smith	Ceist
Semantic Constituents	Argawal, Shah, Mannem	Lindberg et al.

which used semantic role labeling combined with templates. This latter approach most closely parallels our own; however, our approach is domain-independent, and our system generates answers as well as questions.

Table 1 is provided as an assist in classifying these various approaches. On one axis, approaches are classified according to whether they are manipulating syntactic or semantic constituents of a sentence. On the other axis, they are classified according to whether the rules for this manipulation are internal to the program or kept externally, as in the form of templates. The examples shown are to provide a general frame of reference, not to imply that any one system entirely fits into one category. Most systems cross the boundary lines of Table 1.

2 Approach

The question generation system presented here utilizes semantic role labels and templates. Sentences are processed by SENNA [5], which provides the tokenizing, pos tagging, syntactic constituency parsing and semantic role labeling, using the 2005 Propbank coding scheme [2]. SENNA produces separate semantic arguments for each predicate in the sentence which are matched with appropriate templates. Question generation patterns use the more common semantic roles A0 (proto-agent), A1 (proto-patient), and A2 - A4 (meaning varies by predicate), as well as the ArgM modifiers: directionals, locatives, manner, purpose, cause, discourse, adverbials, and temporal. Templates contain five fields: (1) the question type identifier, (2) required fields, (3) question frame, (4) answer, and (5) filter fields. Generated questions are stored by the question type identifier for later retrieval by question type/depth. The system at the time of this evaluation had 42 question types. Required fields specify what semantic argument should be present, or absent, and any required verb forms. The answer field specifies which semantic argument is the answer to the question. Filter fields will cause a question to not be generated for conditions such as arguments that do not contain nouns. Filters help prevent generating vague or confusing questions.

Table 3 provides examples of questions and the patterns from which they were generated. The question generated in Example 1 uses the form of the verb found in the source sentence. This template requires arguments A0, A1 and ArgM-locative; fields A1 and ArgM-locative are placed in the question, argument A0 is the answer. The full template also had a required field indicating that the verb must be a form of *be* and a filter that excluded predicate-argument sets that included an A2 argument.

Table 2. Examples of Generated Questions

Example 1. Question Frame: What |verb| |A1| |AM-LOC|? **Source text:** Ice wedging is the main form of mechanical weathering in any climate that regularly cycles above and below the freezing point. **Question:** What is the main form of mechanical weathering in any climate that regularly cycles above and below the freezing point? **Answer:** ice wedging
Example 2. Question Frame: How |do A0| |V| |A1|? **Source text:** By examining the arrangement of these dark absorption lines, astronomers can determine the composition of elements that make up a distant star. **Question:** How do astronomers determine the composition of elements that make up a distant star? **Answer:** by examining the arrangement of these dark absoption lines
Example 3. Question Frame: What happens |if|? **Source text:** If the atoms are pulled apart, potential energy goes up because you are separating particles that attract each other. **Question:** What happens if the atoms are pulled apart? **Answer:** potential energy goes up

In Example 2, there are three fields in the question frame that must be replaced with source sentence text. The |do| field will be replaced by *do*, *did* or *does*, depending on the plurality of the nouns and the tense of the verb. The verb will be in its lexical form. Filters in the full template specify that A0 cannot start with a preposition and A1 cannot start with a personal pronoun. The first filter helps with question naturalness and the latter filter helps avoid vague questions. A required field specifies that the ArgM-manner argument which forms the answer must contain a gerund.

In Example 3, the |if| of the question frame will be replaced with the text from the ArgM-adverbial. The full template specification has a filter which indicates that the ArgM-adverbial must contain nouns. This is another filter for vague questions.

3 Linguistic and Pedagogical Evaluations

For these evaluations, we utilized Amazon's Mechanical Turk service. Previous work by Heilman and Smith [9] demonstrates that satisfactory results can be achieved by submitting work in small batches, and closely monitoring each batch. For these evaluations we set up two separate tasks: a linguistic evaluation and a pedagogical evaluation. For the linguistic evaluation, each worker was asked to read the source sentence and question, then rate the question on a 1 to 3 scale for grammaticality and clarity. For the pedagogical evaluation, workers were asked to consider whether this question would help them remember or understand the meaning of the sentence. For all tasks we requested two workers and submitted the questions in batches of 50 or fewer questions.

For these two evaluation tasks we compiled two corpora representing the domains of social studies and science. The social studies text was taken from SparkNotes *Other Topics*. Five files were randomly chosen representing the following domains: Economics: the money supply, History: American History, Government: Federalism, Philosophy: an overview of John Locke's work, and Civics: the development of the nation-state. These files range in length from 27 to 39 sentences, with an average of 33 sentences. The science text was extracted from middle-school and high-school science textbooks downloaded from ck12.org, a non-profit that creates and freely distributes K-12 STEM material. The files represent the following science domains: Life Science: the body, Chemistry: bonds, Biology: the cell, Physics: matter and energy, and Earth Science: weathering. The science files ranged in length from 53 to 69 sentences, with an average of 60 sentences.

Table 3. Inter-rater agreement for Mechanical Turk workers

	Social Studies		Science	
	Linguistics	Pedagogy	Linguistics	Pedagogy
Mean agreement	0.72	064	0.69	0.62
Pearson's r	0.58	0.46	0.57	0.45

Table 3 shows the inter-rater agreement between two sets of workers over all annotations. Mean agreement is calculated as shown below, where i ranges over the N questions rated by the annotators, $r_{1,i}$ is annotator 1's normalized rating $(rating-1)/2$ for the *ith* question (normalized ratings fall in the range [0,1]). We also provide Pearson's correlation coefficient numbers, which indicate a strong positive relationship[1] and are statistically significant, $p < 0.001$.

$$1 - \frac{1}{N}\sum_{i=1}^{N} |r_{1,i} - r_{2,i}| \tag{1}$$

The evaluations described here compare the questions generated by the system described in this paper (M&N), Heilman and Smith's system (H&S), and the Lindberg et al. system (LPN&W). Heilman and Smith's system is available online[2]; David Lindberg graciously shared his code with us. For the following evaluations, 50 questions were randomly selected from all questions generated by each system for a given input file. Table 4 shows the number of questions remaining after a given evaluation filtered out lower-quality questions. The table shows this data for both the social studies and science corpora. For both the linguistic and pedagogical evaluations, the questions that remained were those that received a 3 from one worker, and at least a 2 from the other.

From Table 4, the linguistics evaluation for both data sets are remarkably similar. The average number of questions that remained after applying the linguistics filter to the social studies data was 28, 30, 37 (H&S, LPN&W, M&N),

[1] http://faculty.quinnipiac.edu/libarts/polsci/statistics.html
[2] http://www.ark.cs.cmu.edu/mheilman/questions/

Table 4. Number of acceptable questions for social studies and science corpora

Social Studies		Linguistic Evaluation			Pedagogical Evaluation		
File	Questions	H&S	LPN&W	M&N	H&S	LPN&W	M&N
money	50	36	42	45	18	22	22
amhist	50	29	37	36	22	15	20
federalism	50	27	28	40	11	5	20
locke	50	19	21	30	6	8	10
state	50	27	21	32	11	10	13
Average	50	27.6	29.8	36.6	13.6	12	17
Percent		55.2	59.6	**73.2**	27.2	24.0	**34.0**
Science		Linguistic Evaluation			Pedagogical Evaluation		
File	Questions	H&S	LPN&W	M&N	H&S	LPN&W	M&N
body	50	33	27	42	23	14	25
bonds	50	30	34	31	16	19	19
cell	50	30	26	37	20	14	25
matter	50	25	32	32	12	17	18
weathering	50	18	31	38	9	21	26
Average	50	27.2	30	36	16	17	22.6
Percent		54.4	60.0	**72.0**	32.0	34.0	**45.2**

and for the science data: 27, 30, 36. This speaks both to the consistency of all 3 systems across domains, and to the validity of using MTurk for this evaluation.

Discussion. The question generation systems described in this work begin with expository text. Our system takes this input directly into SENNA. The Heilman and Smith system performs NLP transformations on the input text in order to simplify complex sentences, which they note is "particularly prone to errors"[8]. Using a semantic role labeler essentially performs this simplification itself since it identifies semantic arguments for each predicate in the sentence even within subordinate clauses. The Lindberg et al. system likewise did not perform sentence simplification because they note that important semantic content can be lost, such as temporal information in prepositional phrases [10].

An additional advantage of semantic role labeling is that it can help identify the most salient aspects of a sentence. From: *As the ball gains height, it regains potential energy because of gravity*, a syntactic approach generates the question: What regains potential energy because of gravity as the ball gains height? In contrast, our approach identifies an ArgM-causation argument and can generate a deeper question: Why does the ball regain potential energy?

Heilman and Smith's system provides the answer as well as the generated question, as does our system. The Lindberg et al. system does not provide answers which frees it to ask questions that may not be directly answerable from a sentence. Whether or not this is desirable may depend upon the application.

4 Conclusion

We have evaluated three question generation systems in terms of both the linguistic quality of the produced questions, as well as their pedagogical utility. These types of question generation systems can be integrated into educational technology applications such as Intelligent Tutoring Systems, in order to ensure that students engage deeply with the material. Our system outperformed prior work in both the linguistic and pedagogical evaluations.

Acknowledgements. This research was supported by the Institute of Education Sciences, U.S. Dept. of Ed., Grant R305A120808 to UNT. The opinions expressed are those of the authors.

References

1. Agarwal, M., Shah, R., Mannem, P.: Automatic question generation using discourse cues. In: Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications, pp. 1–9. Association for Computational Linguistics (2011)
2. Babko-Malaya, O.: Propbank annotation guidelines (2005), `http://www.verbs`
3. Beck, J.E., Mostow, J., Bey, J.: Can automated questions scaffold childrens reading comprehension? In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 478–490. Springer, Heidelberg (2004)
4. Boyer, K., Piwek, P. (eds.): Proc. QG2010: The Third Workshop on Question Generation, Pittsburgh, PA (2010)
5. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Machine Language Research 12, 2493–2537 (2011)
6. Graesser, A.C., Rus, V., Cai, Z., Hu, X.: Question answering and generation. In: McCarthy, P.M., Boonthum, C. (eds.) Applied NLP. IGI Global, Hershey (2011)
7. Graesser, A.C., Rus, V., Cai, Z.: Question classification schemes. In: Proc. WS of the QGSTEC (2008)
8. Heilman, M., Smith, N.A.: Question generation via overgeneration transformations and ranking. No. CMU-LTI-09-013. Carnegie-Mellon University, Pittsburgh (2009)
9. Heilman, M., Smith, N.A.: Rating computer-generated questions with Mechanical Turk. In: Proc. of the NAACL HLT 2010 Workshop CSLDAMT, pp. 35–40. Association for Computational Linguistics (2010)
10. Lindberg, D., Popowich, F., Nesbit, J., Winne, P.: Generating natural language questions to support learning on-line. In: Proc. 14th European Workshop NLG, pp. 105–114. Association for Computational Linguistics (2013)
11. Mannem, P., Prasad, R., Joshi, A.: Question generation from paragraphs at UPenn. In: Proc. QG2010: The Third Workshop on Question Generation, Pittsburgh, PA, pp. 84–91 (2010)
12. Roediger, H.L., Pyc, M.A.: Inexpensive techniques to improve education. J. Applied Research in Memory and Cognition 1(4), 242–248 (2012)
13. Wolfe, J.H.: Automatic question generation from text. ACM SIGCUE Outlook 10(SI), 104–112 (1976)
14. Wyse, B., Piwek, P.: Generating questions from openlearn study units (2009)

Content-Dependent Question Generation for History Learning in Semantic Open Learning Space

Corentin Jouault and Kazuhisa Seta

Graduate School of Science, Osaka Prefecture University, Osaka, Japan
jouault.corentin@gmail.com

Abstract. This research's objective is to support learners in self-directed learning of history in an open learning space. Learners who request help are provided with a list of questions to orient them to new information. All the support is provided only on request and gives them multiple possibilities to give them more freedom in self-directed learning. The originality of our research is that the generated questions are content-dependent. To be able to generate such support, we had to overcome one major problem: the information in the open learning space needs to be understood by our system. The construction of this "semantic open learning space" permits the system to generate questions depending on the studied contents and the learner's concept map.

Keywords: Self-directed Learning, History Learning, Question Generation, Semantic Open Learning Space, Adaptive Learning Support.

1 Introduction

When learning in an open learning space such as the Internet, learners encounter a quantity of information far more superior than in classroom learning. This quantity of information can easily overwhelm a learner [1]. Learners can have difficulties in planning their learning if they do not have the necessary skills.

Our objective is to support learners in self-directed learning of history in an open learning space. Our approach is to provide to learners content-dependent advice depending on their knowledge level. To encourage history thinking, we chose to provide advice in the form of enquiry questions [13]. The advice is provided only on request of the learners to orient without forcing them.

In history learning, an understanding of chronology is necessary [15]. Chronology is defined by Smart [14] as "the sequencing of events/people in relation to other and existing knowledge of other, already known, events/people". Learning history is not only remembering a series of facts, learners need to construct an image of the past in their mind. Learners need, of course, to know the events but they also need to understand their context. We need to support learners in acquiring information about both the chronology and the context.

To achieve our objective, there are two major issues to solve:

- The system needs to be able to understand the learning materials to be able to provide content-dependent advice. In an open learning space, preparing all the documents is impossible, semantic information becomes necessary.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 300–305, 2014.

- The generated questions need to orient the learners to new information that they can use to reinforce their understanding. The questions should encourage the learners to improve their knowledge of the context. Having a good knowledge of the context will reinforce their understanding of the events [8].

In this paper, we will discuss the technical issues to overcome to generate content-dependent questions to support learners in an open learning space.

2 Related Work

To support learners in self-directed learning, previous research already created systems such as the Navigation Planning Assistant [9], which provides an environment used to describe learners' learning plans and state of understanding to prompt their self-regulation in an open learning space. The limitation of this system, however, is that its support is content-independent due to the difficulty of working with natural language information on the Web. Of course, this problem can be overcome by preparing the learning materials in advance. This is the case of the Betty's Brain system [10], which uses concept map in an environment for learning by teaching using concept map, or the Kit Build method [7], which provides a knowledge externalization environment for building a concept map and supporting the learner during the concept map construction. However, for both systems, the preparation requires a considerable amount of time even for constructing a closed learning space. It is not possible to use the same method in an open learning space because there is too much material. In our system, the quality of the semantic information is not as good as manually prepared information but the process can be applied automatically for every concept. Therefore, it can be applied to an open learning space.

To create questions in an open learning space, automatic question generation is necessary. Research from Mostow [11] generates multiple choices questions from a text to diagnose comprehension failure. However, to encourage history thinking, we need to generate opened questions [13]. With a similar objective is research from Heilman [6] which has successfully generated factual questions from complex sentences. However, our process is different because the starting point of the question generation is not natural language but semantic information. We have at our disposition a large quantity of semantic information, thus, we do not need to process the natural language.

3 The Semantic Open Learning Space

When using the system, learners are provided with the same starting point. At first, they are provided with a document introducing the main subject of learning. Our current working example is World War 1. The document will appear in the document window, in Fig. 1(b). All documents are taken from Wikipedia and contain mentions to other concepts. When a learner clicks on a mentioned concept in the document, it is added to the concept map in the concept map window, in Fig. 1(c). The concept map display is designed for history learning: it focuses on the events. The center of the concept map shows the timeline of the events. The others related concepts that form

the context are displayed around the timeline. The learner can also interact with the concept map to add relations between concepts or request more detailed information. If they try to add a wrong relation between two concepts, they are advised to study the related concepts again. Every concept map will be different depending on the learners' knowledge and interests. All the information input to the concept map is controlled by the system and can be analyzed easily by comparing to the semantic information in the system.

Support is only provided on request. Leaners are instructed to use the support function only when they have difficulties in directing their learning. When support is requested, the system will generate a list of questions depending on the learner's concept map; this list is displayed in the question window, in Fig. 1(a). Selecting a question will provide a document containing information that can be used to answer it. The learner can learn from the document and add the concepts that answer the question to the concept map. S/he can also discover new leads to pursue his/her learning.

To understand the learning materials, the system uses three information sources, Wikipedia for natural language information and two for semantic information: DBpedia [2] and Freebase [3]. They are both projects that aim to create a semantic copy of Wikipedia. Both databases provide links to the related Wikipedia document, thus, the system can identify as the same on both databases. The main difference between the two projects is that Freebase's information is provided by humans but DBpedia's information is automatically extracted from Wikipedia.

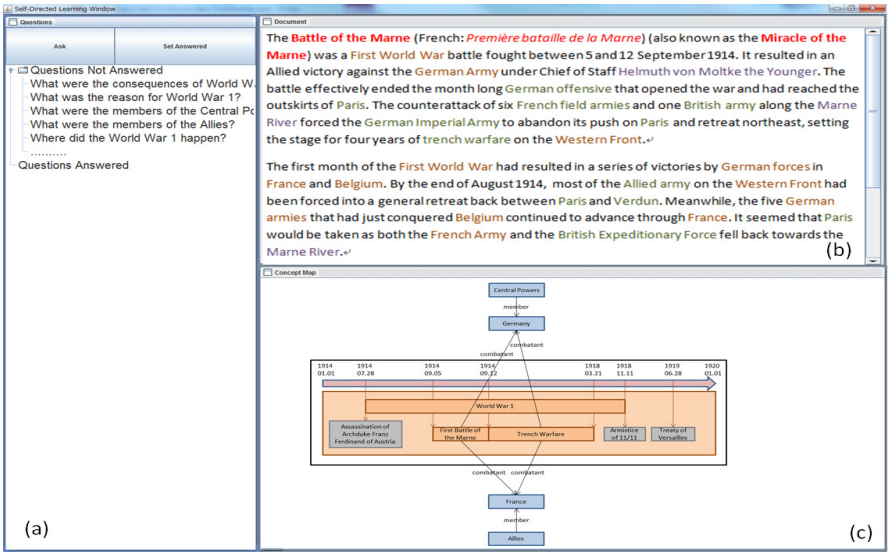


Fig. 1. System Interface

4 Generating the Content-Dependent Questions

The problem is that learners cannot always generate good questions [12]. The quality of the learning depends on the quality of the questions during this process [4].

According to Riley [13], a good enquiry question in history should: “Capture the interest of your pupils, place an aspect of historical thinking at the forefront of the pupils' minds and result in a tangible "outcome activity" through which pupils can genuinely answer the enquiry question”. Questions shouldn’t be descriptive but encourage learners to build their understanding of history like “Did the First Battle of the Marne changed the course of WW1?” When answering a question, learners should first look for the information and then analyze it to build their own interpretation [8].

Fig. 2 shows the question generation process for two concepts, the Military Conflict Race to the Sea and the Country German Empire. All concepts and relations have a type which is associated to a question in the ontology. The system requests the corresponding question to the ontology. The natural language pattern contains a marker giving the position where to insert the concept name. For example, the type Military Conflict is associated to the Question Pattern “How did the X influence the rest of the conflict?” and the marker “X” will be replaced by the concept name “Race to the Sea”.

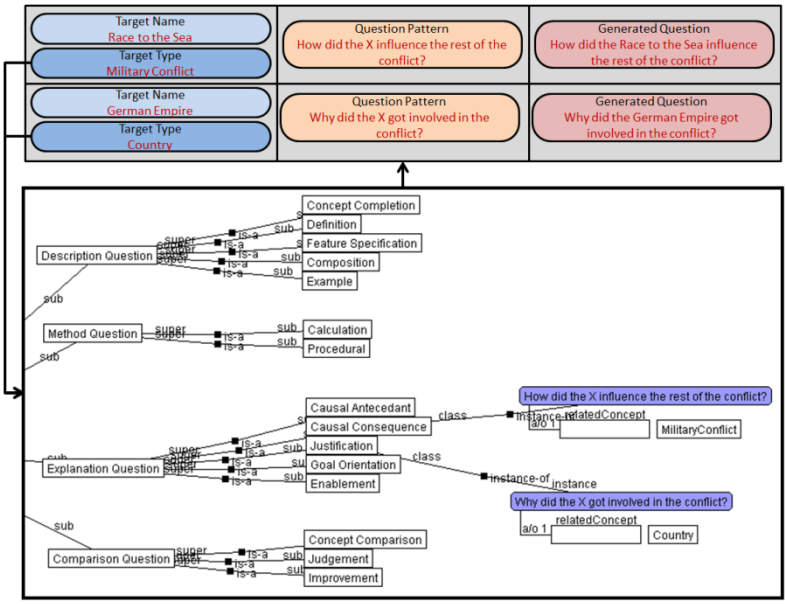


Fig. 2. Natural Language Question Generation

By working on answering such questions, learners will develop their knowledge of the context to reinforce their understanding. With the questions, learners can focus on acquiring knowledge to answer the question. It becomes easier for them to navigate the hyperspace because they have an objective. Since they lack planning skills, this reduces their meta-cognitive charge to help them focus on learning the contents.

The targets of the questions are identified by comparing the concept map built by the learner and the one built by the system using the semantic information. The resulting questions will be generated from the concepts which the learner knows the least also giving priority to the most important and reliable concepts.

Once the targets have been identified, the system needs to generate the natural language questions for the learner. The generated questions use types defined in Graesser's taxonomy [5] and are content-independent. The History Dependent Question Generation Ontology shown at the bottom of the Fig.2 makes the link among the questions types, the concept types and the relation types. This natural language patterns are hand written for every concept type.

5 Concluding Remarks and Evaluation of the Generated Questions

In this paper, we described a way to generate content-dependent questions in an open learning space. We first described the semantic open learning space we created by combining Wikipedia with two semantic information sources: Freebase and DBpedia. Then, we describe our methodology to generate questions to trigger history thinking. The questions are generated from the semantic information using questions pattern and become content-dependent questions.

To evaluate our method, we generated questions for 600 concepts in the WWI category on Wikipedia. For about 50% of the concepts, no semantic information was available. These concepts are very minor, for example, most plane or boat models used during the war have their own page containing close to no information. For the remaining concepts, the questions have been generated and organized by order of importance and reliability as calculated by the system. The following are 5 questions judged important by the system:

- What were the consequences of World War I on Austria-Hungary?
- Did the Siberian Intervention change the course of World War I?
- Would World War I have been different without Ferdinand Foch?
- Did the First Battle of the Marne change the course of World War I?
- How was the German Papiermark used during World War I?

These questions are very relevant to the study of WWI. Using our method, we have very little error of syntax in the generated questions. For about 5% of the generated questions are not relevant to studied domain. However, the use of the Wikipedia categories makes this problem a rare occurrence.

References

1. Biswas, G., Roscoe, R., Jeong, H., Sulcer, B.: Promoting self-regulated learning skills in agent-based learning environments. In: Proceedings of the 17th International Conference on Computers in Education, pp. 67–74 (2009)
2. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia-A crystallization point for the Web of Data. Web Semantics: Science, Services and Agents on the World Wide Web 7(3), 154–165 (2009)
3. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: A collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250 (June 2008)

4. Bransford, J.D., Brown, A., Cocking, R.: How people learn: Mind, brain, experience, and school. National Research Council, Washington, DC (1999)
5. Graesser, A., Ozuru, Y., Sullins, J.: What is a good question? In: Bringing reading research to life. Guilford Press (2010)
6. Heilman, M., Smith, N.A.: Extracting simplified statements for factual question generation. In: Proceedings of QG 2010: The Third Workshop on Question Generation, p. 11 (June 2010)
7. Hirashima, T., Yamasaki, K., Fukuda, H., Funaoi, H.: Kit-build concept map for automatic diagnosis. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 466–468. Springer, Heidelberg (2011)
8. Husbands, C.: What is history teaching?: Language, ideas and meaning in learning about the past. Open University Press, Berkshire (1996)
9. Kashihara, A., Taira, K.: Developing Navigation Planning Skill with Learner-Adaptable Scaffolding. In: Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, pp. 433–440. IOS Press (July 2009)
10. Leelawong, K., Biswas, G.: Designing learning by teaching agents: The Betty's Brain system. International Journal of Artificial Intelligence in Education 18(3), 181–208 (2008)
11. Mostow, J., Jang, H.: Generating diagnostic multiple choice comprehension cloze questions. In: Proceedings of the Seventh Workshop on Building Educational Applications Using NLP, pp. 136–146. Association for Computational Linguistics (June 2012)
12. Otero, J.: Question generation and anomaly detection in texts. In: Handbook of Metacognition in Education, pp. 47–59 (2009)
13. Riley, M.: Into the Key Stage 3 history garden: Choosing and planting your enquiry questions. In: Teaching History. Historical Association, London (2000)
14. Smart, L.: Using I.T. in primary school history. Cassell, London (1996)
15. Stow, W., Haydn, T.: 7 Issues in the teaching of chronology. Issues in History Teaching 83 (2000)

Data-Driven Program Synthesis for Hint Generation in Programming Tutors

Timotej Lazar and Ivan Bratko

Faculty of Computer and Information Science, University of Ljubljana, Slovenia
{timotej.lazar,ivan.bratko}@fri.uni-lj.si

Abstract. One of the main functions of intelligent tutoring systems is providing feedback to help students solve problems. We present a novel approach to program synthesis that can be used as a basis for automatic hint generation in programming tutors. Instead of using a state-space representation of the problem-solving process, our method finds a set of textual edits commonly used by students on program code. Given an incorrect program it then synthesizes new programs by applying sequences of edits until a solution is found. The edit sequence can be used to provide hints with varying levels of detail. Experimental results confirm the feasibility of our approach.

Keywords: programming tutors, hint generation, program synthesis.

1 Introduction

Providing interactive feedback is one of the defining features of intelligent tutoring systems. Feedback serves as a mechanism for both instruction and motivation, by explaining misconceptions behind student errors and providing guidance when students are stuck. In order to give useful feedback, a tutor must have some knowledge of the problem-solving process. Modeling this process is particularly challenging in open-ended domains such as programming, where it cannot easily be decomposed into a well-defined sequence of independent steps.

Most programming tutors use manually constructed domain models. Such models can be very effective, but are difficult to create and are only usable for a limited set of problems. Data-driven tutors build a domain model automatically by analyzing past student attempts. They can generate hints for new exercises after enough students have solved them, and improve the quality of feedback as more solutions are observed. Existing data-driven approaches model student actions in terms of changes to program structure (e.g. adding and removing nodes in the abstract syntax tree), resulting in a layer of abstraction between the tutor's model and what the students are actually doing – editing program code. While this gives a useful high-level view of a program's evolution, it also limits the granularity with which individual changes can be tracked.

We propose a new data-driven approach that models programming directly in terms of textual edits, allowing us to trace student actions more closely. Our method is generative: given an incorrect program, it finds a sequence of edits

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 306–311, 2014.

that transforms it into a correct solution. We are implementing a Prolog tutor based on our approach. It uses almost no language-specific information and can potentially be adapted for other programming languages.

This paper presents the method for synthesizing new programs from an incorrect solution. The next section gives an overview of related work; our approach is described in Sect. 3 and evaluated in Sect. 4, while the final section concludes and presents directions for further work.

2 Related Work

A substantial amount of work exists concerning feedback in programming tutors. Typical approaches describe the solution to each problem using either a set of constraints [1,2] or reference programs [3,4]. Hints are generated by analyzing the differences between solution description and the student's code. The tutor must be programmed with specific information for each supported exercise.

With increased use of technology in education, large amounts of educational data are becoming available [5]. Data-driven approaches exploit this data to "learn" how to solve problems from actual student solutions, reducing the need for expert input. Typically, the problem-solving process is modeled as a search through a state space of partial and complete solutions. Any path from the starting state to a goal state corresponds to a sequence of actions solving the problem. Goal-oriented feedback is generated by finding the most likely transition from the current state toward a goal state. The Deep Thought logic tutor constructs a Markov decision process for each problem from past student solutions [6].

A similar approach can be used for programming tutors. However, representing each program by a separate state is intractable even for the simplest problems. Linkage graphs [7] and program canonicalization [8] have been used to reduce the state-space size by grouping equal or similar programs.

3 Text-Based Program Synthesis

Given a programming exercise and an incorrect program, the task is to find a sequence of transformations that fixes the program. We model programming as a line-oriented text-editing process, and use *line edits* as basic operations for transforming programs. For example, line edits commonly used by students when programming the predicate `del/3`[1] in Prolog include

```
del(A,[B|C],D) → del(A,[B|C],[B|D]) ,
del(A,B,C)     → del(A,[A|D],D) and
C=del(A,B)     → del(A,B,C) .
```

In general, a line edit $u \to v$ replaces a line matching u with v. If u or v is empty, the edit inserts or removes the whole line.

[1] Predicate `del(X,L,L2)` holds iff the list `L2` equals `L` with one occurrence of `X` removed.

To discover commonly occurring line edits, we store the interaction history called *trace* for every attempt. Each trace includes the complete sequence of characters the student inserted and removed. We find line edits in a trace by splitting the sequence of actions into contiguous blocks modifying the same line. The last example above corresponds to a block of four actions removing the characters `C=` and inserting `,C`. The order of actions within a block is irrelevant.

We count the number of times each line and line edit appears in the set of all traces. To reduce noise, we tokenize the left- and right-hand sides of every edit and standardize variable names. For example,

```
del( E, List, New) → del( E, New, List)
```

becomes

```
del(A,B,C) → del(A,C,B) .
```

This way edits from all traces can be compared while ignoring variations due to whitespace, comments and identifier names. When applying edits to a student's program, standardized names must be mapped to actual variables in the affected line. Finally, we calculate the conditional probability of applying each edit $u \to v$ given a line matching u as

$$P(u \to v|u) = \frac{\#\text{ of times } u \to v \text{ appears all traces}}{\#\text{ of times } u \text{ appears in all traces}} \ . \tag{1}$$

3.1 Search Algorithm

After we have found a catalog of line edits, we can use them to correct students' submissions. Given an incorrect program p_0, the goal is to find a sequence of line edits that transforms it into a working program. This is done as a best-first search among sequences of line edits.

A priority queue of potential solutions is maintained. For each program in the queue we also store the sequence of edits needed to reach it from p_0 and its score, defined below. We initialize the queue by adding p_0 with score 1.

In each iteration of the algorithm we remove a program p with the highest score s_p from the queue. If p passes all tests (see Sect. 4), we are done. Otherwise we search the catalog for edits $u \to v$ where the left-hand side u matches a line l in p. We apply each such edit to p to obtain a new program p', and add p' to the queue with the new score $s_{p'}$ calculated as

$$s_{p'} = s_p * P(u \to v|u) \ , \tag{2}$$

where $P(u \to v|u)$ is the conditional probability defined by (1). This way shorter sequences of common edits receive higher scores and are thus considered sooner. A more sophisticated scoring function would take into account additional features of p and p', and likely yield better results. However, (2) works sufficiently well to demonstrate the potential of our approach.

3.2 Hint Generation

To use our method in a programming tutor, we must be able to provide hints based on a sequence of line edits. For example, our method finds the edits (shown in comments) required to fix the buggy implementation of `conc/3`[2] containing two typical mistakes:

```
conc(L,[],L).          % conc(L,[],L) → conc([],L,L)
conc([H|T],L,L2) :-    % conc([H|T],L,L2) → conc([H|T],L,[H|L2])
  conc(T,L,L2).
```

Line edits themselves can serve as bottom-out hints. However, a tutor must also be able to guide the student toward a solution without giving it away. Given a sequence of edits, we can highlight the affected lines, tokens or variables. By parsing the program we could generate descriptive hints, e.g. "Last argument in the head of the second rule should be a list" for the above program. If the student must insert or remove a rule or a subgoal, a message can be shown to that effect. Some help can be provided even for programs our method is unable to fix, by marking lines or identifiers that appear in few or no existing solutions.

4 Evaluation

We collected traces of students' attempts using a modified version of the tuProlog environment[3]. Each trace contains at least one correct version of the program submitted for testing. Most traces also include several incorrect submissions. Correctness of a program is determined by running it against a set of manually selected queries. Programs are small and simple so this gives accurate results.

We evaluated the algorithm on several problems. For each problem we used our method to fix all incorrect student submissions. We limited the time to find a solution to 10 seconds; a longer timeout would make it impractical for real-time use. Our method can fix 50–70% of incorrect programs for most problems; results are shown in Table 1. The success rate for `conc/3` is likely lower because it is

Table 1. Percentage of incorrect programs fixed by our method

Problem	Traces	Incorrect	Fixed	%
conc/3	93	83	29	0.35
del/3	84	100	55	0.55
duplicate/2	50	83	51	0.63
is_sorted/1	55	118	70	0.59
length/2	68	36	26	0.72
palindrome/1	56	85	58	0.68
shiftleft/2	60	66	41	0.62

[2] Predicate `conc(L1,L2,L)` holds iff the list `L` is a concatenation of `L1` and `L2`.
[3] `https://apice.unibo.it/xwiki/bin/view/Tuprolog/`

the first problem dealing with lists the students solved; hence, many programs contain unique mistakes due to confusion about Prolog syntax. Some classes of errors, such as typos, are specific to individual traces and are generally handled poorly by our method, which considers frequently occurring edits first.

The histogram in Fig. 1 shows how many solutions were found after generating 10, 20, 30,... programs, for 229 corrected submissions. Our method behaves as expected and handles the common cases better. We can correct typical bugs by generating less than 50 programs. Submissions with typos, incorrect syntax or multiple errors are more difficult to fix.

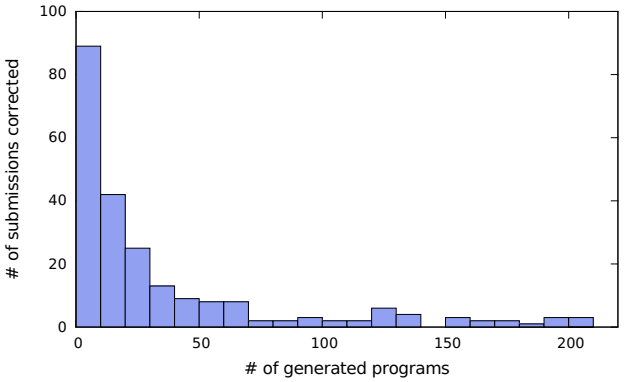


Fig. 1. Number of programs generated before finding a solution

A useful advantage of our method is that it does not have to "understand" a program to fix it. This allows us to find known mistakes in novel programs that differ from all previous submissions. For example, the following implementation of `conc/3`.

```
conc(L1,L2,L) :-
  L1 = [],
  L = L2
  ;
  L1 = [H|T],
  conc(T,L2,TL),
  L = [H,TL].      % L = [H,TL] → L = [H|TL]
```

is the only solution that uses explicit matching and the `;` operator. Our method finds an edit that fixes the buggy line while disregarding other parts of the program. A similar result could be achieved by normalizing the buggy program to use separate rules and implicit matching, like the one in Sect. 3.2. This would require a more detailed model of the Prolog language. Our method works without relying on such a model, although it might be enhanced by it.

5 Conclusion

We have presented a novel approach to hint generation in programming tutors. Our method synthesizes solutions by searching for a sequence of line edits that fixes a buggy program. This sequence can be used as a basis for providing hints. The main advantages of our method are: (a) the set of line edits is learned automatically from past student attempts, (b) it can handle completely novel programs that do not map to a known approach, and (c) relative independence from the target programming language. An unoptimized implementation of our method was able to fix up to 70% of incorrect student submissions.

Besides implementing hint generation in an actual tutor, our future work will consist mainly of improving the search algorithm for program synthesis. When searching for edit sequences, the scoring function only considers edits in the current sequence. This will be improved to also include an estimated distance to a solution. While accurately estimating the "wrongness" of an incorrect program is in general impossible and in any case difficult, a few rudimentary rules-of-thumb can greatly reduce the size of the search space. Options include classifying lines according to their function, and taking inter-line dependencies into account when calculating probabilities.

References

1. Mitrovic, A., Martin, B., Mayo, M.: Using evaluation to shape ITS design: Results and experiences with SQL-Tutor. User Modeling and User-Adapted Interaction 12(2-3), 243–279 (2002)
2. Le, N.-T., Menzel, W.: Using weighted constraints to diagnose errors in logic programming - the case of an ill-defined domain. International Journal of Artificial Intelligence in Education 19(4), 381–400 (2009)
3. Hong, J.: Guided programming and automated error analysis in an intelligent Prolog tutor. International Journal of Human-Computer Studies 61(4), 505–534 (2004)
4. Gerdes, A., Jeuring, J., Heeren, B.: An interactive functional programming tutor. In: ITICSE 2012, pp. 250–255. ACM (2012)
5. Koedinger, K.R., Brunskill, E., Baker, R.S., McLaughlin, E.A., Stamper, J.C.: New potentials for data-driven intelligent tutoring system development and optimization. AI Magazine 34(3), 27–41 (2013)
6. Barnes, T., Stamper, J.: Automatic hint generation for logic proof tutoring using historical data. Educational Technology & Society 13(1), 3–12 (2010)
7. Jin, W., Barnes, T., Stamper, J., Eagle, M.J., Johnson, M.W., Lehmann, L.: Program representation for automatic hint generation for a data-driven novice programming tutor. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 304–309. Springer, Heidelberg (2012)
8. Rivers, K., Koedinger, K.R.: Automatic generation of programming feedback; A data-driven approach. In: Artificial Intelligence in Education, pp. 50–59 (2013)

Building Games to Learn from Their Players: Generating Hints in a Serious Game

Andrew Hicks, Barry Peddycord III, and Tiffany Barnes

North Carolina State University,
Raleigh, NC
{aghicks3,bwpeddyc,tmbarnes}@ncsu.edu

Abstract. This paper presents a method for generating hints based on observed world states in a serious game. BOTS is an educational puzzle game designed to teach programming fundamentals. To incorporate intelligent feedback in the form of personalized hints, we apply data-driven hint-generation methods. This is especially challenging for games like BOTS because of the open-ended nature of the problems. By using a modified representation of player data focused on outputs rather than actions, we are able to generate hints for players who are in similar (rather than identical) states, creating hints for multiple cases without requiring expert knowledge. Our contributions in this work are twofold. Firstly, we generalize techniques from the ITS community in hint generation to an educational game. Secondly, we introduce a novel approach to modeling student states for open-ended problems, like programming in BOTS. These techniques are potentially generalizable to programming tutors for mainstream languages.

Keywords: Serious Games, Hint Generation, Data-Driven Methods.

1 Introduction

BOTS is a serious game designed to teach basic programming concepts to novice computer users and programmers [6]. BOTS, in its current state, contains no mechanisms for personalized feedback or problem ordering. One method of providing such feedback is to have experts create it for each problem. However, BOTS features open-ended problems with many possible solutions, as well as user-generated problems, making such expert annotation difficult. In this paper, we describe an effort to incorporate ITS-like personalization through data-driven hint generation.

Our contributions in this work are twofold. We generalize ITS hint-generation techniques to an educational game, and introduce a novel approach to modeling student states for open-ended programming problems. It is our hope that these techniques can be further generalized to programming tutors for mainstream languages in future work.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 312–317, 2014.

2 Prior Work

Intelligent Tutoring Systems (ITS) have been shown to be effective at improving student performance [1,8]. ITS originally relied heavily on subject matter experts to anticipate common mistakes and misconceptions, but in spite of subject matter knowledge, experts are not always able to detect difficulties or misconceptions (the "expert blind spot") [10]. Additionally, such content is very costly, with Murray [9] estimating a cost of around 300 expert hours to create one hour of content in an ITS. Data-driven methods are proposed as a way of combating these effects, and can provide students individualized help based on previous observations.

The developers of Deep Thought (a propositional logic tutor) employed a method called Hint Factory [12]. As users work on problems, their actions are used to build a Markov Decision Process (MDP). This was later generalized by Eagle, et al, defining an Interaction Network as a complex network containing data about student-tutor interactions. [4] Hints can be generated from this data by searching the Interaction Network for users with the same solution path. Based on the previous users' actions, a potential next step can be suggested. If no user has succeeded on that path before, wec an suggest the current user try a different approach.

Systems such as the Lisp Tutor [1] and ACT Programming Tutor[3] were developed using knowledge engineering. Recent attempts to automate programming tutors have started with hint generation; however, when compared with domains like Propositional Logic, representing programming using state-action pairs poses many more challenges. For example, equivalent solutions to a problem can be expressed in many different ways. Directly applying Stamper's Hint Factory could result in a sparse state space, and we would need many more records in order to provide hints to most students. Some approaches have attempted to condense these similar solutions. One approach converts solutions into a canonical form by strictly ordering the dependencies of statements in a program [11]. Another approach compares *linkage graphs* modelling how a program creates and modifies variables, with nested states created when a loop or branch appears in the code [7].

3 Context

In BOTS [6], the goal of each puzzle is to program a robot to move blocks into specific 'goal' positions on the map. The player controls a robot by writing a program in a graphical, drag-and-drop programming language, as shown in Figure 1. The language supports basic robot operations (move, turn, pick up block) and flow control constructs (variables, loops, and functions). Once the puzzle is completed and the solution terminates without an error, the player is given a score based on the number of instructions they have used. After completing the puzzle, the player is encouraged to make modifications to their program and complete the level again using fewer instructions.

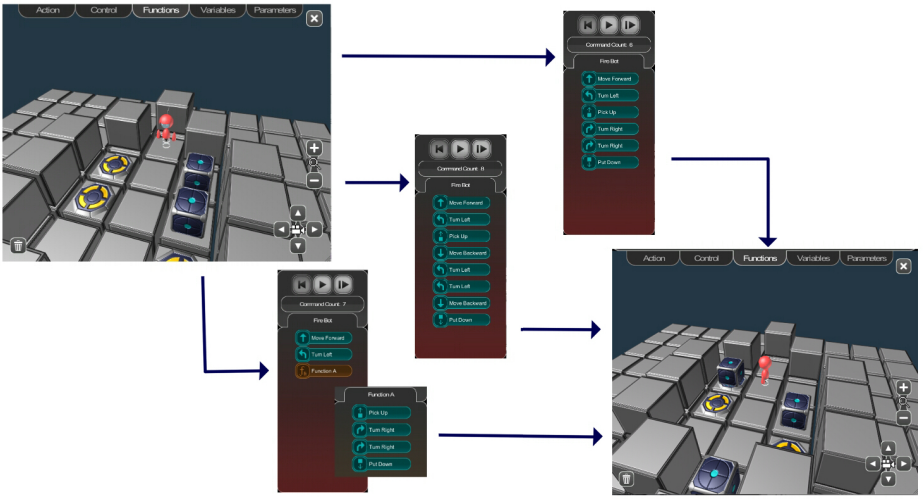


Fig. 1. In the game, players direct a robot to solve puzzles using a simple drag-and-drop programming language. Here we see three different programs which result in the same final state: The robot has moved a block from one side of the room to the other.

4 Methods

The intelligent tutoring system literature agrees on the definition of *interactions* as the low-level, click-by-click behavior of a student in a tutor [13]. This low-level representation is not ideal for our context, as the state space in BOTS is large and sparsely populated when compared to other tutors which have used Hint Factory. Instead, we will use the output of programs that players have written. For example, in Figure 1, the screenshot depicts the initial configuration on the left, and three distinct programs that each result in the output on the right. In our representation, one "World State" would encompass all three programs. We will show that this representation substantially reduces the state space and also facilitates the generation of meaningful hints.

To develop our alternative model, we first looked to other tutors that use data-driven hint generation, such as Deep Thought, a tutor for propositional logic used in introductory discrete math courses [2,4], and iList, a tutor that teaches the concepts of linked lists [5]. Both of these tutors use Hint Factory [12] to generate hints, but do so with different underlying models of the student states.

An interaction in Deep Thought is a single user input such as selecting a rule to apply. These states are represented as vertices of a graph, with edges between vertices being labelled with the logical rule (modus ponens, modus tollens) that was used to derive the most recently added state. The developers of iList also use Hint Factory, but their underlying model is based on snapshots of the tutor's internal state rather than the sequences of user interactions [5]. The authors look at the *results* of the student actions rather than at the actions themselves,

automatically resolving the situation in which multiple unique sequences result in an identical state. In order to find similar states, the authors compute which internal states are isomorphic to each other. For this work, we represent the output of a student's program as a grid representing the size of the stage, with unique markers for boxes, switches, and robots, as well as a height map of the stage. An example can be seen in Figure 2. This way, regardless of the contents of their programs, students who are performing the same actions (such as putting a particular block on a particular switch) will be grouped into the same state.

5 Analysis

To test the practical applicability of this state representation for analysing student solutions and providing hints, we used a corpus of past data collected from middle school aged players in classes and STEM-related afterschool programs.

Table 1. Results of our method for 24 puzzles. Rows indicate the Puzzle ID, number of students who attempted the puzzle, number of individual attempts, number of unique programs, number of "hintable" output states, and number of unique output states.

Puzzle	Students	Attempts	Unique Programs	Hint-Generating States	Unique States
1	60	95	9	3	5
2	57	284	234	41	65
3	50	189	121	15	21
4	43	77	39	9	9
5	42	181	193	22	24
6	42	84	26	5	7
7	40	127	182	31	41
8	35	50	16	8	10
9	35	89	81	25	29
10	**33**	**227**	**325**	**79**	**130**
11	31	53	77	20	25
12	28	79	41	3	4
13	27	145	187	50	75
14	22	40	57	19	23
15	21	76	119	16	18
16	19	40	96	33	39
17	18	76	103	26	32
18	15	44	59	4	35
19	15	34	64	16	38
20	14	56	43	5	25
21	13	33	34	15	20
22	10	67	71	18	23
23	8	30	25	13	16
24	8	13	32	0	22

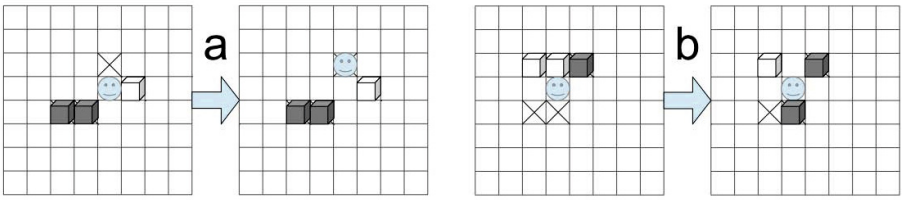


Fig. 2. Two of the generated hints for a simple puzzle. The blue icon represents the robot. The 'X' icon represents a goal. Shaded boxes are boxes placed on goals, while unshaded boxes are not on goals.

5.1 Hint Generation and State Space Coverage

To evalute how much our method was able to improve hint coverage, we compared the number of unique programs written to the number of unique output states. We then considered the number of those states for which a hint was available as shown in Figure 1. For the problems analyzed, our approach was consistently able to reduce the state space. For puzzle 10, a puzzle with a rich data set of solutions, we were able to reduce the state space from 325 unique programs to 130 unique output states. However, this reduction is meaningless unless we are able to provide useful hints from the created states. Out of 130 unique observed states, 79 states had potential to generate hints (that is, a student was in that state and then correctly solved the puzzle). 33 of these hints led to Error nodes, in cases where the Error was the only observed next-step. Of the remaining 45 hints, we found 42 to be meaningful. It is important to note that while this problem contained more records and students than other problems in our data set, the number of records was still quite small. Despite the lack of data we were able to provide hints more than half of the time, and able to provide hints for every state reached by multiple users.

6 Conclusions and Future Work

We have developed an approach to modeling student interaction with a serious game. This approach can be used to automatically generate hints with the Hint Factory algorithm. Rather than attempting to encode the programs or step-by-step interactions of the user, we instead use the resulting configuration of the world after each compilation of the student's code. Doing so, we are able to cover all of the unique code submissions with only a fraction of the states in the graph. While we use a naive implementation of Hint Factory, the hints that are generated are still useful and interesting, particularly those that lead out of error states. This work demonstrates that even with a small number of records, useful hints can be generated by grouping user actions according to their results. A similar system could be used in real-time games, generating hints based on important results or milestones rather than from low-level interaction data.

Acknowledgements. Thanks to the additional developers who have worked on this project or helped with our outreach activities so far, including Aaron Quidley, Veronica Catete, Trevor Brennan, Irena Rindos, Vincent Bugica, Victoria Cooper, Dustin Culler, Shaun Pickford, Antoine Campbell, and Javier Olaya. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 0900860 and Grant No. 1252376.

References

1. Anderson, J.R., Reiser, B.J.: The lisp tutor. Byte 10(4), 159–175 (1985)
2. Barnes, T., Stamper, J.C.: Automatic hint generation for logic proof tutoring using historical data. Educational Technology & Society 13(1), 3–12 (2010)
3. Corbett, A.T., Anderson, J.R.: Student modeling and mastery learning in a computer-based programming tutor. In: Frasson, C., McCalla, G.I., Gauthier, G. (eds.) ITS 1992. LNCS, vol. 608, pp. 413–420. Springer, Heidelberg (1992)
4. Eagle, M., Johnson, M., Barnes, T.: Interaction networks: Generating high level hints based on network community clusterings. In: EDM, pp. 164–167 (2012)
5. Fossati, D., Di Eugenio, B., Ohlsson, S., Brown, C.W., Chen, L., Cosejo, D.G.: I learn from you, you learn from me: How to make iList learn from students. In: AIED, pp. 491–498 (2009)
6. Hicks, A.: Creation, evaluation, and presentation of user-generated content in community game-based tutors. In: Proceedings of the International Conference on the Foundations of Digital Games, FDG 2012, pp. 276–278. ACM, New York (2012)
7. Jin, W., Barnes, T., Stamper, J., Eagle, M.J., Johnson, M.W., Lehmann, L.: Program representation for automatic hint generation for a data-driven novice programming tutor. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 304–309. Springer, Heidelberg (2012)
8. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A., et al.: Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education (IJAIED) 8, 30–43 (1997)
9. Murray, T.: An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. In: Authoring Tools for Advanced Technology Learning Environments, pp. 491–544. Springer (2003)
10. Nathan, M.J., Koedinger, K.R., Alibali, M.W.: Expert blind spot: When content knowledge eclipses pedagogical content knowledge. In: Proceedings of the Third International Conference on Cognitive Science, pp. 644–648 (2001)
11. Rivers, K., Koedinger, K.R.: Automatic generation of programming feedback: A data-driven approach. In: The First Workshop on AI-supported Education for Computer Science (AIEDCS 2013), p. 50 (2013)
12. Stamper, J., Barnes, T., Lehmann, L., Croy, M.: The hint factory: Automatic generation of contextualized help for existing computer aided instruction. In: Proceedings of the 9th International Conference on Intelligent Tutoring Systems Young Researchers Track, pp. 71–78 (2008)
13. Stamper, J., Koedinger, K., Baker, R.S.J.d., Skogsholm, A., Leber, B., Rankin, J., Demi, S.: PSLC datashop: A data analysis service for the learning science community. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part II. LNCS, vol. 6095, pp. 455–455. Springer, Heidelberg (2010)

Evaluation of Guided-Planning and Assisted-Coding with Task Relevant Dynamic Hinting

Wei Jin[1], Albert Corbett[2], Will Lloyd[1], Lewis Baumstark[1], and Christine Rolka[1]

[1] University of West Georgia, Carrollton, GA, USA
wjin@westga.edu, wjin@shawu.edu
[2] Carnegie Mellon University, Pittsburgh, PA, USA
corbett@cmu.edu

Abstract. We describe a programming tutor framework that consists of two configurable components, a guided-planning component and an assisted-coding component that offers task relevant automatically-generated hints on demand to students. We evaluate the effectiveness of the new integrated planning and coding environment by comparing it to three other tutor conditions: planning-only, coding-only, and planning-only interleaved with planning-coding. We conclude that the integrated planning and coding tutor environment is more effective than tutored planning-only activities and that students make more efficient use of tutor feedback in the integrated environment than in the coding only environment.

Keywords: Intelligent tutoring systems, automatic hint generation, programming tutors.

1 Introduction

With the increasing demands for skilled workers in the computing fields, there have been increasing efforts in developing effective and innovative approaches to recruit and retain students in computing majors.

Our approach is to help students learn more effectively. Many educators have observed the difficulties that students have with mastering programming [12]. The high failure rate of introductory programming courses and, as a result, the high drop-out rate from the computing majors during the first two years of college is a commonly known problem in many institutions [6]. It is a manifesto that ineffective learning and the resulting frustration are an important factor in the retention problem.

Our approach is to use computer tutors that help students develop problem solving and program writing skills. It has long been known that one-on-one in-person tutoring with an area expert is most effective [4]. A later study [17] shows that an ITS designed with proper granularity can be just as effective as human tutor.

An effective human tutor is insightful and adaptive. He/she can detect and address the crux of a student's problem. For example, the tutor may decide that a student has not mastered a programming construct, and the tutor will review the programming construct with the student before embarking on the homework problem the student

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 318–328, 2014.

comes for. The tutor may also see that the student is totally stuck in a programming task, even though the student has a good understanding of the relevant programming constructs. The tutor would analyze the problem and help the student plan a solution. With the plan, the student is competent to carry out the coding portion. If the student already has a rough plan in mind, but has difficulty converting the plan to code, the tutor may ask probing questions to help students implement the plan with code.

An adaptive human tutor has a bag of "tools", which he/she chooses according to how to best help a student. In this paper, we describe two such tools/components, which are closely integrated in an automated programming tutor we have developed, and present evaluation results for this new tutor environment. The first tool is guided-planning, where the tutor guides students in both decomposing the program into sub-goals and planning a solution to each sub-goal. The second tool is assisted-coding, where the tutor provides help as needed for the student to code each sub-goal.

The effectiveness of intelligent tutoring system support for writing code is well-established [7][13][18]. Several intelligent programming tutors have also been developed that support interactive support for program planning [5][11][14], but the effectiveness of such planning interactions is not well established.

In the remainder of the paper, we will first describe the guided-planning and assisted coding components. We then present the evaluation results and conclusion.

2 The Guided-Planning Component

The guided-planning component is a step-by-step problem solving process that guides students in the right direction. It consists of two levels of granularities. At the larger granularity, the tutor divides problem solving into a sequence of sub-problems (e.g. variable analysis, flow analysis, input section, computation, output section). At the finer granularity, the tutor guides students in developing a detailed coding plan for each sub-problem, which may include concrete code snippets. Figure 1 shows the tabbed planning interface, as the student designs a solution for this problem: "Write a program that asks the user to enter how many days of vacation they took (integer). The program will print how many weeks and days this is." The figure shows the student working on the second, variable analysis, sub-problem. In the previous, IO analysis, sub-problem, the tutor has guided students in identifying the relevant quantities in the problem that will be assigned to variables. As shown in the figure, at the current sub-problem, the student indicates the data type and assigns a name to each of the variables. The tutor provides right-or-wrong immediate feedback to students. A wrong answer will be highlighted in red font. Students can right-click on the wrong answer for hints as to why it was wrong.

In a planning-only version of the environment, the tutor automatically fills in corresponding program code in the window at the upper right as the student completes each of the planning sub-problems. In the planning-and-coding version, as shown in Figure 1, the student generates the corresponding program code, as described in the following section, after completing each planning sub-problem with the tutor's assistance.

The guided-planning component is created for each tutoring problem by the teacher using a GUI based authoring tool. A tutor engine program renders this component as a tabbed interface, with a tab for each sub-problem. The authoring tool makes it fast to develop the planning specification for a new problem or update the one for an existing problem. Even though the planning component is statically defined, its rendering does not have to be so. Our future work includes extending the tutor engine to allow the tutor to dynamically adjust the planning component's granularity of interaction with students.

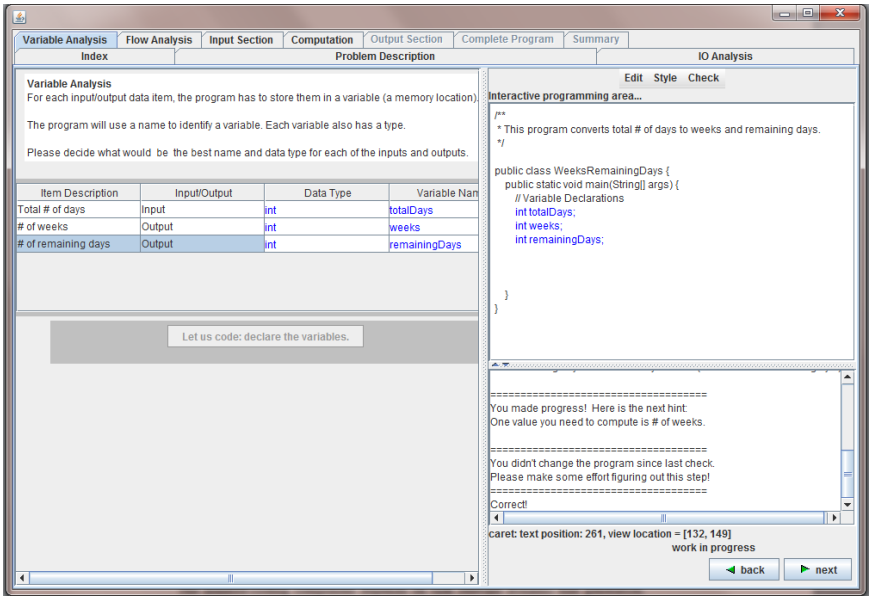


Fig. 1. Tabbed Interface divides problem solving into a sequence of sub-problems

3 Dynamic Hint Generation and the Assisted-Coding Component

The assisted-coding component depends on task relevant dynamic hint generation. We will first discuss dynamic hint generation and then describe how it is used in the assisted-coding component.

3.1 Task Relevant Dynamic Hint Generation

Task-relevant hints are an important component for tutoring systems. Both statically authored hints and dynamically generated hints have been used. Statically authored hints have been used in many tutoring systems for different subjects of study. CTAT uses authoring tools to insert hints into example tracing tutors [1].

Hints generated by the rule-based tutors are a form of dynamic hints [2], however, developing rule-based systems is time consuming. In a second, logic tutor approach, dynamic hints are constructed based on the matching of current student's problem solving path with solutions from previous submitted solutions [3][16]. In our approach, dynamic hints are generated from a single, or at most a few, instructor-provided solution(s).

Dynamic hinting has the advantage of being adaptive; however, it is a challenge for programming tutors. For a programming problem, there are many possible correct solutions. The number of correct solutions is combinatorial in nature and can be infinite if we treat programs with different identifiers as different.

Programming has two distinctively opposite aspects: constraints and freedoms. Some things have to be done a certain order (constraints), but there is a lot of room for creativity (freedoms). For example, programmers have total control over the order of the operations that do not have dependency on each other, naming of identifiers, and the programming constructs to use. The number of possible solutions is combinatorial to the complexity of the problem.

The freedoms a programmer has make it a challenge to automatically generate task-relevant hints. A proper program representation that normalizes the freedom portion of programming is needed. Several promising program representation methods have been proposed [8][9][15]. Our approach is based on the program representation, linkage graph, proposed in [9]. In this approach, a teacher needs to supply a correct solution or several different correct solutions to a programming problem. For each correct solution, the teacher also needs to supply a specification file for variables used in the file.

We have extended this approach in the following ways:

- Instead of one variable specification file per solution, only one specification file per programming problem is needed.
- For a program that a student is working on, there are potentially several different ways to proceed or there are may be more than one logic errors. We developed a module to decide what the best next-step is.
- After the best next-step is decided, the hint presentation module starts with a general hint related to that step, and can progressively provide more detailed hints, including a bottom-out hint.

3.2 Assisted-Coding

Dynamic hint generation is the core of the assisted-coding component. This component can be used alone or together with the guided-planning component. When used alone, after a student is given a problem specification, the student starts coding right away without a planning stage. During the coding process, the student can request hints from the tutor as to what to do next in terms of what is wrong and how to fix the error.

When used together with the guided-planning component, the two components work in an inter-leaved fashion as the student completes a problem. The guided planning component invokes the assisted-coding component after a student finishes the planning activities for a sub-problem. At the GUI interface, on each tabbed page for a

sub-problem, a student performs the planning activities on the left panel and the assisted-coding on the right panel.

Figure 2 shows the coding panel (in the upper right of the full screen) as the student writes the code for the final, output section sub-problem of the example program. The tutor leaves a segment below the current goal ("3. Display the area" in this case) editable and the student write the corresponding code in that segment. All the other lines of the coding panel are not editable. This limits students' attention to the current task at hand.

The coding tutor provides feedback and advice only upon student request. At any time during coding, the student clicks on the check button, requesting the tutor to check if the code segment is correctly done. If there are errors, the tutor provides hints on what the student needs to do. The hints get progressively more specific if the tutor detects that the student is still at the same place after several tries. For example, if a student forgets to declare a variable for input data (e.g. temperature in Celsius), the tutor would first tell the student that a variable for an input data item should be declared. If the student does not fix this problem, the tutor will tell the student to declare a variable for the temperature Celsius. If the student still fails to do so, the tutor will display the declaration statement to the student.

Feedback-on-demand is more compatible than immediate feedback with our method of hint generation, which relies in part on compiling the student's code. [7] showed that feedback on demand is less efficient than immediate feedback for writing full programs, but we hypothesize that the greater freedom offered by feedback-on-demand is more feasible when students are coding one short, pre-planned program segment at a time. To evaluate this hypothesis, in the coding-only version of the tutor, the student codes complete program solutions, again with feedback-on-demand, without assistance in decomposing the program into sub-goals.

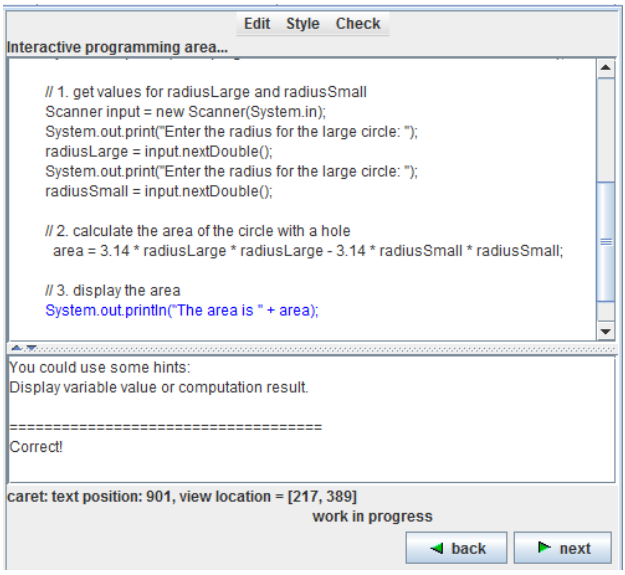


Fig. 2. The right panel of the tabbed interface where students write code

4 Evaluation

Planning and coding are two important aspects of programming. Usually planning is implicitly done during the coding process. Our approach makes planning an explicit stage. In a previous study [10], the authors evaluated how explicit planning had improved student performance in programming. The explicit planning process together with the automated planning tutor used throughout the learning of several programming constructs have yielded substantial gains over traditional instruction. Now with dynamic hint generation and the assisted coding component, we can examine the effectiveness of an integrated guided-planning and assisted-coding intelligent tutor.

4.1 Experiment

The experiment was conducted at the University of West Georgia in four sections of an introductory programming course, which is a first programming class open to students of all majors, but not intended for computer science majors. We conducted the study six weeks into the semester. The study included four conditions:

- Planning-Coding (PC): For each problem, the tutor guides students through planning activities with immediate feedback and then assists students with the coding, with feedback on demand.
- Planning-Only (PO): For each problem, the tutor guides students through planning activities with immediate feedback, then generates the code for students.
- IPOPC: The tutor alternates between Planning-Only and Planning-Coding on successive problems, starting with a PO exercise.
- Coding-Only (CO): Tutor provides no decomposition or planning support. Students write code to solve problems, with feedback on demand.

Each of the four course sections was assigned to one of the conditions. The assignment of students to different sections was done by students themselves during the registration period.

The experiment consisted of a single class session in which students completed a pretest, worked with the tutor for about 40 minutes, and then completed a posttest. The pretest and posttest consisted of two problems in which students were asked to write programs similar to the programs that students would develop in the tutoring session. For students in each condition, we applied pre/posttest balance control and have two test forms with comparable problems. We divide the students into two halves, subgroup A and sub-group B. Their pre/posttests are opposite to each other, i.e. subgroup A's pre/post tests are subgroup B's post/pretests. This is to even out the difficulty difference between pre/posttests.

4.2 Evaluation Results

Eighty-five students who completed both pre/post tests are included for evaluation. There were 20 students in the Planning-Coding (PC) condition, 19 in the Planning-Only (PO)

condition, 14 in the IPOPC condition and 32 in the Coding-Only (CO) condition. The instructor for the four sessions graded all the pre/post tests manually according to the rubrics described below. While the instructor was aware of the four different conditions, the instructor was not informed of any hypotheses concerning which of the conditions would be more or less effective.

Test Performance. Each program on the pretests and posttests was scored with a grading rubric, consisting of the following six categories:

- *Variable declaration:* Whether students have declared an appropriate set of variables for the program.
- *Variable type:* Whether students have set proper variable types.
- *Input:* Whether students wrote correct code to get input from the user.
- *Computation:* Whether students wrote correct arithmetic expressions to calculate the result values.
- *Output:* Whether students wrote correct code to display computation results.
- *Order:* Whether students coded the operations (input, computation and output) in an appropriate order.

Each program was assigned a score between 0 and 2 for each rubric. The six rubric scores were averaged across the two programs in the test, and the total score for a test is the sum of these six averages. The left side of Table 1 displays the average total pretest scores and posttest scores for the four conditions, followed by the learning gains. The right side of the table shows the six component scores for just the posttests. As can be seen, the learning gains in the PC condition are about three times larger than in the other three conditions.

Table 1. Average Test Performance in the Four Conditions: Pretest Scores, Posttest scores, Learning Gains, and Six Component Scores for the Postests

	Full Tests			Posttest Component Scores					
	Pret-est	Post-test	Gain	Var decl	Var type	Input	Comp	Out-put	Order
PC	2.28	5.05	2.78	0.93	0.86	0.86	0.71	0.70	0.99
PO	3.28	4.20	0.95	0.86	0.77	0.70	0.56	0.59	0.73
IPOPC	3.48	4.25	0.77	0.86	1.00	0.66	0.48	0.50	0.75
CO	2.72	3.72	0.91	0.75	0.84	0.75	0.31	0.41	0.66

We first performed an ANOVA on just the pretest scores, and the differences among the four conditions were not significant. We then performed a repeated-measures ANOVA on the pretests and posttests, with two between-student factors, condition and test form (i.e. subgroup A or B). In this ANOVA, the main effect of the repeated test measure was significant, $F(1,77)=18.301$, $p < .01$, indicating that the overall learning gains are reliable. The main effects of condition and of test form were not significant. More importantly, the interaction of condition and the repeated test

measure was marginally significant, $F(3,77)=2.212$, $p < .10$, indicating that the learning gains were larger in some conditions than in others.

To examine this interaction further, we performed an ANCOVA on just the posttest scores, with condition and test form as factors, and pretest as a covariate. The main effect of condition in this analysis is not significant, $F(3,76)=3,76$, $p < .16$. However, three pairwise planned comparisons in this analysis revealed that posttest scores in the planning and coding (PC) condition were significantly higher than in the coding-only (CO) condition, $p < .05$, and marginally higher than in the planning only (PO) condition, $p < .09$ and marginally higher than in the interleaved (IPOPC) condition, $p < .08$.

Finally, we inspected each of the six posttest coding heuristics individually. As shown in Table 1, the differences among the groups on the first three, relatively easy components are small, while the differences among the groups are larger on the harder, final components. We collapsed the first three heuristics into a single composite measure and the last three heuristics into a second composite measure, and performed a repeated measure ANCOVA on these two composite scores, with condition and form as factors, and pretest as the covariate. Again, the effect of condition is not significant, but the main effect of the repeated composite heuristic scores is significant, $F(1,76)=19.581$, $p<.01$ and, again, most importantly, the interaction of heuristic and condition is significant, $F(3,76)=4.312$, $p<.01$. In a follow up ANCOVA on just the second composite measure, the main effect of condition is significant, $F(3,76)=3.054$, $p <.05$, and the three planned comparisons of the PC condition with each of the other three conditions are all significant at the .05 level. In contrast, in a follow up ANCOVA on the first composite measure, neither the main effect of condition nor any of the three planned comparisons were significant.

Table 2. Tutor Log Summary

	Average # Questions Finished	At least one problem completed	At least two problems completed
PC	0.75	65%	10%
PO	5.42	100%	100%
IPOPC	1.36	86%	50%
CO	0.03	3%	0

Tutor Performance. Table 2 displays the average number of tutor problems the students completed in each condition. (The tutor only records a log file when the student completes a problem in the first three conditions and does not log partially completed problems. In the last, coding-only condition, log entries were saved after the 2nd hint request, then after every 8 subsequent hint requests, and at the end of a problem.) As can be seen, the number of problems completed varied widely across conditions, but across the board these students in this study struggled in completing the problems.

In the PC condition, 13 of 20 students completed at least one problem correctly during the 40-minute session, and 2 students completed a second problem. In sharp contrast, only 1 student of 32 in the CO condition completed a correct problem. This striking

difference in the number of students who successfully completed a relatively short programming problem with feedback on demand, confirms that this feedback mode is more feasible when students are coding short, pre-planned program sub-goals one at a time, rather than coding complete programs.

Further inspection of the log files revealed some of the disadvantages of the CO condition. Seven of the 32 CO students made poor use of the tutor's hint capabilities, issuing fewer than 2 hint requests (thereby failing to generate log files). Of the remaining 25 students, 9 may have effectively given up; they stopped asking for tutor hints with substantial time remaining in the session (from 6.5 to 15.4 minutes). This suggests that subtask decomposition, and the students' accompanying sense of progressive accomplishment, is key to viability of feedback-on-demand in programming.

At the other extreme, the 19 students in the PO condition, averaged more than 5 complete problems. In fact, all PO students completed at least 3 problems and only 4 students failed to complete all 6 problems in the curriculum. Although the planning template is reasonably close to surface code, the guided-planning activity alone does not translate into enhanced performance on the programming posttest. Instead of the number of exercises students complete, it seems that the types of exercises that students have worked on and completed matter more.

Finally, in the IPOPC condition, 12 of 14 students completed the first, planning-only problem, while 50% finished the second planning-and-coding problem, and no one finished additional problems. Note that only 50% of IPOPC students finished one PC problem, compared with 75% of PC students finished one or two PC problems. Again, these results along with the posttest outcomes suggest that subdividing the work so that the student actively plans the code and the tutor provides the code is not a viable alternative to requiring the student to actively write the code.

5 Conclusion and Future Work

This paper evaluates a newly developed guided-planning and assisted-coding (PC) intelligent programming tutor. Students working with this tutor achieved larger pretest-posttest learning gains than students working in a planning-only (PO) environment, a coding-only (CO) environment or in an environment that interleaves planning-only and planning-coding environments (IPOPC).

The follow up tutor log file analysis indicates that the integrated planning and coding activities in the PC environment are intrinsically more effective than the PO and IPOPC environments. Students completed more tutor problems in the latter two environments, but still did not learn as much as students using the PC environment. This demonstrates that code planning activities are not sufficient for students to succeed. After planning programs, writing the program is better than viewing a computer-generated program. This is true even in the current environment, in which the planning template is reasonably similar to program code.

The tutor log file analyses also indicate that the PC environment is more effective than the CO condition, because it is more efficient; that is, students were more likely to plan and code a complete correct program in the PC than to code a complete correct program in the CO condition. Both environments employed feedback-on-demand

and at least part of the advantage of the PC environment is that students were able to exercise their enhanced feedback control more efficiently within the sub-goal coding structure imposed by the PC environment.

Future work will include examining the separate contributions of sub-goals, sub-goal planning, and feedback mode to the enhanced learning gains observed in the integrated planning and coding environment. Future work will also include expanding the new environment, by extending the dynamic hinting module in the assisted-coding component to include most programming constructs for introductory programming courses, such as selection, methods and classes.

Acknowledgement. This work was partially supported by the National Science Foundation under Grant No. 0837505 (awarded to Shaw University). Our thanks to Yingqi Wang for assistance in developing the tutors and to Bob Kraut for advice on the manuscript.

References

1. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: Example-Tracing Tutors: A New Paradigm for Intelligent Tutoring Systems. International Journal of Artificial Intelligence in Education 19, 105–154 (2009)
2. Anderson, J.R., Conrad, F.G., Corbett, A.T.: Skill Acquisition and the LISP Tutor. Cognitive Science 13(4), 467–505 (1989)
3. Barnes, T., Stamper, J.: Automatic hint generation for logic proof tutoring using historical data. Journal Educational Technology & Society 13(1), 3–12 (2010); Special issue on Intelligent Tutoring Systems
4. Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher 13, 4–16 (1984)
5. Bonar, J., Cunningham, R.: BRIDGE: An intelligent tutor for thinking about programming. In: Self, J. (ed.) Artificial Intelligence and Human Learning, ch. 24, pp. 391–409. Chapman and Hall (1988)
6. Coheen, J., Chen, L.Y.: Migrating out of computer science. Computing Research News 15(2) (2003)
7. Corbett, A.T., Anderson, J.R.: Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes. In: ACM Conference on Human Factors in Computing Systems, pp. 245–252 (2001)
8. Huang, J., Piech, C., Nguyen, A., Guibas, L.: Syntactic and Functional Variability of a Million Code Submissions in a Machine Learning MOOC Stanford University. In: 1st Workshop on Massive Open Online Courses at the 16th Annual Conference on Artificial Intelligence in Education (2013)
9. Jin, W., Barnes, T., Stamper, J., Eagle, M.J., Johnson, M.W., Lehmann, L.: Program Representation for Automatic Hint Generation for a Data-Driven Novice Programming Tutor. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 304–309. Springer, Heidelberg (2012)
10. Jin, W., Corbett, A.: Effectiveness of Cognitive Apprenticeship Learning (CAL) and Cognitive Tutors (CT) for Problem Solving Using Fundamental Programming Concepts. In: 42nd ACM SIGCSE Technical Symposium on Computer Science Education, pp. 305–310 (2011)

11. Lane, H.C., VanLehn, K.: Teaching program planning skills to novices with natural language tutoring. Computer Science Education 15(3), 183–201 (2005)
12. McCracken, M., et al.: A multi-national multi-institutional study of assessment of programming skills of first-year CS students. SIGCSE Bulletin 34(11) (March 2002)
13. Mitrovic, A., Ohlsson, S.: Evaluation of a Constraint-Based Tutor for a Database Language. International Journal of AI in Education 10(3-4), 238–256 (1999)
14. Pirolli, P.: A cognitive model and computer tutor for programming recursion. Human Computer Interaction 2, 319–355 (1986)
15. Rivers, K., Koedinger, K.R.: Automatic Generation of Programming Feedback: A Data-Driven Approach. In: The Workshops at the 16th International Conference on Artificial Intelligence in Education, pp. 50–59 (2013)
16. Stamper, J., Barnes, T., Croy, M.: Enhancing the automatic generation of hints with expert seeding. Intl Journal of AI in Education 21(1), 153–167 (2011)
17. VanLehn, K.: The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. Educational Psychologist 46(4), 197–221 (2011)
18. Weber, G., Brusilovsky, P.: ELM-ART: An adaptive versatile system for Web-based instruction. International Journal of Artificial Intelligence in Education 12(4), 351–384 (2001); Special Issue on Adaptive and Intelligent Web-based Educational Systems

Automating Hint Generation with Solution Space Path Construction

Kelly Rivers and Kenneth R. Koedinger

Carnegie Mellon University, Pittsburgh, PA
{krivers,koedinger}@cs.cmu.edu

Abstract. Developing intelligent tutoring systems from student solution data is a promising approach to facilitating more widespread application of tutors. In principle, tutor feedback can be generated by matching student solution attempts to stored intermediate solution states, and next-step hints can be generated by finding a path from a student's current state to a correct solution state. However, exact matching of states and paths does not work for many domains, like programming, where the number of solution states and paths is too large to cover with data. It has previously been demonstrated that the state space can be substantially reduced using canonicalizing operations that abstract states. In this paper, we show how solution paths can be constructed from these abstract states that go beyond the paths directly observed in the data. We describe a domain-independent algorithm that can automate hint generation through use of these paths. Through path construction, less data is needed for more complete hint generation. We provide examples of hints generated by this algorithm in the domain of programming.

Keywords: automatic hint generation, feedback, learning path construction, solution space, programming tutor.

1 Introduction

We have seen a recent boom of interest in educational technology through the emergence of Massive Open Online Courses and the creation of many educational technology start-up companies. Much emphasis there has been on providing students access to high quality lectures, but there is great unmet promise to better scale the kind of learn-by-doing support that intelligent tutoring systems can provide. A key barrier is the difficulty in authoring intelligent tutors and a key opportunity is the use of past student solution data to ease that development process.

The use of historical student data in generating hints has been examined before [1], but in previous work, solution paths were entirely collected from students. This limited the options for future students who would request hints from such systems, requiring them to stay on the paths that their predecessors had travelled. In such a system, a student who tried to go off-path would not be able to receive hints, even if they were not particularly far from a solution.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 329–339, 2014.

Furthermore, relying entirely on historical student data makes it difficult to generate hints for new problems that do not have collections of data, which makes building tutors for new problems nearly impossible. Therefore, we examine the problem of whether it is possible to construct new student solution paths automatically, using only the solution states that have already been collected from previous students. We center this problem around programming solutions, as programming problems provide a large solution space to work in that cannot be mapped out by hand.

In this paper, we focus specifically on the problem of **path construction**: how do we find a set of states that can lead a student from her current state to a correct state if no paths have been generated for that state before? With path construction, tutors would not need to rely on previous solution paths, though it could benefit from them; it could theoretically generate hints even for a problem which has only been given a few correct examples. If we can identify a path from the new solution to a correct state, it is possible to construct a hint for the student based on the steps taken within that path. We first examine the relevant work in the field of automatic hint generation, then frame the problem by defining relevant features in the domain that the problem is based in. Finally, we elaborate on the algorithm used to do the path construction, and evaluate it on a dataset of real student solutions.

2 Related Work

As mentioned above, researchers have already examined the problem of generating hints automatically based on historical student data. The Hint Factory [1] builds solution spaces out of data recorded from students' past work and has been applied in the domain of propositional logic proofs. This system constructs solution paths, that is, sequences of solution steps students enter in the interface, by combining all the steps that prior students have taken into a graph. Hints can be provided to new student solution attempts as long as they exactly match a step previously taken and stored in this graph. Because the search space in propositional logic proofs is reasonably constrained, the stored graph provides good coverage of the possible solution space. As long as a student's solution steps stay within the graph, hints can be provided. In practice, the Hint Factory was demonstrated in the logic proof domain to generate hints for students who asked for them about 80% of the time. In other words, the system provides tutoring in the majority of situations without any AI programming. Ideally, we would like an intelligent tutor to provide hints for the other 20% of requests as well. Even more challenging, we would like to see this data-driven approach to developing intelligent tutoring extend to domains with much larger solution spaces.

A different approach to generating feedback for new states is to cluster or abstract the original states into equivalent groups, and provide feedback based on which cluster the new state falls into. There have also been attempts to utilize clusters in larger graphs, so that feedback could be propagated out from [4] or compared to [3] a most common correct solution in order to generate feedback for

many solutions with little work. There is great promise in the use of clustering to provide feedback on students' solution attempts, but clustering does pose a challenge to providing detailed feedback that is personalized to a student's particular solution, especially in domains where there can be great variety in solutions. It is also hard to tell how solutions which do not fit into the space could be paired with a single cluster. Most importantly, while clustering facilitates providing feedback on what's wrong with a solution, it does not, by itself, provide students with hints as to a reasonable next-step they could take (based on their solution so far) when they are stuck; it can only take a student directly to the known solution. Providing next-step hints is an important, powerful feature of intelligent tutoring systems [7].

3 The Domain

In this paper, we describe how path construction would work in the domain of computer science, where each solution state is a program. The technique itself can be extended into other domains, however, assuming that a few constraints are met. Therefore, in this section we detail the features required for our algorithm to generate feedback within a domain.

First, there must be a **collection of solution states**, where each state is represented as a tree structure and has data on how many students have generated it before. The tree should contain enough data about the student solution that it can represent the student's work accurately without requiring every detail. Our states are the abstract syntax trees of the programs submitted by students. In the case that hints need to be generated for a new problem, this collection could be composed of a few correct solutions generated ahead of time as exemplars.

Second, there must be a **method for testing solution states**, *test(s)*, which returns a number between 0 (completely incorrect) and 1 (correct). In our example, we run multiple test functions over a submitted program and average the results of all the tests. Each test function provides a program with specific inputs and checks whether the returned output is the expected value, and together they provide a range of scores that a student can achieve.

Finally, there must be a **method for comparing solution states**, *diff(a,b)*, which returns a number between 0 (identical solutions) and 1 (completely different). If the states have been stored as trees, it should be possible to build a comparison function for them; in fact, how to do this in a domain-general way has already been explored [5].

It is worth noting that there may be several superficial differences between solution states that should not be accounted for when comparing solutions; it can be helpful to use a canonicalization process [6], which removes syntactic differences while ensuring semantic equality, to ensure that any differences between solutions provide actual semantic meaning. Using a normalizing process has the added benefit of reducing the number of states used in the solution space significantly, while not reducing the range of solution types that the space covers.

The solution states we use as examples in this paper come from final submissions made by students on programming assignments at the introductory programming

course at Carnegie Mellon University. Due to this, our data is quite sparse- few of the solutions in our data set provide true intermediate steps, as most are attempted full solutions (though many have small bugs). Therefore, to generate hints we must rely on our path construction algorithm heavily.

4 Path Construction Method

Due to the huge potential size of the solution space, it is impossible to construct full solution paths for students based on what others have done in the past; there are many options for where the student should go, and it is difficult to specify exactly how they should get there. To more efficiently provide feedback and hints online, we construct an initial **solution space** offline based on the collection of solution states that have already been gathered. A solution space can be thought of as a graph containing the paths that a student might take while solving a problem; the solution states form the nodes inside the graph, and they are connected by edges which express the edits required to move from one state to the next. Some paths within this space are more desirable than others; for example, paths that involve fewer steps to get to a final solution are usually preferable. Other factors may be important in the tutoring context, such as whether one solution or another may be more easily understood by a novice student. In this case, we can use the frequency of a state in previous observations as a heuristic for how useful it may be to a new student.

In the following sections, we describe the path construction algorithm that is used on each of the incorrect states to cumulatively create this solution space. The algorithm can also be used for a new student if they request a hint but have a solution which is not currently in the solution space. Thus, our algorithm not only expands the solution space beyond prior paths, but is also capable of providing hints for student states that have never been observed before.

4.1 Identify the Optimal Goal State

First, given a solution state which is not yet correct (that is, a state which has a test score not equal to 1), we need to find a nearby goal state within the solution space. This state will serve as our approximation for the student's intended final product, and can be used to generate hints that will guide the student towards his or her own goal.

To find the optimal goal, we first iterate through all the correct states in the solution space to find the state which is closest to the current state (i.e., has the lowest diff score). While this state is a possible goal for the student, it is equally plausible that there is a better goal available; after all, while some of the differences between the current state and the new goal may provide corrections, others may only change superficial features, as in Figure 1.

To determine which of the changes between the two states are actually necessary, we generate all the possible **change vectors** between their solution and the goal. We define a change vector to hold a *tree path*, which is a set of nodes

```
def findPattern(dna, pat, start):
    if findAtIndex(dna,pat,start):
        return start
    while(len(dna)>start+len(pat)):
        if findAtIndex(dna,pat,start):
            return start
        else:
            start+=1
```

```
def findPattern(dna, pat, start):
    while start < len(dna):
        check=findAtIndex(dna,pat,start):
        if check ==True:
            return start
        else:
            start += 1
    return -1
```

Fig. 1. In this solution-goal pair, the while loop's test value edit and the addition of the outer return statement are needed, but the removal of the if statement is not necessary

that can be used to find a specific position in a tree, an old subtree that will be removed, and a new subtree that will be added. This change vector can be used to represent the usual edits we wish to perform when modifying trees- additions, which only have a new subtree; deletions, which only have the old subtree; and edits, which use both. Given a solution state and a change vector, we should be able to apply the vector to the state in order to transform it accordingly.

To find all the change vectors between the solution and the goal, we use the diff function to find the nodes where the two trees differ, and return each difference as a vector. In cases where the trees have a set of elements in a single child (such as the body of a function), we can find an optimal matching of the elements according to their types, only deleting and adding lines where it is necessary.

Once all of the change vectors have been found, we begin the process of locating the optimal subset of them which can create a better goal. To do this, we run the test function on the intermediate solution states that result from applying first the changes individually, then all pairs of changes, and so on. If the algorithm can find a state which is correct and closer to the solution state than the current goal, it becomes the new goal state. It is important to note that, in the worst case, this algorithm requires generation of all possible combinations of change vectors (the power set of the original set), which requires exponential time to execute. However, the algorithm can often halt early if it locates a new goal which is closer to the solution than any of the other sets it is investigating.

At this point, an optimal goal for the solution state will have been found. It is worth asking why we can't stop here and simply give a hint to the student based on the difference between their solution and the goal. In some cases, this is a valid solution; for example, showing the student a comparison of their incorrect solution to a close correct version can serve as a valuable example and may indeed improve learning. However, if the algorithm can generate multiple steps for the student by chunking the hints into groups, we can help them focus on individual components of how to improve their solution, which may help them identify similar components when they work on future problems.

4.2 Identify Valid Intermediate States

Once the goal state of the current state has been identified, the set of all change vector combinations between it and the solution is generated. These states represent all possible intermediate states that might be included in the solution

path. However, not all of these states are good states; some combinations of edits might produce states which would not seem reasonable to a student. We identify three properties that are required in possible intermediate states:

- A valid state must be well-formed, compatible with the solution language.
- A valid state must be closer to the goal than the original solution state.
- A valid state must do no worse when tested than the original solution state.

The first two properties are easily defended- there is no sense in telling a student to go to a state that is not well-formed, and there is little point in making a change if it does not move the student closer to the solution. In fact, if the diff function is well made, these two properties should always be met. However, the third property can be debated; sometimes, one needs to break a solution to make it better overall. While it is possible that this sort of backtracking may eventually improve a student's solution, it is unlikely that a student will apply a change if they see that it reduces their score, so we retain this property for the initial version of the algorithm.

4.3 Find the Optimal Change Path

At this point, we have found the optimal goal for the solution state and identified all possible intermediate states between the state and the goal. Now we need to create a path out of the intermediate states to lead from the solution to the goal. To do this, we identify several properties that are desirable in stable next states:

- **Seen Before:** a state which has been seen before is a state which we know is fathomable; otherwise, it would not have been submitted by a student in the past. This does not ensure that the state is good, or even reasonable, but it does provide some confidence in the state's stability.
- **Near Current State:** it is best if a state is close to the student's original solution; this ensures that the student will not need to make too many changes based on the hint. This also gives the student a chance to make further changes on their own, so they don't need to rely on the hints.
- **Well-performing:** a stable next state should do as well on the test cases as possible, to ensure that the student makes good progress.
- **Close to Goal:** the state should be as close to the goal as possible, so as to lead the student directly there.

We combine these four desirable properties to create a **desirability metric** defined by the formula (1). This metric is used to rank possible next states. The weights in the formula can be adjusted to reflect how important each of the properties are within the domain in question. In our data set, we found that some of the properties were modestly correlated (such as closeness to solution and score), and adjusted the weights to account for this double-counting and give preference to shorter hints.

$$0.3 * frequency(s) + 0.4 * (1 - diff(s, n)) + 0.1 * test(n) + 0.2 * (1 - diff(n, g)) \quad (1)$$

After ranking all of the change states by desirability, we can pick the best state- the one with the desirability score closest to 1- and set it as the first state on the path to the solution. Then we identify each of the next states that would follow the first by locating all states between the chosen state and the goal (e.g., all states which have change vectors containing the vectors in the first state) and iterating on this step. This will generate an entire solution path that extends from the original state to the goal.

At this point, the algorithm can be used to generate the next states for any solution state given to it. With this, the solution space can be fully constructed. Now, when a student needs a hint on a problem, we can locate their solution within this solution space and find their next state. Turning this into a human-readable message is beyond the scope of this paper, but can be done by transforming the change vectors to match the student's original solution and framing them within a few simple templates.

5 Evaluation

Our research question in this project focused on whether we could construct new student solution paths automatically using only a collection of prior student solution states. To determine whether we have met this goal we test our system on three metrics: whether it is possible to generate hints for incorrect states that are stored in our solution corpus, how long hint generation takes, and how well-aligned produced hints are to the students' intentions.

For the purpose of this testing, we utilized the solution sets from five different programming problems assigned in the introductory course at Carnegie Mellon University. These problems are all fairly complex, requiring the use of conditionals and loops, and had on average 34.5% of their normalized solution sets composed of incorrect solutions.

5.1 How Often Can We Generate Hints?

Theoretically, we should be able to generate hints for any incoming solution state as long as we have at least one correct solution state in the solution space. After all, in the very worst case we should be able to ask the student to undo all of the work they've done and then take them step by step through the correct solution. While this is not an optimal choice, it does provide a help option for the student, rather than forcing them to work through the problem on their own.

However, the current implementation has an efficiency limitation in the second step of the path construction process where the change vectors are generated. The algorithm uses the power set of the set of all possible edits to find all possible intermediate states, which means that the algorithm grows at an exponential rate. This is not sustainable when the number of edits grows larger; while, in principle,

the algorithm can generate a hint for a student who is far away from all previously seen states, it may not always do so quickly enough to be useful.

To evaluate the extent of this efficiency limitation, we analyzed the incorrect solution states to determine the number of edits between the current solutions and end goals. We compared these to the number of change vectors between the solution and the optimized goal, to see how much the optimization could improve the process. On average, the original goal found in the solution space was about five edits away from the solution. Looking for optimized goals decreased this distance to 2.5; more importantly, however, looking for an optimized goal greatly increased the number of goals that were only one edit away. As is shown in Figure 2, the number of edits required decreases at all levels of edit distance, which means that hints will be better targeted at what the student actually needs to do to fix their solution.

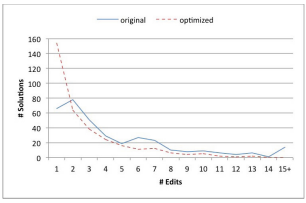


Fig. 2. A comparison of the number of edits between current solution and goal across the dataset. When the goal state is optimized, it is more likely to be only one edit away from the solution.

5.2 How Long Does It Take Generate a New Feedback Message?

As was mentioned before, an automatically generated hint is not particularly useful if the student does not receive it in a timely fashion. This is not a problem if the student's state has been seen before, as the hint will have been stored in the solution space, and can be delivered immediately. However, it is more interesting to look at the cases when the feedback needs to be generated.

In measuring the time taken to generate feedback, we found that the vast majority of solutions are not particularly far from their goals; 60% take less than a second to generate feedback, and 90% take less than a minute. However, we did find a few solution states which took a tremendous amount of time to run, making it infeasible to generate paths online. 12 of the 351 solutions we examined took longer than 20 minutes to run, and all but one of these had 15 or more edits between themselves and the goal. For the rest of the solutions, the time required to run the algorithm was exponentially related to the number of edits between the solution and the goal (see Figure 3). This is related to the power set generated in finding the change vectors, and thus is difficult to address.

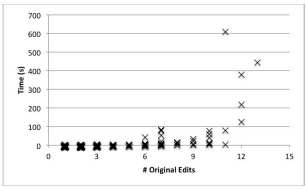


Fig. 3. A comparison of number of edits between current state and end goal and the amount of time it takes the algorithm to generate a hint. The time grows exponentially with the number of edits, though the majority of the states are quick to generate.

5.3 Are the Chosen Goals Aligned with the Students' Real Intentions?

A critical question to address is whether the hints we provide will be well aligned with the students' intentions. We cannot know the student's intention from their submission alone, but we can approximate it by identifying how close the original solution states are to the final goal states. On average, the current and goal states have 64% similarity, suggesting that the solutions are somewhat close.

Since the data set we are using is composed of final submissions, it seems likely that most students should have been close to their final answer when they submitted; therefore, any great distance between a solution and its goal might reflect poor goal choice on our part, rather than a student truly being far away from a correct answer. To determine if this is the case, we examine real hints generated by the algorithm, both in cases where the goal was very close to the solution state and in cases where it was far away.

Our first example, shown to the left in Figure 4, involves a solution that only has one small bug. This solution is two edits away from the closest correct state in the graph, but only one of those edits is necessary; this is represented in the optimal goal that the graph generates. The final hint message generated, "Replace '23456789YJQKA' with '23456789TJQKA' in line 3", pinpoints the bug efficiently. On the other hand, the second example, shown on the right, is very far away from all correct states in the solution space. This would not be

```
def intToPlayingCard(value):
    faceValue = value%13
    face = "23456789YJQKA"[faceValue]
    suitValue = (value-faceValue)%4
    suit = "CDHS"[suitValue]
    return face+suit
```

```
def intToPlayingCard(value):
    suit = "CDHS"
    facen = "23456789TJQKA"
    suitn = ((value) / 13)
    face = (value - (13 * suitn))+2
    findsuit = suit. find [suitn]
    gaga = findsuit + face
    return gaga
```

Fig. 4. Two student solutions to a programming problem on mapping integers to playing cards. On the left, a solution that is close to its goal; the only mistake is a single typo. On the right, a solution that is far away; it requires almost a full rewrite.

a problem if a closer goal could be generated, but the algorithm fails to create one. In examining the code, it is clear why this is the case- the solution has many problems, and hard to map to a goal solution. Gently suggesting that the student start over might be the best we can do.

6 Discussion

As we investigate whether the suggested goals are related to student intentions, we should also question whether the solution paths we are building look anything like solution paths students generate while working on their own. Naturally we do not want to make the student's experience with hints identical to their experience without them; after all, hints are supposed to improve their learning. However, we can test whether the hints provided will seem natural to students.

This question has been explored before in the field of learning analytics, through examination of several detailed case studies of student work [2]. Blikstein found that different students had different methods of approaching programming. Those who mostly focused on writing their own code made small changes while iterating on their approach. Therefore, we should also suggest small steps when possible; if we suggest that a student try a large change, they may not be willing to modify so much of their code at once.

In our future work, we aim to determine whether the hints generated by our system are truly beneficial to students. We are currently taking steps to run a study in a classroom, and plan to use the resulting data to continue improving the path construction system. As we gather more data on problems, we should be able to provide hints that are closer and closer to the students' original goals; and hopefully, with a large enough solution space, we will be able to create messages for all students, regardless of how unexpected their solutions may be.

Acknowledgements. This work was supported in part by Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (# R305B090023).

References

1. Barnes, T., Stamper, J.: Toward automatic hint generation for logic proof tutoring using historical student data. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 373–382. Springer, Heidelberg (2008)
2. Blikstein, P.: Using learning analytics to assess students' behavior in open-ended programming tasks. In: Proceedings of the 1st International Conference on Learning Analytics and Knowledge, pp. 110–116 (2011)
3. Gross, S., Mokbel, B., Hammer, B., Pinkwart, N.: Feedback Provision Strategies in Intelligent Tutoring Systems Based on Clustered Solution Spaces. In: DeLFI 2012: Die 10. e-Learning Fachtagung Informatik, pp. 27–38 (2012)
4. Huang, J., Piech, C., Nguyen, A., Guibas, L.: Syntactic and Functional Variability of a Million Code Submissions in a Machine Learning MOOC. In: AIED 2013 Workshops Proceedings Volume, pp. 25–32 (2013)

5. Mokbel, B., Gross, S., Paassen, B., Pinkwart, N., Hammer, B.: Domain-Independent Proximity Measures in Intelligent Tutoring Systems. In: Proceedings of the 6th International Conference on Educational Data Mining (EDM), pp. 334–335 (2013)
6. Rivers, K., Koedinger, K.R.: A Canonicalizing Model for Building Programming Tutors. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 591–593. Springer, Heidelberg (2012)
7. Vanlehn, K.: The behavior of tutoring systems. International Journal of Artificial Intelligence in Education 16(3), 227–265 (2006)

How to Select an Example? A Comparison of Selection Strategies in Example-Based Learning

Sebastian Gross[1], Bassam Mokbel[2], Barbara Hammer[2], and Niels Pinkwart[1]

[1] Humboldt-Universität zu Berlin, Germany
{sebastian.gross,niels.pinkwart}@hu-berlin.de
[2] Bielefeld University, Germany
{bmokbel,bhammer}@techfak.uni-bielefeld.de

Abstract. In this paper, we investigate an Intelligent Tutoring System (ITS) for Java programming that implements an example-based learning approach. The approach does not require an explicit formalization of the domain knowledge but automatically identifies appropriate examples from a data set consisting of learners' solution attempts and sample solution steps created by experts. In a field experiment conducted in an introductory course for Java programming, we examined four example selection strategies for selecting appropriate examples for feedback provision and analyzed how learners' solution attempts changed depending on the selection strategy. The results indicate that solutions created by experts are more beneficial to support learning than solution attempts of other learners, and that examples modeling steps of problem solving are more appropriate for very beginners than complete sample solutions.

Keywords: intelligent tutoring system, example-based learning, programming.

1 Introduction

Intelligent Tutoring Systems typically rely on an explicit formalization of knowledge about the domain being taught. For example, constraint-based tutors and model-tracing tutors are two prominent approaches that use an explicit representation of the underlying domain knowledge [10]. These techniques are not applicable if such a formalization of knowledge is not or only hardly possible. To deal with this issue, a domain model could be approximated using data sets consisting of correct (and also erroneous) problem solving examples: especially larger sets of solutions (regardless whether from experts or learners) can be expected to implicitly reflect the underlying knowledge about the problem at hand, as many structural and semantic similarities of solutions will be shared between elements in the set. Based on this implicit model of knowledge, feedback can for instance be provided as self-explanation prompts, asking a learner to compare her solution to a similar (but not identical) example contained in the data set.

In this paper, we present an example-based learning approach implemented in an ITS for Java programming that makes use of a data-driven implicit domain model. We propose and compare several strategies for selecting suitable

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 340–347, 2014.

examples. We conduct a field experiment which supports the suitability of the approach.

The paper is organized as follows. Section 2 reviews existing data-driven methods for supporting learners, in particular approaches that use examples. In Section 3, we present an implementation of an ITS for Java programming using an example-based feedback provision approach, and we discuss methods for selecting examples for feedback provision. In Section 4, we present the evaluation of the approach and discuss the results in Section 5. Finally, we conclude and give an outlook of future work in Section 6.

2 State-of-Art

Several approaches that support learning, particularly for domains that lack a strong domain theory [9], are data-driven in the regard that data sets (organized in models or databases) are used to adapt support to learners' needs, to provide feedback and to instruct learners. In [13], models of discussion posts were used to provide feedback to learners. Student solutions were analyzed and compared to the data set using keyword extraction. Also dialogue-based tutors often rely on a model learned from text corpora in order to automatically adapt dialogue responses to learners' questions and explanations. Here, Dzikovska and colleagues [6] proposed a new approach for grading student answers in a tutorial dialogue setting based on an annotated corpus.

Another educational technology method that is heavily data-driven is example-based learning. Examples are typically used to help learners in their acquisition of problem-solving skills in the way that a learner is instructed how to solve a problem using one or more examples, and after that a learner tries to solve a similar problem on her own. Example-based learning has shown to be effective in supporting learning also in ill-defined domains [4, 14]. For example, in the NavEx tutor, annotated program code examples were provided to students in order to give explanations to learners instead of providing bare solutions [3]. Also the Cognitive Tutor Authoring Tools (CTAT) were extended to support example-tracing tutors [1]. This paradigm allows to create tutors that evaluate student behavior by comparing it to generalized examples of problem solving behavior. A further interesting aspect of example-based learning is that not only correct but also erroneous examples can be used to foster learning under certain conditions [8].

The approaches mentioned above mainly use static data sets of examples that need to be created manually by experts. Also, methods like example-tracing tutors model solution processes at a very fine granularity that, while effective for feedback provision, might not be feasible for all domains [9]. In the approach presented in this paper, we assume that a data set consisting of a mix of expert solutions and (possibly erroneous) student solution attempts is available – we do not assume that a fine granular model of correct solution process exists. We describe and compare four data-driven methods for selecting suitable examples for feedback provision under these circumstances.

3 Example-Based Learning Approach

3.1 System Description

As an ITS user interface, we developed a web-based programming environment which enables students to write, compile and execute Java programs. The editor supports code-highlighting. For code debugging, compiler and program output are displayed, so that users have access to the error messages generated by the Java compiler. Technically, this user interface interacts with an ITS middleware system [7] that provides services for data access, proximity measurement and feedback provision via web service and web socket connections. Using a module for feedback provision implemented in this middleware, learners are able to request feedback on their programs. A newly submitted learner solution attempt is analyzed and compared to a data set consisting of sample programs created by experts and programs of other learners. For analyzing and comparing Java programs, a custom parser was implemented in the middleware that uses the official Java Compiler API provided by Oracle. This parser first transforms each newly submitted solution attempt of a learner into its corresponding Abstract Syntax Tree (AST) representation and then calculates semantic relations between elements in the tree. The result of the parsing process is thus a graph that consists of nodes representing syntactic elements, and edges representing hierarchic dependencies between nodes. Nodes and edges can have additional meta information (e.g., code position). In order to calculate the pairwise proximities between Java programs, we used the normalized compression distance (NCD), a proximity measure for strings [5]. Each graph was transformed to a concatenation of strings using depth first search of the underlying tree where each string represents a node or an edge with corresponding meta information. These strings were then compared using the NCD. Based on these comparisons, an appropriate example is then identified, and the feedback module generates and provides feedback, asking learners to think about differences in the programs.

3.2 Example Based Selection Strategies

As argued above, our goal is to identify a suitable counterpart to a newly submitted solution attempt within the data set. This counterpart can then be used to provide feedback to a learner by supporting her in finding mistakes in her own solution. This approach employs example-based learning principles. It requires a learner to understand the counterpart, and to identify differences between her solution attempt and the example in order to find mistakes. Given a data set consisting of sample solutions created by experts and solution attempts from other learners, we need to select a counterpart that is appropriate (in terms of its correctness and its similarity to the learner's solution attempt), and balances the (extraneous) cognitive load [12] depending on the learner's level of knowledge and stage in problem solving. A complete sample solution might be appropriate in terms of correctness but might overload a learner related to her cognitive capacity. On the other hand, a solution attempt of another learner might be

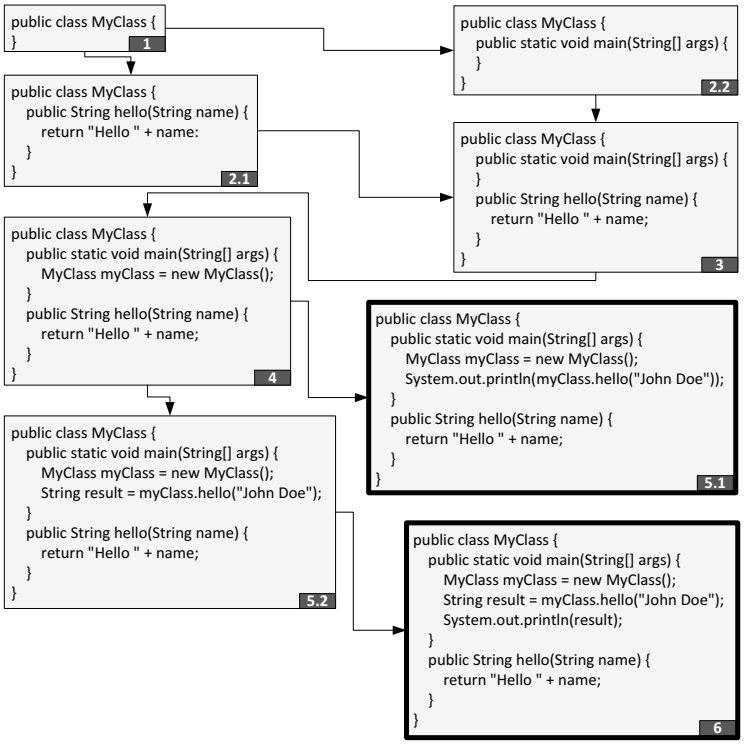


Fig. 1. Model of a sample solution composed of consecutive solution steps (and variations). Final solution steps (representing complete sample solutions) are emphasized.

inappropriate (due to mistakes made by the other learner) but could, due to a high similarity to the learner's solution, add little extraneous cognitive load.

In cognitive tutors based on the ACT-R theory [2], feedback is provided to a learner depending on her actions and the expected actions defined in a cognitive model. One idea for adapting this principle of model-tracing to a data-driven scenario is to use not only sample solutions which completely solve a problem, but to select counterparts depending on the learner's progress. We therefore propose to generate reasonable problem solving steps from complete sample solutions, and to use these steps for feedback provision – thus reducing the cognitive load in feedback provision due to simpler examples. Similar to example-tracing tutors, these sets of partial solutions can then be used to model learning paths towards a complete solution, taking on the role of a (still very high level) domain model with the following differences as compared to explicit models: it allows for multiple solution parts and it does not rely on explicit learner actions. Figure 1 illustrates how a model of steps towards a sample solution in the domain of Java programming could be designed.

Considering these aspects, we implemented four selection strategies: (1) the nearest learner solution (**NLS**) is the most similar solution that has been submitted by

another learner, (2) the nearest sample solution (**NSS**) is the most similar sample solution (part) created by an expert, (3) the next complete sample solution (**NCSS**) is the most similar sample solution created by an expert that completely solves the given problem, and (4) the next step of the nearest sample solution (**NSNSS**) is the next solution step of the most similar sample solution part contained in the data set. For example, given a Java Program as follows, strategy **NSS** would select solution step 3, **NCSS** would select step 5.1 or 6, and **NSNSS** would select step 4 (see Figure 1).

```
public class MyClass {
    public static void main(String[] args) {
    }
    public String hello(String name) {
        return Hello name
    }
}
```

Even if using peer learner solutions as examples, strategy (**NLS**) could also be beneficial for learning: these examples might be more intuitive to comprehend for learners and they might provide another perspective on how to solve a given problem. We hypothesize that feedback based on expert-created solutions would be superior since the correctness of the example can be guaranteed. In addition, we hypothesize that selecting the nearest sample solution (**NSS**) or its consecutive step (**NSNSS**) is more effective than selecting the nearest complete sample solution (**NCSS**) for the reason that cognitive load is reduced. We do not expect great differences between **NSS** and **NSNSS** since (depending on the underlying model as illustrated in Figure 1) examples selected by strategy **NSS** and **NSNSS** only differ in details – while **NSS** might be better if a learner has progressed from a correct solution step in a wrong way, **NSNSS** is likely superior if a learner has progressed from a correct solution step in a correct way but the learner is stuck.

4 Evaluation Design

To compare the effectiveness of the four example selection strategies in a realistic scenario, we conducted a field study in an introductory course for Java programming at Humboldt-Universität zu Berlin. We designed a curriculum composed of 17 tasks that involve simple problems such as how to define a class, and more complex problems such as how to repeatedly execute code with loops. For each task, two experienced Java programmers designed one or more (depending on the complexity of the task and possible alternative solution variants) sample solutions. After that, based on these complete sample solutions, we modeled correct solution steps. All these expert solutions were included in the data set.

Over a period of four weeks, students were able to use the ITS (see Section 3.1) and could request feedback to their solution attempts (which were also included in the data set and used for strategy **NLS**). The number of feedback requests (and also the time between two requests) was not limited. After each feedback provision, a learner was asked to rate the helpfulness of the provided feedback on a 3-point scale from 0 to 1 (0 = not helpful, 0.5 = fair, 1 = helpful).

The system was configured to randomly select each of the four selection strategies with a probability of 25%. Within a task, an initially chosen strategy for a specific learner was then used for each feedback provision. Since selection strategy **NLS** requires at least one program of another learner in the data set, this strategy was applied only when such programs were available within a data set for a specific task. Students were not informed about the different strategies.

5 Results

During the study, 22 students used the system, 16 of these requested feedback at least once. Table 1 summarizes the number of feedback requests and the average student-assigned scores of the ratings depending on the strategy. While the system chose each strategy equally often, the strategy heavily influenced the number of feedback requests. Selecting a complete sample solution (**NCSS**) resulted in fewer feedback requests. This can be explained by the fact that this strategy reveals the complete problem solving to learners (so that, even if they were overwhelmed with this information and did not understand the example, they could still solve the problem with copy and paste).

Learners' ratings tend to confirm our hypothesis that (at least from the students' point of view), examples created by experts are more beneficial to support learning than using solution attempts of other learners as examples. However, due to the small number of ratings, we did not conduct further statistical tests. Instead, we analyzed students' solution attempts qualitatively by determining the correctness and completeness of each step of the student's problem solving. We therefore asked an experienced Java programmer to determine whether a student's program is (i) syntactically correct, and (ii) semantically correct in terms of the problem that should be solved. Based on this assessment, the human expert should classify whether a program changed *qualitatively* between two solution attempts considering syntactic and semantic changes. We defined three conditions for classification as follows. A program has been **improved** if previous mistakes were fixed (even if further extensions made to the program contained new mistakes), or if it was correctly extended (even if previous mistakes were not corrected). A program has been remained **unchanged** if it was not extended and if previous mistakes were not fixed or previous mistakes were replaced by a new mistakes. A program has been **worsened** if it was extended

Table 1. Average score of the ratings, standard deviation and median

Strategy	Requests	Ratings	Average score	Standard deviation	Median
NLS	29	6	0.583	0.492	0.75
NSS	24	5	0.7	0.447	1
NCSS	13	3	0.833	0.289	1
NSNSS	29	6	0.917	0.204	1
	95	20	0.75	0.37	1

Table 2. Changes between student's solution attempts

Strategy	improved	unchanged	worsened	total
NLS	8	9	2	19
NSS	13	4	1	18
NCSS	8	1	0	9
NSNSS	8	8	1	17
	38	21	4	63

by new mistakes, and previous mistakes were not fixed. Table 2 summarizes the changes between solution steps depending on the example selection strategy.

It is observable that, while **NSNSS** was rated slightly more positive than **NSS** by students, the latter was more effective in terms of leading to solution improvements. Since the students who used the system in our study were very beginners, many probably rather needed a correct example that is very similar to their solution attempt in order to fix mistakes than instructions on to proceed in problem solving (even if they liked the latter). As predictable, **NCSS** immediately lead to improvements (but maybe not to learning), while **NLS** was comparable to **NSNSS** in this measure.

6 Conclusion and Outlook

In this paper, we compared four strategies for selecting an appropriate example from a data set consisting of learners' solution attempts and sample solution parts created by experts. The selected examples were used to provide feedback to learners. While the frequency of system use was too low to allow for strong claims and statistical evidence, our results support the hypotheses that using a data set consisting of expert solution steps is superior to using sample solutions only and to using learner solution attempts only. Apparently, the appropriateness of the strategies **NSS** and **NSNSS** is similar but might depend on the learner's situation. Hence, a goal of future work will be to analyze learner's needs, and to adapt feedback provision applying strategies **NSS** or **NSNSS** depending on her progress. Also, we will further investigate strategy **NLS**. While it was outperformed in this study, the data set of student solutions was relatively small. If a larger data set of student solutions is used for this strategy, the increased similarity of examples might well increase the utility of this approach – which would be appealing, since this would eliminate the need for expert sample solutions.

A further aspect of our future research will be to automatically derive solution steps from complete sample solutions in order to reduce the effort for modeling examples. While the representation of Java programs as graph structures is suitable for identifying sub-structures that can be used as simplified examples (which represent solution steps), our proximity measure is not able to identify and compare sub-structures but only whole examples. To deal with this issue, we are currently refining the proximity measure (for more details see [11]).

Acknowledgement. This work was supported by the German Research Foundation (DFG) under the grant "FIT - Learning Feedback in Intelligent Tutoring Systems." (PI 767/6 and HA 2719/6).

References

[1] Aleven, V., Mclaren, B.M., Sewall, J., Koedinger, K.R.: A new paradigm for intelligent tutoring systems: example-tracing tutors. International Journal of Artificial Intelligence in Education, 105–154 (2009)
[2] Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive tutors: Lessons learned. Journal of the Learning Sciences 4(2), 167–207 (1995)
[3] Brusilovsky, P., Yudelson, M.: From webex to navex: Interactive access to annotated program examples. Proceedings of the IEEE 96(6), 990–999 (2008)
[4] Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., Glaser, R.: Self-explanations: How students study and use examples in learning to solve problems. Cognitive Science 13(2), 145–182 (1989)
[5] Cilibrasi, R., Vitanyi, P.: Clustering by compression. IEEE Transactions on Information Theory 51(4), 1523–1545 (2005)
[6] Dzikovska, M.O., Nielsen, R.D., Brew, C.: Towards effective tutorial feedback for explanation questions: A dataset and baselines. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2012, pp. 200–210. Association for Computational Linguistics, Stroudsburg (2012)
[7] Gross, S., Mokbel, B., Hammer, B., Pinkwart, N.: Towards a domain-independent its middleware architecture. In: Chen, N.-S., Huang, R., Kinshuk,, Li, Y., Sampson, D.G. (eds.) Proceedings of the 13th IEEE International Conference on Advanced Learning Technologies (ICALT), pp. 408–409 (2013)
[8] Große, C.S., Renkl, A.: Finding and fixing errors in worked examples: Can this foster learning outcomes? Learning and Instruction 17(6), 612–634 (2007)
[9] Lynch, C., Ashley, K.D., Pinkwart, N., Aleven, V.: Concepts, structures, and goals: Redefining ill-definedness. Int. J. of Artif. Intell. Ed. 19(3), 253–266 (2009)
[10] Mitrovic, A., Koedinger, K.R., Martin, B.: A comparative analysis of cognitive tutoring and constraint-based modeling. In: Brusilovsky, P., Corbett, A., de Rosis, F. (eds.) UM 2003. LNCS, vol. 2702, pp. 313–322. Springer, Heidelberg (2003)
[11] Mokbel, B., Gross, S., Paassen, B., Pinkwart, N., Hammer, B.: Domain-independent proximity measures in intelligent tutoring systems. In: D'Mello, S.K., Calvo, R.A., Olney, A. (eds.) Proceedings of the 6th International Conference on Educational Data Mining (EDM), pp. 334–335 (2013)
[12] Sweller, J., Ayres, P., Kalyuga, S.: Cognitive Load Theory. Explorations in the Learning Sciences, Instructional Systems and Performance Technologies. Springer (2011)
[13] Walker, E., Ogan, A., Aleven, V., Jones, C.: Two approaches for providing adaptive support for discussion in an ill-defined domain. In: Proceedings of a Workshop at ITS 2008, Montreal, Canada, June 23, pp. 1–12 (2008)
[14] Wittwer, J., Renkl, A.: How effective are instructional explanations in example-based learning? a meta-analytic review. Educational Psychology Review 22(4), 393–409 (2010)

Students' Adaptation and Transfer of Strategies across Levels of Scaffolding in an Exploratory Environment

Ido Roll, Nikki Yee, and Adriana Briseno

Centre for Teaching, Learning, and Technology, the University of British Columbia
214-1961 East Mall, Vancouver, BC, V6T 1Z1, Canada
{ido.roll,adriana.briseno}@ubc.ca, nikki.yee@alumni.ubc.ca

Abstract. While the effect of scaffolding on learning has received much attention, less is known about its effect on students' strategy use, especially in transfer activities. This study focuses on students' adaptive behaviours as a function of given scaffolding and when transitioning from a scaffolded to an unstructured activity. We study this in the context of a complex physics simulation in which students choose between 124 different actions. We evaluate (i) how the scaffolding affects students' building and testing behaviours, (ii) whether these behaviours transfer to an unstructured activity, and (iii) the relationship between the adapted behaviours and learning. A repeated-measures MANOVA suggests that students adapt their learning behaviours according to the demands and affordances of the task and the environment, and that these strategies transfer from a scaffolded to an unstructured activity. No significant relationships were found between these patterns and learning.

Keywords: scaffolding, inquiry learning, microworlds, interactive simulations, transfer, self-regulated learning.

1 Introduction

Inquiry learning lets students be the scientists and thus supports learning of important scientific skills such as collaboration and self-regulated learning [1]. Within the Science Education community, the focus of the discussion seems to have shifted from asking whether inquiry learning is effective, to asking about the timing and types of scaffolding that are most effective within an inquiry framework [1-3].

Research on scaffolding within scientific inquiry environments has largely focused on the effect of scaffolding on learning outcomes. Relatively few studies evaluate the effect of scaffolding on students' use of strategies within inquiry activities [2-6]. To better understand the manner in which scaffolding supports acquisition of inquiry strategies and attitudes, one should evaluate students' learning behaviours in a transfer activity, once scaffolding has been removed [7]. However, so far there is only limited evidence that strategies that are acquired within a supported Intelligent Tutoring System transfer to future, unsupported, learning situations [6-8].

In the present study we investigate the effect of scaffolding on strategy use by evaluating how students adapt their behaviours to given scaffolding. We further ask

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 348–353, 2014.

whether the adapted behaviours transfer to new activities once the scaffolding has been removed. Last, we evaluate whether students adapt their behaviours to better match demands and affordances of the activity.

We study students' strategy use in a microworld environment. Microworlds are computational simulations where students can explore key concepts and ideas [2,5]. We use the D/C Circuit Construction Kit, a popular environment within the PhET family of 120 STEM simulations, developed by the University of Colorado Boulder (phet.colorado.edu) [9]. This simulation allows students to explore D/C circuits using wires, batteries, light bulbs, switches and other components on a virtual test bed (see Figure 1). Students can also use instruments to measure properties of the circuits such as voltage and current. PhET simulations offer implicit scaffolding through their constraints and affordances [10]. One important feature of the simulation is its visualizations. For example, changing the current in a working circuit affects the light intensity of the attached light bulbs. By showing students the consequences of their actions within the environment, the simulation provides grounded feedback to learners on their explorations [11]. The simulation also illustrates the speed of electrons, and batteries can catch on (virtual) fire. By combining grounded with explicit feedback (using the measurement instruments), PhET simulations provide a differentiated learning experience for students who may be at various stages in their understanding.

Students in the D/C simulation can choose from 124 types of actions at each moment. To make sense of the range of behaviours, we grouped these to four broad bins:

- *Building:* actions that are associated with construction of circuits, as long as no visible feedback is given. This includes adding light bulbs to circuits without batteries, creating junctions in open circuits, moving components, etc.
- *Grounded feedback*: actions that are associated with construction and have visible feedback. This includes changing resistance of resistors or voltage of batteries in working circuits, adding loops to working circuits, etc.
- *Testing:* actions that are associated with explicit testing, such as adding a voltmeter or connecting an ammeter.
- *Pauses:* non-actions that last more than 15 seconds but less than 5 minutes.

Other actions, such as those that relate to interface (e.g., enabling different views), were removed from the analysis because they do not involve the domain.

PhET Simulations are platforms for instructors to create their own activities. Thus, in addition to the scaffolding that is embedded in the simulation, scaffolding is often provided through written assignments external to the microworld environment. We previously found that similar paper-based scaffolding affects students' attitudes towards the activity, and that the adjusted attitudes transfer to a second

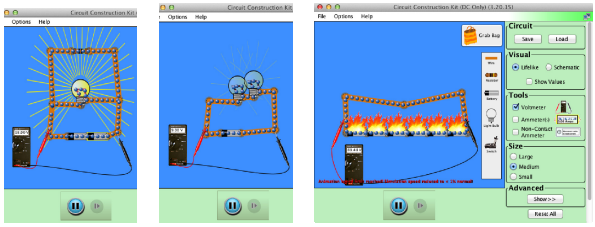


Fig. 1. The D/C Circuit Construction Kit simulation

activity once scaffolding is removed [12]. In the current work we use students' log-files to evaluate the effect of the scaffolding on students' behaviours across activities and levels of scaffolding.

2 Methods

Ninety-seven post-secondary students from introductory physics courses at a large Canadian university participated in the study. Following a pre-test, all students had 25 minutes to complete an activity about light bulbs in D/C circuits, in which they were asked to use the simulation to explore how voltage, current, and brightness of light bulbs depend on their number and arrangement. Students were randomly assigned to either a Scaffolded or an Unstructured condition. Those in the Scaffolded condition were given a worksheet that included diagrams, tables, and prompts that asked students to construct, measure, and contrast specific circuit configurations. The design of this worksheet was based on recommended activities by the designers of the simulation, in consultation with the course instructor (see Figure 3a). Students in the Unstructured condition received only the learning goal.

After a short break and a mid-survey, students were given 25 minutes to complete a second activity using the same simulation. This activity asked to investigate what happens to the current and voltage when resistors with different resistances are combined. All participants were given the same Unstructured version of this activity. Finally, students were given a post-test of learning outcomes and attitudes. All test items were conceptual and did not require calculations (see Figure 2b). The test was found to be a reliable measure of student learning, with Cronbach α = 0.75 [12]. Participants did not use the simulation during the tests.

Notably, the grounded (visual) feedback that is given by the environment can help to learn about light bulbs in Activity 1, as their light intensity is responsive to the circuit configuration. However, resistors do not provide the same kind of visual feedback. Thus, making sense of Activity 2 requires the use of measurement instruments.

We evaluate the effect of scaffolding on behaviours using a Repeated Measures MANOVA. Evaluation of productivity was done by correlating behaviours with post-test scores, controlling for pre-test. We corrected for multiple comparisons by applying Bonferroni Correction, thus using an effective alpha of 0.05 / 4 = 0.0125. While we report absolute number

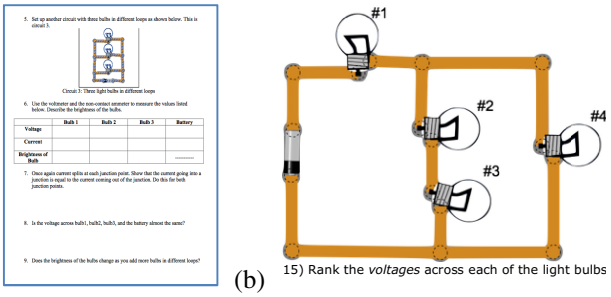


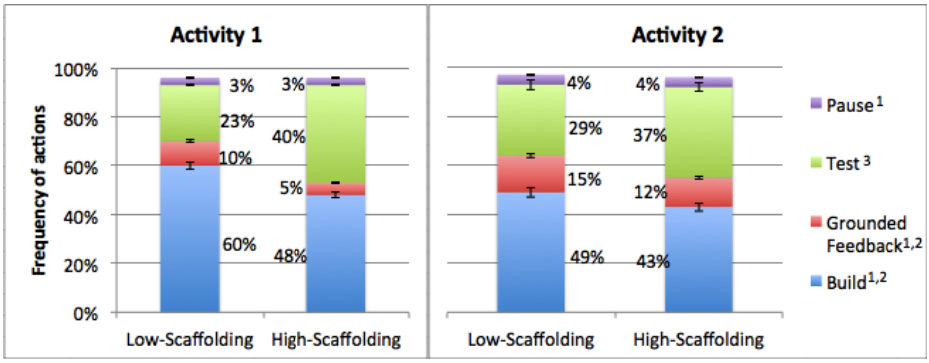
Fig 2. (a) An example from the worksheet of the Scaffolded condition. (b) An example of a post-test item

of actions, most of the analysis uses frequencies (out of 100% per student) to control for inter-student variability in number of actions.

3 Results

There was no effect for condition on the overall number of actions in either activity: Mean (SD) for Activity 1, Scaffolded: 545 (181); Unstructured: 531 (143); Activity 2, Scaffolded: 412 (152); Unstructured: 402 (163).

A repeated measures MANOVA with frequencies of Build, Grounded Feedback, Test, and Pause as outcome variables, Condition as a between-subject factor, and Activity (1 or 2) and Condition-Activity interaction as within-subject factors, was used to evaluate the effect of activity on strategy use (see Figure 3). A main effect for Condition would suggest that the given scaffolding on Activity 1 has an effect across both activities, that is, that the given scaffolding has an effect on students' strategies in Activity 1, where scaffolding differs between conditions, and in Activity 2, where there is no difference in scaffolding. A Condition-Activity interaction would suggest that the effect of the Condition on students' behaviours depends on the activity.



[1]Significant main effect for Activity; [2]Significant main effect for Condition; [3]Significant Condition-Activity interaction

Fig 3. Frequency of actions by condition and activity

Significant differences between activities were found for Build: $F(4,91) = 37$, $p < 0.0005$, $\eta^2 = 0.28$; Grounded Feedback: $F(4,91) = 87$, $p < 0.0005$, $\eta^2 = 0.48$; and Pause: $F(4,91) = 10$, $p = 0.002$, $\eta^2 = 0.10$. Significant differences between conditions were also found for Build: $F(4,91) = 22$, $p < 0.0005$, $\eta^2 = 0.19$; Grounded Feedback: $F(4,91) = 26$, $p < 0.0005$, $\eta^2 = 0.22$; and Test: $F(4,91) = 34$, $p < 0.0005$, $\eta^2 = 0.27$. The only significant Condition-Activity interaction was for Test, $F(4,91) = 12$, $p = 0.001$, $\eta^2 = 0.115$. Planned contrasts showed that Scaffolded students used more Testing than Unstructured students in either activity. Activity 1: $t(94) = 8.2$, $p < 0.0005$; Activity 2: $t(94) = 2.9$, $p = 0.005$. We also compared the rate of Test for each group, comparing Activity 1 and 2 within condition. The Unstructured students *increased* their use of Test significantly form Activity 1 to Activity 2: $t(47) = 2.9$, $p = 0.005$. Scaffolded students did not change their use of Test from Activity 1 to Activity 2: $t(47) = 1.9$, $p = 0.06$.

There were no significant differences between conditions with regard to learning gains. Mean (SD) for pre-test, Activity 1 post-test, and Activity 2 post-test were: Low-Scaffolding: 47% (18%), 62% (22%), and 56% (19%). Scaffolded: 47% (17%), 62% (19%), and 59% (19%). A higher frequency of Pauses in Activity 2 correlated with better performance on post-test (controlling for pre-test). However, this correlation did not reach significance under the adjusted alpha level: *partial-r* = 0.23, p = 0.025. No other correlations between the abovementioned behaviours and post-test performance were significant.

4 Discussion and Summary

The results presented above show three clear findings. First, there were differences between conditions on three of the four behaviours (Build, Grounded Feedback, and Test). These differences were found in both activities, even once scaffolding was removed. Second, Testing behaviours showed an interaction where students in the Unstructured condition increased their testing from Activity 1 to Activity 2, while students in the Scaffolded condition did not alter their testing behaviour. Last, the only behaviour that correlated with learning to some degree was Pauses. However, condition did not play a role in this behaviour.

With regard to our first research question, and as expected, students' behaviours changed significantly based on the scaffolding provided. Students in the Scaffolded condition performed more explicit tests throughout the activities, as was expected given the nature of their task.

Our second question asks whether the effect holds once scaffolding is removed. The results clearly show that this is indeed the case, and students transferred their scaffolded behaviours on all three categories form Activity 1 to Activity 2. In such a short intervention, we speculate that the scaffolding primed and triggered existing mindsets, rather than helped students acquire new skills. This is supported by our previous findings about the transfer of attitudes between Activity 1 and 2 [12].

Our third question asks whether students shift towards more appropriate strategies. Notably, students did not shift towards behaviours that are associated with learning. It is very likely that learning in such complex environment cannot be captured by a simple count of actions. Thus, more than telling us about students' adaptive behaviours, the lack of correlation with learning suggests that our process measures may be too rough to capture learning. A more intensive data-mining approach may be able to identify the nature of the relationship between student behaviours and learning [13]. At the same time, the pattern of the results strongly suggests that students indeed adapted their behaviours towards more appropriate behaviours, as suggested by the affordances of the available scaffolding and the activity. In addition to students in the Scaffolded condition responding to the requirements of their task, students in the Unstructured group increased their use of explicit testing when grounded feedback was no longer useful in Activity 2.

Overall, these results clearly show that students adapt their behaviours to match the affordances of the given scaffolding and activity. These results also show that

students transfer their adapted behaviours once scaffolding is removed. Thus, the study sheds some light on the manner in which prior experiences shape subsequent interactions and leads to skill acquisition at the inquiry level. It is well known that students' prior knowledge and attitudes play important roles in learning. The current study further shows how prior *experiences* change the way students engage in learning activities in terms of the strategies that they choose to apply.

Acknowledgements. This work is supported by the Social Sciences and Humanities Research Council of Canada (SSHRC) grant #430-2012-0521 and by the Betty and Gordon Moore Foundation. We would like to thank Kathy Perkins, John Blanco, and the PhET project team for their assistance.

References

1. Hmelo-Silver, C.E., Golan Duncan, R., Chinn, C.A.: Scaffolding and Achievement in Problem-Based and Inquiry Learning: A Response to Kirschner, Sweller, and Clark. Educ. Psych. 42(2), 99–107 (2007)
2. Mulder, Y.G., Lazonder, A.W., de Jong, T.: Finding Out How They Find It Out: An Empirical Analysis of Inquiry Learners' Need for Support. Int'l J. of Sci. Ed., 1–21 (2009)
3. Holmes, N.G., Day, J., Park, A.H.K., Bonn, D.A., Roll, I.: Making the failure more productive: Scaffolding the invention process to improve inquiry behaviours and outcomes in productive failure activities. Instructional Science (2013), doi:10.1007/s11251-013-9300-7
4. Sao Pedro, M.A., Baker, R.S.J.d., Gobert, J.D., Montalvo, O., Nakama, A.: Leveraging Machine-learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skill. User Modeling and User-Adapted Interaction 23, 1–39 (2011)
5. Gobert, J., Raziuddin, J., Koedinger, K.R.: Auto-scoring Discovery and Confirmation Bias in Interpreting Data during Science Inquiry in a Microworld. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 770–773. Springer, Heidelberg (2013)
6. Jeong, H., Biswas, G.: Mining Student Behavior Models in Learning-by-Teaching Environments. In: de Baker, R.S.J., Barnes, T., Beck, J.E. (eds.) Proceeds of the First International Conference on Educational Data Mining, Montreal, Quebec (2008)
7. Koedinger, K.R., Aleven, V., Roll, I., Baker, R.: In vivo experiments on whether supporting metacognition in intelligent tutoring systems yields robust learning. In: Handbook of Metacognition in Education, pp. 897–964 (2009)
8. Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R.: Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. Learning and Instruction 21, 267–280 (2011)
9. Wieman, C.E., Adams, W.K., Perkins, K.K.: PhET: Simulations that enhance learning. Science 322(5902), 682–683 (2008)
10. Podolefsky, N.S., Perkins, K.K., Adams, W.K.: Factors promoting engaged exploration with computer simulations. Phys. Rev. Special Topics - Phys. Ed. Res. 6(2) (2010)
11. Nathan, M.J.: Knowledge and situational feedback in a learning environment for algebra story problem solving. Interactive Learning Environments 5(1), 135–159 (1998)
12. Roll, I., Briseno, A., Yee, N.: Not a magic bullet: The effect of scaffolding on knowledge and attitudes in online simulations. In: Proceedings of ICLS (2014)
13. Kardan, S., Roll, I., Conati, C.: The usefulness of log based clustering in a complex simulation environment. In: Trausan-Matu, S., Boyer, K.E., Crosby, M., Panourgia, K. (eds.) ITS 2014. LNCS, vol. 8474, pp. 168–177. Springer, Heidelberg (2014)

Exploring the Assistance Dilemma: Comparing Instructional Support in Examples and Problems

Bruce M. McLaren[1], Tamara van Gog[2], Craig Ganoe[1], David Yaron[1], and Michael Karabinos[1]

[1] Carnegie Mellon University, Pittsburgh, PA, USA
{bmclaren,ganoe}@cs.cmu.edu, yaron@cmu.edu, mk7@andrew.cmu.edu
[2] Erasmus University Rotterdam, The Netherlands
vangog@fsw.eur.nl

Abstract. An important question for teachers and developers of instructional software is how much guidance or assistance should be provided to help students learn. This question has been framed within the field of educational technology as the 'assistance dilemma' and has been the subject of a variety of studies. In the study reported in this paper, we explore the learning benefits of four types of computer-based instructional materials, which span from highly assistive (worked examples) to no assistance (conventional problems to solve), with support levels in between these two extremes (tutored problems to solve, erroneous examples). In this never-before conducted comparison of the four instructional materials, we found that worked examples are the most efficient instructional material in terms of time and mental effort spent on the intervention problems, but we did not find that the materials differentially benefitted learners of high and low prior knowledge levels. We conjecture why this somewhat surprising result was found and propose a follow-up study to investigate this issue.

Keywords: assistance dilemma, classroom studies, empirical studies worked examples, erroneous examples, tutored problems to solve, problem solving.

1 Introduction

A major and recurring question for teachers and developers of instructional software is how much assistance they should provide in order to foster students' acquisition of problem-solving skills, i.e., the 'assistance dilemma' [1]. On the high assistance side of the continuum are worked examples, which present students with a fully worked-out problem solution to study and (possibly) explain. On the low assistance side of the continuum are conventional problems, which students try to solve themselves without any instructional guidance whatsoever. In between these two extremes are intelligently-tutored problems, which provide step-by-step feedback and hints either when an error is made or on demand, and erroneous examples, which are worked examples with errors in one or more of the problem-solving steps that students have to find and fix. It is straightforward to place these instructional materials on a continuum of assistance, but an important question is: How can the level and type of assistance best support learners with varying levels of prior knowledge?

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 354–361, 2014.

These types of instructional materials have all been investigated in various empirical studies, in different combinations, although never all together in a single study. For instance, the learning benefits of worked examples have been shown in a plethora of studies (for reviews see [2-4]), particularly for low prior knowledge (i.e., novice) students. Worked examples lessen the demands on cognitive resources, as compared to problem solving, when students are unfamiliar with a problem domain, and allow them to devote available cognitive resources to learning how problems should be solved [4]. In order to foster more active processing of worked examples, successful variations and strategies have been developed [5, 6]. For high prior knowledge learners, worked examples lose their effectiveness or may even become less effective for learning than practicing with conventional problem solving [7], because the assistance provided by the examples is redundant for high prior knowledge learners.

A variety of studies have also demonstrated the learning benefits of intelligently tutored problems [8 9]. Intelligent tutors, like worked examples, tend to benefit lower prior knowledge learners, those who one would expect require the type of support provided by the tutors, more than higher prior knowledge learners [10]. There are also indications that tackling worked examples before working with tutored problems improves learning efficiency (i.e., students learn as much, in less time), and, in some cases, learning outcomes, as compared to tutored problem solving alone [3, 11].

Recent studies – a relatively small number compared to worked examples and intelligently tutored problems – have also investigated the effects of erroneous examples [12-14]. Presenting students with errors might help eradicate those errors by prompting more reflection than would occur naturally. Erroneous examples have so far been shown to be particularly beneficial to learners with some prior knowledge [13], which makes intuitive sense, since a student who has not yet understood the basic concepts and problem-solving procedures within a domain is less likely to be able to differentiate and make sense of correct and incorrect problem solutions.

Finally, as mentioned above, giving students problems to solve, without feedback or support, has been shown to be most beneficial to more advanced students, ones with sufficient prior knowledge to gain from practice without assistance [7].

There is some variability among studies in whether or not feedback was provided to students in the conventional problems group. Paas provided students with feedback on practice problems, which consisted of worked examples. Still, studying worked examples (with a practice problem after two examples) was found to be more effective than practicing with conventional problem solving with feedback [6].

In this study, we intended to compare the learning benefits of these four types of instructional materials (developed for and deployed on the web) at different levels of expertise (lower, higher). Although such comparisons have been partially made, no studies have compared the effectiveness and efficiency of all four support strategies to each other. This study aimed to make that comparison, taking into account students' prior knowledge level in order to take a first step towards testing our hypothesis that worked examples and tutored problem solving are more suitable learning materials for students with lower prior knowledge, while erroneous examples and conventional problem solving are more suitable for students with higher prior knowledge.

2 Method

Participants and Design. Participants were 179 10th and 11th grade students from two high schools in the U.S. Twenty-four participants were excluded because they did not fully complete all phases of the study. The remaining 155 students had a mean age of 15.4 (SD = 0.59), with 75 males, 80 females. Participants were randomly assigned to one of the 4 instructional conditions: (1) Worked Examples (*WE*), (2) Erroneous Examples (*ErrEx*), (3) Tutored Problems to Solve (*TPS*), or (4) Problems to Solve (*PS*).

Materials. A web-based stoichiometry tutor used in earlier studies [3, 15] was revised to support this study. Stoichiometry is a subdomain of chemistry in which basic mathematics (i.e., multiplication of ratios) is applied to chemistry concepts.

Table 1. Conditions and Materials used in the study. *Italicized* items vary across conditions

	WE	***TPS***	***ErrEx***	***PS***
	Questionnaire	Questionnaire	Questionnaire	Questionnaire
	Pretest (A or B)	Pretest (A or B)	Pretest (A or B)	Pretest (A or B)
	WE Intro video	*PS Intro video*	*ErrEx Intro video*	*PS Intro video*
	Stoich videos (both at beginning and interspersed)	Stoich videos (both at beginning and interspersed)	Stoich videos (both at beginning and interspersed)	Stoich videos (both at beginning and interspersed)
x5 {	*WE-1*	*TPS-1*	*ErrEx-1*	*PS-1*
x5 {	*WE-2*	*TPS-2*	*ErrEx-2*	*PS-2*
x5 {	Embedded-Test-1	Embedded-Test-1	Embedded-Test-1	Embedded-Test-1
	Posttest (A or B)	Posttest (A or B)	Posttest (A or B)	Posttest (A or B)

Questionnaire. Students were asked demographic, computer use, and self-perceived prior knowledge questions.

Pretest and Posttest. The pretest and posttest consisted of four stoichiometry problems to solve (of the same form as the Intervention Problems) and four conceptual knowledge questions to answer. There was an A and B form of the test, isomorphic to one another and counter-balanced within condition (i.e., approximately ½ of the students in each condition received Test A as pretest and Test B as posttest, and vice versa). The stoichiometry problems consisted of 94 steps in total (one point per correct step). The conceptual questions consisted of 7 possible answers (one point per correct answer). This resulted in a maximum total score of 101 points.

Intro and Instructional videos. After taking the pretest, all students watched a video specific to their condition, which used a narrated example to explain how to interact with the user interface. In addition, students in all conditions were presented with the same instructional videos about stoichiometry and problem solving techniques, starting at the beginning of the intervention and spread throughout the intervention.

Intervention Problems. Students were presented with 10 intervention problems, specific to condition and grouped in isomorphic pairs, as shown in Table 1 (e.g., WE-1 and WE-2 are an isomorphic pair, TPS-1 and TPS-2, etc.). The complexity of the stoichiometry problems presented in the intervention gradually increased.

The *WE* items consisted of problem statements and screen-recorded videos (30-70 sec.) of how to solve the problem. When the video finished, students had to select the "reason" for each step from a drop-down menu. Then they click the "Done" button and feedback appeared. When they were correct, they were encouraged to study the final correct problem state; when they were incorrect a fully worked-out final solution appeared below the problem that students could study self-paced (see Figure 1).

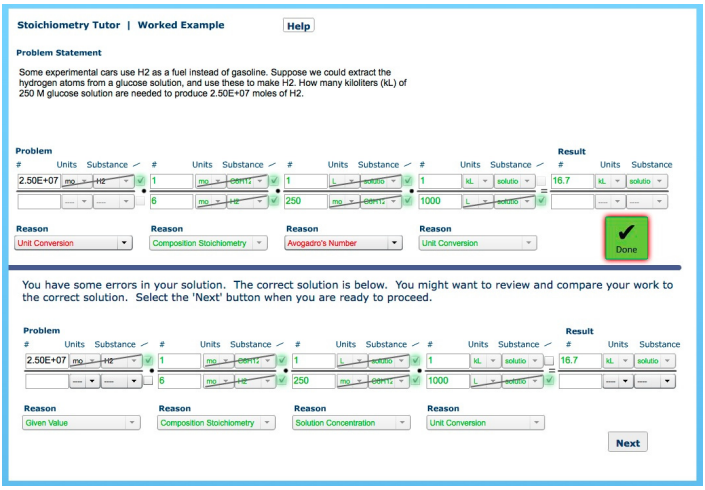


Fig. 1. WE with incorrect reasons resulting in correct worked example feedback

The *TPS* items consisted of a problem statement and fields to fill in (similar to the top of Figure 1) and students had to attempt to solve the problem assisted by on-demand hints and error feedback. There were up to 5 levels of hints per step, with the bottom-out hint giving the answer to that step. Because the tutored problems always ended in a correct final problem state due to the given hints, no feedback was given at the end but students were encouraged to study the final correct problem state.

The *ErrEx* items also consisted of screen-recorded video (30-70 sec.) demonstrating how to solve the problem, except the items contained 1 to 4 errors that students were instructed to find and fix. They had to fix at least one step before they could click 'Done', at which point the same 'correct' or 'incorrect' feedback messages as in the *WE* condition appeared, with a correct example shown if errors were still present.

The *PS* items consisted of a problem statement and fields to fill in (similar to the top of Figure 1) and students had to attempt to solve the problem themselves, without any assistance. They had to fill out at least one step before they could click the 'Done' button. When they clicked the 'Done' button, the same 'correct' or 'incorrect' feedback messages as in the *WE* condition appeared, with a correct example shown if errors were still present.

Embedded test problems. After every two intervention items, an embedded test problem was given that was identical to the first intervention item of the two (i.e., intervention problems 1, 3, 5, 7, and 9), but in *PS* form without any guidance or feedback. These problems consisted of a total of 122 steps (one point per correct step).

Mental effort rating scale. A 9-point mental effort rating scale [6] was administered after each intervention problem.

Procedure. The experiment was conducted at students' schools within their regular science classrooms. In total, the study took 6 class periods to complete. Students received a login for the web-based environment and could work at their own pace (for the phases and tasks they encountered, see Table 1). When they had finished with the intervention phase, however, they could not progress to the posttest; this test took place on the sixth and final period for all students.

3 Results

As mentioned in the introduction, we intended to compare the learning benefits of the four types of instructional materials (developed for and deployed on the web) at different levels of expertise (lower, higher). However, apart from differences in prior knowledge, these analyses did not yield additional insights about the instructional conditions compared to analysis of the overall sample. Because of page limitations, we therefore report only the overall sample results here.

Data are presented in Table 2 and were analyzed with ANOVA. There were no significant differences among conditions in pretest ($p = .783$)[1], posttest ($p = .693$), or embedded test problem performance ($p = .326$).

Table 2. Mean performance, mental effort, and time on task per condition

	***WE* (*n*=39)**	***TPS* (*n*=36)**	***ErrEx* (*n*=43)**	***PS* (*n*=37)**
Pretest (max=101)	48.6 (12.8)	49.4 (13.5)	48.8 (15.4)	46.3 (14.3)
Posttest (max=101)	68.5 (17.3)	71.1 (13.4)	68.3 (18.4)	66.4 (17.1)
Embedded test (max=122)	89.4 (23.7)	95.3 (23.3)	88.3 (27.0)	84.8 (23.1)
Mental eff. inter. probs. (1-9)	4.4 (1.8) *	6.1 (1.7)	5.8 (1.4)	6.1 (1.3)
Intervention time (mins)	19.8 (5.8) *	62.4 (17.2)	37.2 (9.6) #	52.1 (25.2) ~
Reflection time (mins)	1.7 (1.1)	1.3 (1.0)	4.3 (2.6) ^	6.5 (3.9) *

* - Significant difference to all other conditions
# - Significant difference to *TPS* and *PS* conditions
^ - Significant difference to *WE* and *TPS* conditions
~ - Significant difference to *TPS*

However, mean mental effort invested on the intervention problems differed significantly among conditions ($p < .001$, $\eta_p^2 = .166$); Bonferroni post hoc tests showed effort was lower in the *WE* condition than in all other conditions (*ErrEx*: $p < .001$, $d = 0.891$; *TPS*: $p < .001$, $d = 0.954$; *PS*: $p < .001$, $d = 1.04$).

Intervention time also differed significantly among conditions ($p < .001$, $\eta_p^2 = .503$); Bonferroni post hoc tests showed that time spent in the *WE* condition was

[1] Due to space limitations, and for readability, only *p* and effect size values are reported in this paper. F statistics and further statistical details are available from the first author.

lower than in all other conditions (*ErrEx*: $p < .001$, $d = 2.195$; *TPS*: $p < .001$, $d = 3.312$; *PS*: $p < .001$, $d = 1.762$)), in the *ErrEx* condition was lower than in the *TPS* and *PS* conditions (*TPS*: $p < .001$, $d = 1.812$; *PS*: $p < .001$, $d = 0.782$), and in the *PS* condition was lower than in the *TPS* condition ($p = .038$, $d = 0.478$). Note that the last finding makes sense, given that the *TPS* condition also received instructional assistance and feedback *during* intervention problems. Reflection time on the correct worked example given as feedback differed significantly among conditions ($p < .001$, $\eta_p^2 = .418$); Bonferroni post hoc tests showed it was lower in the *WE* and *TPS* conditions (which did not differ from each other, $p = 1.000$) than in the *ErrEx* (*WE* vs. *ErrEx*: $p < .001$, $d = 1.253$; *TPS* vs. *ErrEx*: $p < .001$, $d = 1.507$) and *PS* conditions (*WE* vs. *PS*: $p < .001$, $d = 1.670$; *TPS* vs. *PS*: $p < .001$, $d = 1.848$). Reflection time in the *PS* condition was significantly higher than in all other conditions (*WE* vs. *PS*: $p < .001$, $d = 1.670$; *ErrEx* vs. *PS*: $p < .001$, $d = 0.672$; *TPS* vs. *PS*: $p < .001$, $d = 1.848$).

4 Discussion and Conclusions

Our findings suggest that example study was more efficient in terms of the learning process: the *WE* condition attained equal test performance with less time and mental effort on the intervention problems than all other conditions. This is in line with findings from prior studies that compared studying worked examples to conventional problem solving [cf. 16], as well as to tutored problem solving [3, 11].

In contrast to other studies on the worked example effect [6, 7, 16], we did not find a learning outcome benefit for worked examples, either overall or in the lower prior knowledge sample. Also, our hypothesis that the instructional materials would be differentially beneficial to learners based on prior knowledge level was not supported. A distinguishing aspect of this study is the use of a common user interface for conditions ranging from the highly assistive (*WE*) through unassisted problem solving (*PS*). In *WE* and *ErrEx*, the examples are implemented as videos of problem solving within the interface. In PS and TPS, students use the interface to solve problems, with conditions differing with regard to immediate versus delayed feedback. This design has the advantage of allowing tight control of conditions, with differences arising only in the nature of student interaction with the interface. The finding of equal learning gains across conditions is interesting, given the substantial differences in the nature of the student interactions as well as in the mental effort and time across condition.

A common feature across conditions that may account for these findings is the presence of a fully and correctly worked example at the end of each problem-solving episode, which students could study as long as they wished. We provided students with feedback in order to make the comparison among the conditions as fair as possible; however, providing feedback in the form of fully worked-out solutions led to a very strong presence of worked examples in *every* condition. *TPS* students generate the solution, but they also effectively get worked examples by drilling down to bottom-out hints. In the *ErrEx* and *PS* conditions, in which errors occurred often (81% of the time) and a correct example was then provided, the mean time spent reflecting on comparing student work to a correctly worked example (*ErrEx* = 31.1 secs and

PS = 42.8 secs) is comparable to the amount of time students in the *WE* condition spent watching the animated worked example (i.e., between 30 and 70 seconds, as earlier mentioned). Few other studies [cf. 6] on the worked example effect provided students in the *PS* condition with worked examples as feedback, and in those studies they could review the feedback for a restricted amount of time that was less than the amount of time students in our *WE* condition could study the examples.

Because the use of worked examples may have made the conditions too similar, we will next run a study in which the conditions will be more distinct. We will drop the worked examples as a form of feedback in the *WE*, *ErrEx*, and *PS* conditions. Instead of receiving the correct worked example as feedback, students will only see feedback highlighting the steps they correctly and incorrectly complete.

Acknowledgement. National Science Foundation Award No SBE-0354420 ("Pittsburgh Science of Learning Center") funded this research.

References

1. Koedinger, K.R., Aleven, V.: Exploring the assistance dilemma in experiments with cognitive tutors. Educational Psychology Review 19, 239–264 (2007)
2. Atkinson, R.K., Derry, S.J., Renkl, A., Wortham, D.: Learning from examples: Instructional principles from the worked examples research. Review of Ed R'ch 70, 181–214 (2000)
3. McLaren, B.M., Lim, S., Koedinger, K.R.: When and how often should worked examples be given to students? New results and a summary of the current state of research. In: Love, B.C., McRae, K., Sloutsky, V.M. (eds.) Proceedings of the 30th Annual Conference of the Cog. Sci. Society, pp. 2176–2181. Cognitive Science Society, Austin (2008)
4. Sweller, J., Van Merriënboer, J.J.G., Paas, F.: Cognitive architecture and instructional design. Educational Psychology Review 10, 251–295 (1998)
5. Chi, M.T.H., Bassok, M., Lewis, M.W., Reimann, P., Glaser, R.: Self-explanations: How students study and use examples in learning to solve problems. CogSci. 13, 145–182 (1989)
6. Paas, F.: Training strategies for attaining transfer of problem-solving skill in statistics: A cognitive load approach. Journal of Educational Psychology 84, 429–434 (1992)
7. Kalyuga, S., Chandler, P., Tuovinen, J., Sweller, J.: When problem solving is superior to studying worked examples. Journal of Educational Psychology 93, 579–588 (2001)
8. VanLehn, K.: The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems. Educational Psychologist 46(4), 197–221 (2011)
9. Graesser, A.C., Chipman, P., Haynes, B.C., Olney, A.: AutoTutor: An intelligent tutoring system with mixed-initiative dialogue. IEEE Transactions in Ed. 48, 612–618 (2005)
10. Ritter, S., Anderson, J.R., Koedinger, K., Corbett, A.: Cognitive Tutor: Applied research in mathematics education. Psychonomic Bulletin and Review 14, 249–255 (2007)
11. Salden, R., Koedinger, K.R., Renkl, A., Aleven, V., McLaren, B.: Accounting for beneficial effects of worked examples in tutored problem solving. Ed. Psy. Rev. 22(4), 379–392 (2010)
12. Durkin, K., Rittle-Johnson, B.: The effectiveness of using incorrect examples to support learning about decimal magnitude. Learning and Instruction 22, 206–214 (2012)

13. Grosse, C.S., Renkl, A.: Finding and fixing errors in worked examples: Can this foster learning outcomes? Learning and Instruction 17, 612–634 (2007)
14. McLaren, B.M., et al.: To err is human, to explain and correct is divine: A study of interactive erroneous examples with middle school math students. In: Ravenscroft, A., Lindstaedt, S., Kloos, C.D., Hernández-Leo, D. (eds.) EC-TEL 2012. LNCS, vol. 7563, pp. 222–235. Springer, Heidelberg (2012)
15. McLaren, B.M., DeLeeuw, K.E., Mayer, R.E.: Polite web-based intelligent tutors: Can they improve learning in classrooms? Computers & Education 56(3), 574–584 (2011)
16. Van Gog, T., Paas, F., Van Merriënboer, J.J.G.: Effects of process-oriented worked examples on troubleshooting transfer performance. Learning and Instruction 16, 154–164 (2006)

A Systematic Approach for Providing Personalized Pedagogical Recommendations Based on Educational Data Mining

Ranilson Oscar Araujo Paiva[1], Ig Ibert Bittencourt Santa Pinto[2], Alan Pedro da Silva[2], Seiji Isotani[3], and Patricia Jaques[4]

[1] Universidade Federal de Campina Grande, COPIN - Rua Aprígio Veloso, 882 - Bodocongo CEP: 58109-900, Campina Grande/ PB - Brasil
ranilson@copin.ufcg.edu.br

[2] Universidade Federal de Alagoas, NEES, IC - Av. Lourival Melo Mota, s/n - Tabuleiro do Martins CEP: 57072-970, Maceió/ AL - Brasil
{ig.ibert, alanpredro}@ic.ufal.br

[3] Universidade de São Paulo, ICMC - Avenida Trabalhador São-Carlense, 400 - Centro CEP: 13566-590, São Carlos/ SP - Brasil
sisotani@icmc.usp.br

[4] Universidade do Vale do Rio dos Sinos, PIPCA - Av. Unisinos, 950 - Cristo Rei CEP: 93022-000, São Leopoldo/ RS - Brasil
pjaques@unisinos.br

Abstract. This work presents an approach to assist teachers, tutors and students from online learning environments. It is a four-steps process called Pedagogical Recommendation Process that uses the coordinated efforts of human actors (pedagogical and technological specialists) and artificial actors (computational artifacts). The process' objective is to find relevant information in educational data to help creating personalized recommendations. Using the process it was possible to detect issues within a learning environment (UFAL Línguas), and discovered why some students were facing difficulties, and what other students were doing in order to succeed in the course. This information was used to personalize pedagogical recommendations.

Keywords: Pedagogical Recommendation Process, Personalized Recommendations, Educational Data Mining, Online Learning Environments, Online Courses.

1 Introduction

Studying some works on the 8th grand challenge, namely, Learning for Life: "Conceptualize learning environments and understand how people will engage with learning, and what learning for life will be like" [5], revealed a trend towards a paradigm where education is available and accessible to anyone, from anywhere and at any time (Anyone, Anywhere and Anytime Learning - AAAL [2]). It relies on information and communication technology and its adoption increased the offering of computer-based online courses [3]. In these environments

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 362–367, 2014.

students interact with some educational resources (exercises, tests, videos, forums, chat, etc.), generating substantial quantities of data. Dealing with these data raised some research questions: (1) How can data from learning environments be appropriately used? (2) How can the outcomes be used to improve students' learning experience? (3) How can this process become transparent for teachers and tutors? To answer these questions this work proposes a systematic approach named Pedagogical Recommendation Process.

2 Proposal

Our proposal is a systematic approach to help teachers and tutors, from online learning environments, to assist students in their pedagogical needs. This approach is a cyclic and iterative process named Pedagogical Recommendation Process - PRP. The process is semi-automatic and composed of four steps (figure 1). Each step requires the coordinated actions of human actors (specialists in the pedagogical and technological domains) and artificial actors (computational artifacts). It uses the students interactional data as input, applying educational data mining techniques to detect and discover pedagogical difficulties in order to personalize pedagogical recommendations. Finally, the students' performance is monitored and evaluated.

Fig. 1. The Pedagogical Recommendation Process

2.1 Background

In order to understand the process, two concepts used in this work are introduced: Mining Capsules and Pedagogical Recommendations.

Mining Capsules organize and encapsulate the mining process and its parameters. The mining capsules are intended to promote reuse and automate data mining tasks associated to a pedagogical scenario. For example: students' interactions patterns can be analyzed [6], predict students' results in exercises' and tests' can be predicted [8], students can be grouped according to their interests and level of engagement to the course [1], etc. To create a capsule it is necessary to define (1) What is the mining capsule's objective? (2) What data are necessary to reach these objectives? (3) How are these data processed (choose the appropriate data mining task to reach the goal [9], and specify the details of the mining)?

Pedagogical Recommendations are reactive or preventive actions, associated with some defined pedagogical issues. These actions may use the learning environments' native educational resources, or external validated educational resources. Their objective is to improve learning experience and solve known pedagogical problems [7]. They use the learning environment's educational resources in order to offer students personalized ways to practice or improve on topics they are not performing well.

2.2 The Pedagogical Recommendation Process

Step 1 - Detect Practices. This steps objective is to define metrics and their intervals of interest, as triggers to detect occurrences in the learning environment that affect students' learning experience. The pedagogical actor is responsible for defining the metrics and their respective intervals of interest. The technological actor associates these definitions with the data available in the learning environment. The computational actor operationalizes this step, generating alerts when a pedagogical practice is detected.

Step 2 - Discover Patterns. This step's objective is to discover a possible explanation for the practices detected (hypothesis). The pedagogical actor creates one or more hypotheses to discover the reason for the practice detected, defining its acceptance/rejection criteria. The technological actor defines the data and methods to reach these criteria, setting up the mining process to discover the details. The computational actor operationalizes the step's definition by: (1) executing specific statistical analysis to accept/reject the hypothesis (2) executing the defined mining process, presenting the outcome in a way the other actors may extract relevant information that explain the practices.

Step 3 - Recommend. This step's objective is to provide personalized recommendations, for the practices detected, based on the patterns discovered. In this step the pedagogical actor creates a general version for each type of recommendation, based on the patterns discovered. For example, "Answer Quantity Difficulty Level questions about Topic", that can be personalized to "Answer 10 difficult questions about the Basic Set Operations. The technological actors develop a way to create, store and personalize these general recommendations, associating them to a particular pedagogical issue. The computational actor operationalizes this step, executing the actions programmed by the technological actor.

Step 4 - Monitor and Evaluate. This step's objective is to measure and compare the students' performance, before and after receiving the recommendation. The appropriateness of the recommendations is also monitored and evaluated. The pedagogical actor is responsible for defining the success criteria (issue is solved), and review recommendations marked as "needs reviewing". The technological actor develops a way to monitor and evaluate students' and groups' performance, and the relevance of the recommendations. The computational actor operationalizes this step, monitoring and evaluating students and recommendation items.

3 Case Study

The course UFAL Línguas (Espanhol) used an online learning environment for teaching foreign languages - UFAL Línguas. It received 2075 enrollment requests, from which 200 were accepted [UFAL Línguas, 2012a, 2012b]. The course lasted five months (October 2012 to February 2013), and was composed of six units. A teacher and 8 tutors were responsible for maintaining it. In the end of the course 37 students, with scores above 2500 points, were approved.

3.1 Applying the Pedagogical Recommendation Process

Applying - Detect Practices: A histogram was generated in order to visualize how the students' scores were distributed (figure 2). That showed us three groups: (1) Failing students with very low score. (2) Failing and approved students results might have been different if teachers could quickly react the students' problems. The third group provided us with information on what should be done to perform well.

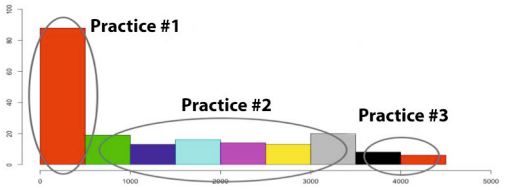


Fig. 2. The Pedagogical Recommendation Process

Applying - Discover Patterns: According to [6] some practices can be explained by discovering patterns in the way students interact with the educational resources (hypothesis: the more students interact with the educational resources, the better their performances are). The hypothesis was valid and the mining process used was transformed in a mining capsule, shown in figure 3.

Code	Data	Preprocessing	Data Mining	Postprocessing
MC1	**Group Data by Course** Accesses Number Available exercises number Answered exercises number Answered exercises number by level Available exercises number Number of tests done Interaction via forum Interactions via chat Final average in tests Final status in the course	Remove outliers (lower than **Q1 - 3 * IQR** and higher than **Q3 + 3 * IQR** Remove dropouts registries Hide ID Treat categorical data Treat non-numerical data Fill missing and null values Store results in ARFF file	**API: Weka** **Algorithm: J48** Cross-validation: 10 fold Parameters (binarySplits = false confidenceFactor = 0.5 minNumObj = 3 reducedErrorPruning = true subtreeRaising = true unpruned = false useLaplace = false)	Add IDs Convert categorical values Generate "decision tree" graphical representation

Fig. 3. Specifications for the mining capsule MC1

In the pre-processing the outliers were removed[1] , dropout students', students' ID and treated null and missing values to avoid influence in the results. For the

[1] For this work's purpose, outliers were values three times above and/or below the interquartile range - IQR.

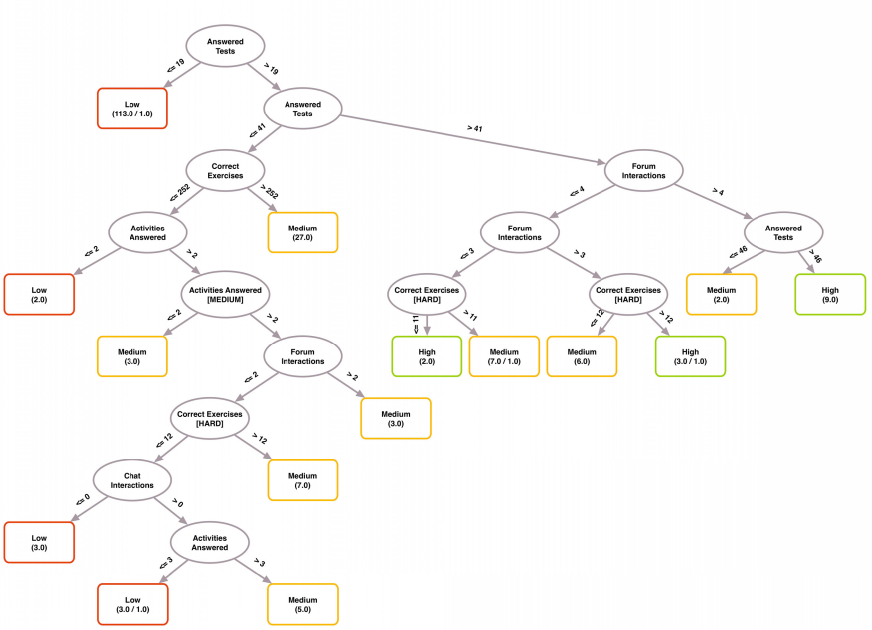


Fig. 4. Decision tree showing the interactions that resulted in low, medium and high performances

Association Rule	Recommendation Created	Teachers' Perceived Relevance															Points	Relevance
IF testsAnswered <= 19	UnansweredTestsPenalty[0%]	5	1	4	1	1	5	1	1	1	2	3	3	2	4	1	1	2	38	44,71
	UnansweredTestsPenalty[50%]	1	3	1	4	2	1	1	5	2	4	2	2	2	1	3	4	2	40	47,06
	RedoTestsPenalty[0%]	1	1	2	1	1	2	1	1	1	1	1	2	1	1	4	1	1	2	24	28,24
	RedoTestsPenalty[50%]	1	4	1	1	4	2	2	1	1	1	1	1	2	2	5	1	4	1	2	35	41,18
	NewTests	3	5	2	2	4	3	5	1	1	1	3	3	5	4	5	4	5	4	**59**	**69,41**
IF testsAnswered <= 19 AND correctExercises <= 252 AND activitiesDone <= 2	UnansweredActivitiesPenalty[0%]	5	3	1	1	1	5	1	1	3	2	3	3	2	4	1	1	2	39	45,88
	UnansweredActivitiesPenalty[50%]	1	4	2	2	2	1	1	5	4	5	2	2	3	1	4	4	2	45	52,94
	ActivityLevel[X]	1	1	1	2	1	3	1	4	1	2	4	1	1	4	2	3	1	33	38,82
	RedoActivitiesPenalty[0%]	1	2	3	2	3	1	3	1	1	1	2	1	3	4	1	1	2	32	37,65
	WatchAgainVideosUnits[X]	3	5	5	3	4	5	5	3	3	2	4	3	4	4	2	1	4	**60**	**70,59**
	ExternalContent[X]	3	4	5	1	4	4	2	1	1	3	4	2	4	3	2	2	2	47	55,29
IF testsAnswered <= 19 AND correctExercises <= 252	RedoExercisesPenalty[0%]	1	5	4	1	3	1	5	1	1	1	4	?	4	4	2	1	2	40	47,06
	WatchAgainVideosUnits[X]	3	5	3	3	4	5	2	4	3	4	3	?	4	1	3	1	4	52	61,18
	ExternalContent[X]	4	4	4	1	3	4	2	2	1	5	3	?	3	1	3	4	3	47	55,29
	ExtraExercises	5	4	4	4	4	5	4	4	3	4	4	?	5	4	4	5	4	**67**	**78,82**
	RedoExercisesLevel[EASY]	2	2	2	2	2	1	1	5	2	2	3	?	2	3	2	2	1	34	40,00
	RedoExercisesLevel[MEDIUM]	2	4	2	2	2	1	1	5	2	2	3	?	3	5	2	2	1	39	45,88
IF forumInteractions <= 3 AND chatInteractions == 0	RequireAccessChat	4	5	1	3	4	5	4	5	3	3	2	?	5	3	4	4	3	58	68,24
	ActivityChat	4	5	1	3	4	4	2	5	4	4	3	?	5	3	1	4	3	55	64,71
	ChatInteractions	5	5	5	4	3	5	4	3	3	4	3	?	5	4	2	3	4	62	72,94
	ChatActivities	3	5	5	3	5	5	3	5	5	4	4	?	5	5	4	3	5	**69**	**81,18**
	ExtraActivityChat	4	5	4	4	5	5	1	4	4	3	4	?	5	4	4	3	4	63	74,12

Fig. 5. Teachers grades regarding their perceived relevance for each recommendation

mining part J48 algorithm (C4.5 algorithm) was used to generate a decision tree [9]. The resulting classifier was 88.89% accurate. For the data post-processing, the outcomes were treated to be used in the Recommend step, allowing the pedagogical specialists to identify relevant information [4], by generating a tree-like representation of the results (figure 4).

Applying - Recommend: Five pedagogical recommendations were created for each node that leads to a low performance. These recommendations' relevance

was later evaluated by an independent group of teachers (figure 5). They were then stored for a future offering of the course.

Applying - Monitor and Evaluate: As the data available was from the end of the course, it was not possible to generate new data on recommendation usage. That is a limitation, but it does not invalidate the work.

4 Conclusions

Applying the Pedagogical Recommendation Process in a case study, we were able to answer our research questions, discovering situations where students were facing difficulties, which helped us generate personalized recommendations based on these specific problems. Although the results were encouraging, showing that we could detect problems (step 1), discover the patterns associated to them (step 2) and, consequently, recommend personalized tasks focusing on the students' needs (step 3), it is still necessary to test the "Monitor and Evaluate" step (step 4), once that we only had access to the interactional data after the end of the course. We conclude that the process can be applied to the next offerings of this course, preparing the learning environment to identify and react to known pedagogical situations. It can also be used in other courses based on the same learning environment.

References

1. Bayer, J., Budzovska, H., Geryk, J., Obsivac, T., Popelinsky, L.: Predicting drop-out from social behaviour of students. In: Educational Data Mining Conference (2012)
2. Bittencourt, I.I., de Barros, C.E., Silva, M., Soares, E.: A Computational Model for Developing Semantic Web-based Educational Systems. Knowledge-Based Systems 22, 302–315 (2009)
3. Chrysafiadi, K., Virvou, M.: Student modeling approaches: A literature review for the last decade. Expert Syst. Appl. 40(11), 4715–4729 (2013)
4. Gibert, K., Izquierdo, J., Holmes, G., Athanasiadis, I., Comas, J., Snchez-Marr, M.: On the role of pre and post-processing in environmental data mining. In: International Congress on Environmental Modeling and Software (2008)
5. Kavanagh, J., Hall, W.: Grand challenges in computing research 2008. In: Grand Challenges in Computing Research, GCCR 2008. UK Computer Research Committee, United Kingdom (2008)
6. Moran, J.M.: O que aprendi sobre avaliação em cursos semi-presenciais. In: Silva, M., Santos, E. (eds.) Avaliação da Aprendizagem em Educação Online. Loyola, São Paulo (2006), `http://www.eca.usp.br/prof/moran/aprendi.html` (accessed in: July 3, 2013)
7. Paiva, R.O.A., Bittencourt, I.I., Pacheco, H., da Silva, A.P., Jaques, P., Isotani, S.: Mineração de Dados e a Gestão Inteligente da Aprendizagem: Desafios e Direcionamentos in XXXII Congresso da Sociedade Brasileira de Computação (2012)
8. Park, J.-H., Choi, H.J.: Factors Influencing Adult Learners' Decision to Drop Out or Persist in Online Learning. Educational Technology & Society 12, 207–217 (2009)
9. Witten, I., Frank, E., Hall, M.: Data Mining: Practical Machine Learning Tools and Techniques, 3rd edn. Massachusetts (2011)

ToneWars: Connecting Language Learners and Native Speakers through Collaborative Mobile Games

Andrew Head[1], Yi Xu[2], and Jingtao Wang[1]

[1] Computer Science & LRDC, University of Pittsburgh, PA, USA
{amh140@,jingtaow@cs}pitt.edu
[2] East Asian Languages & Literatures, University of Pittsburgh, Pittsburgh, USA
xuyi@pitt.edu

Abstract. In this paper, we present ToneWars, a collaborative mobile game for learning Chinese as a Second Language (CSL). ToneWars provides a learning experience that combines mastery learning, microlearning, and group-based interaction between CSL learners and native speakers. The game explores how unique input modalities, like touch gestures and speech recognition, can improve language acquisition tasks on mobile devices. We report the design motivations and lessons learned through the iterative design process. We believe many insights from developing ToneWars are generalizable to designing productive language learning technology. Through a 24-participant (12 CSL and native speaker pairs) user study, we found ToneWars provides learning benefits for second-language learners and engages native speakers.

Keywords: Mobile Learning, Serious Games, Crowdsourcing, Collaborative Learning.

1 Introduction

Learning a second language (L2) can be an extremely rewarding pursuit [14]. However, the process is notoriously challenging. It can take learners thousands of hours [17] to achieve intermediate fluency. Additional challenges are posed when a new language varies greatly from a learner's native tongue, as is the case for English speakers learning Chinese. At the same time, the need to improve L2 learning is just as important as ever. According to ACTFL, Chinese as Second Language (CSL) enrollments in K-12 schools in the U.S have increased 195% from 2004 to 2007 [1].

Numerous challenges arise for CSL students as they acquire listening, speaking, reading, and writing skills. One of the best-known challenges involves dealing with the large quantity of characters [1]. Meanwhile, many consider the most difficult task of CSL acquisition to be correct perception and pronunciation of tones due to the interference of the learner's native language [7], [19]. Tones in Chinese determine *meaning,* whereas tones in western languages, such as English, are used for

[1] According to national standard GB2312 level 1, there are 3,755 Chinese characters defined as "frequently" used in daily communication.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 368–377, 2014.

grammatical and expressive *inflection*. Authentic, low-risk practice with native speakers can provide opportunities to explore L2 nuances [11], [13]. However, native speakers are usually inaccessible to learners.

In response to these challenges, we present ToneWars **(Fig. 1)**, a collaborative educational game to help CSL learners master the Chinese tones. ToneWars connects learners and native speakers in collaboration though gameplay. Learners practice tone recall, perception and production with kinesthetic touch input and voice-based speech input. Furthermore, learners compete with native speakers, achieving mastery [2] over the accuracy and rhythm of their own tone production. We hypothesize that if learners are able to achieve native-level proficiency for a collection of phrases, they will be positively motivated towards further language learning. ToneWars makes use of gameplay elements from popular mobile games like Tetris and Fruit Ninja to provide an engaging experience for both native speakers and learners.

Fig. 1. A CSL learner and native speaker competing through ToneWars

To our knowledge, ToneWars is the first language learning system to connect second language learners and native speakers via collaborative mobile gameplay. In this paper, we discuss the design, implementation, and evaluation of ToneWars. First, we describe our field study of tone teaching, consisting of semi-structured interviews with CSL instructors. Second, we discuss the design of ToneWars in detail, highlighting discoveries from the iterative design process. Finally, we describe our 24-subject user study and evaluate its results to determine the system's learning outcomes.

2 Related Work

2.1 Mobile Language Learning Systems

Smartphones are an ideal platform for language learning [5, 6], [10] due to their portability, connectivity, and affordability, even in developing countries [8, 9]. Researchers have built compelling learning applications that leverage the unique

capabilities of mobile devices, including recording media [10], location awareness [6], and speech recognition [9]. These applications have been built for English as a Second Language (ESL) learners [8, 9], CSL learners [5, 6], and even native speakers of Chinese in elementary schools [10].

The Multimedia Word and Drumming Strokes mobile games by Tian et al [10] enable group-based Chinese learning. These games are played by co-located, native Chinese speaking children who share a single mobile device. ToneWars differs from these games in that during its collaborations, language learners and native speakers are physically separated, communicating over network from separate devices.

Tip Tap Tones [5] by Edge et al. trains CSL learners to perceive Chinese tones through mobile games. Tip Tap Tones provides single player flashcard-style drills at a character-by-character level. ToneWars seeks to build on this work in several ways. First, in ToneWars players can use touch gestures and speech to input tones, providing practice opportunities similar to classroom exercises. Second, it enables phrase-level tone learning. Third, it connects CSL learners with native speakers in collaborative multiplayer gameplay in order to foster greater learning motivation.

2.2 Systems with Users of Multiple Languages

Duolingo [4] by von Ahn et al is a platform for crowdsourcing translations. Duolingo aims to produce high quality, low cost translations by breaking translation tasks into free, bite-sized educational exercises for learners of both the source and target languages. *MonoTrans* [12] by Hu et al uses two-way machine translation and two groups of monolingual users to achieve low cost translation through iterative collaboration. *Busuu.com* is an L2 learning community where learners can act as experts of their own native language *voluntarily*, communicating with other learners of their language through text or video chat and revising others' written exercises. ToneWars, in contrast, brings together learners with native speakers in a mobile setting, motivating L2 learners to engage in learning through competitive gameplay.

3 Field Study

To identify opportunities to include CSL tone pedagogy in designing ToneWars, we conducted semi-structured interviews with two experienced CSL instructors (*T1* and *T2*) from local universities. *T1* and *T2* each had 6 or more years of CSL teaching experience with more than 200 CSL learners at the entry and intermediate levels. We took several observations from these interviews:

F1. Tones Require Continual Practice for Clean Production. Students were exposed to tone and pronunciation practice in the first 1-2 weeks of instruction (*T1* and *T2*). After this, students' tone perception and production deteriorated after a switch in material and the continuing influence of the learner's native language (*T1* and *T2*). At later stages, dedicated practice and personalized feedback could help restore cleaner tone production students achieved in the first weeks. Challenges to tone recognition varied with learning level. According to *T1*, beginner students frequently confused *tone 2* and *tone 4,* and later they confused *tone 1* and *tone 4*. Advanced CSL learners had difficulty recognizing and pronouncing tones in long phrases (*T2*).

F2. Tone-Tracing Hand Gestures May Provide Motivation and Attention. In CSL classrooms, instructors may have students trace tones of characters with their hands as they pronounce them (*T1* and *T2*). Instructors saw several advantages to using these gestures. First, kinesthetic gestures could reinforce learners' memories of the sounds of characters [15]. Second, hand gestures may serve as visual aids to help students recognize incorrect pronunciation. Third, tone tracing might make the classroom more engaging. One instructor had students trace tones with their hands, heads, and feet to make class more fun (*T1*). However, some research suggests the "social acceptability" problem [16] may prevent CSL learners from tracing tones when practicing in public.

F3. Learners Can Benefit from Interaction with Native Speakers. After students engaged in conversation outside the classroom, the instructors believed they gained confidence they could carry into the classroom (*T1* and *T2*). Through these experiences, learners improved their tones and pronunciation, especially at later levels (*T2*). However, instructors saw limitations to these interactions. They could cause frustration: "*the native speaker [could] start with some language that is beyond the comprehension level of the student*" (*T2*). Second, the native speaker's feedback may not be helpful for the student at her current learning level: "*[T]he native speaker is not a teacher who knows what the student should start with*" (*T2*).

Our interviews confirmed the importance and challenges of tone learning for CSL learners. We have designed ToneWars to build upon the premises above to improve the quality of the educational experience. In particular, ToneWars leverages kinesthetic learning and interactions between learners and native speakers.

4 The Design of ToneWars

4.1 Gameplay

We created ToneWars through two rounds of paper prototyping and three rounds of iterative design. In ToneWars, a player[2] must quickly and accurately eliminate Chinese phrases that fall from the top of the screen. A phrase is eliminated only when players enter the correct tone for each character in the phrase sequentially.

Fig. 2. ToneWars gameplay (left to right). (a) Phrases fall and collide; (b) The player selects a phrase; (c) The player traces a tone with a touch gesture; (d) A character locks after an incorrect guess; (e) The phrase stack overflows and clears.

[2] A player is either a second language learner or a native speaker.

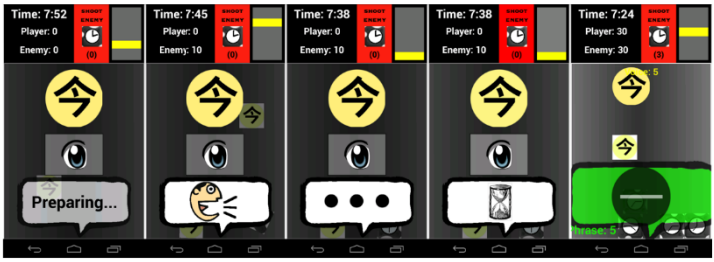


Fig. 3. Voice input for a character (left to right). (a) Recognition starts; (b) ToneWars listens for start of speech; (c) ToneWars listens to speech; (d) Processing speech; (e) Match feedback.

Tones are entered through two input modalities: (1) kinesthetic touch gestures **(Fig. 2)** and (2) voice-based speech recognition **(Fig. 3)**.

Phrases on screen collide with each other and stack up as they fall (**Fig. 2a**). The game penalizes players by halving the score when they cannot eliminate phrases fast enough to prevent phrases from filling the whole screen (**Fig. 2e**).

If the player enters the incorrect tone for a character, the phrase is locked (**Fig 2d**). To unlock it and continue to the next character, the player must first play back a hint (either visual or audio) that describes the correct tone. Then the player enters the correct tone 4 times. This process emulates language learning drill exercises.

Players earn points when they enter a tone correctly, eliminate a phrase, or view hints. Learners compete for higher scores against native speakers. Each player can fire blocks to clutter the other's screen. They earn one block for each phrase eliminated, and can save these blocks to fire all at once to cause maximal damage. Players monitor how cluttered the opponent's screen is through a preview pane in the top-right corner of the screen. When an opponent is not available, ToneWars allows a player to compete with an in-game AI. In the future they will be able to play against pre-recorded action scripts recorded from native speaker players we have collected.

4.2 Tone Input Modalities

ToneWars offers two unique modes for entering tones. The first input mode is kinesthetic touch gestures. Players use their fingers to slash the shapes of tone marks for the characters on the screen. This control scheme was inspired by the "in-the-air" gestures CSL instructors used in the classroom to engage students. We believed learners could experience the same benefits of engagement and aural attention as in class with a touch-based input that is more "socially acceptable" [16] in public.

Rather than training a general-purpose handwriting recognizer for gesture classifications, we created a decision tree based recognizer. This was for three reasons. First, ToneWars only needed to recognize four pre-defined, simple shape tone marks.[3] A heuristic-driven decision tree could easily detect the distinct shape of each mark. Second, the decision tree could be fine-tuned to match user control

[3] The 5th 'neutral' tone gesture, which occurs infrequently, is executed as a tap gesture.

patterns we observed without requiring large amounts of training data. Third, it could allow us to determine the exact cause of why each user's misrecognized gestures were falsely classified. Instead of receiving a cryptic rating for each classification, we could compare the user's touch gesture paths to the hard rules defined in the decision tree.

The second tone input mode is speech (**Fig. 3**). Players enter tones by pronouncing characters into the phone's microphone. If the player does not know a character but wants to guess its tone, she can pronounce a placeholder character with the expected tone. This control method had two advantages. First, oral production helps learners move from declarative knowledge of a language (recognizing words) to productive knowledge (using the words correctly) [3]. Second, by requiring students to guess the sound of unknown characters, they may be able to develop a deeper knowledge of the relationship of a character's written radicals[4] to its pronunciation.

Our tone recognition engine is built on top of Google's speech recognition service for Mandarin, a state-of-the-art speech recognition engine. A mobile device must have Internet access to use all features of the recognition service. In the future, a customized on-device Mandarin tone recognizer could be developed when the network is not available, like the English recognizer by Kumar et al [9].

4.3 Implementation

ToneWars was written in Java for Android 4.0. We used the AndEngine (http://www.andengine.org/) library to speed up game programming. We also used the Box2D physics engine to implement the falling and collision effects in ToneWars. Excluding third party libraries, ToneWars has a total of 11,150 lines of code in Java.

5 User Study

We conducted a 24-subject user study (12 CSL and native speaker pairs) to understand the performance and usability of ToneWars. Before the study, we worked with a CSL instructor to select 25 phrases with a total of 25 unfamiliar characters. We sorted the phrases into 5 groups of equal difficulty. Each phrase group had 5 characters that would be unfamiliar to CSL learners with less than 1 year of experience with *Integrated Chinese*, a popular CSL textbook in North America. No phrase had more than two unfamiliar characters. Although some Chinese characters have tones that can vary based on context, no such characters were chosen for these groups.

The user study consisted of three steps:

1) *Pre-test*. Participants completed a quiz in which they determined tones of 35 characters. The set contained all 25 unfamiliar characters as well as 10 familiar characters from the phrase groups. If participants did not know the tone, they were told it was okay to leave the space for response blank.

[4] According to Wang [18], although Chinese writing is logographic, 77% of characters in modern Chinese have radicals that suggest their pronunciation.

2) *Gameplay sessions*. Participants were grouped into learner-native speaker pairs. They played 5 rounds of ToneWars. In the first 4 rounds, users controlled the game with touch-based tone tracing. The rounds were structured according to a 2-by-2 within-subject design based on hint feedback mechanisms (*visual* vs. *audio*) and collaboration mode (*single-player with AI* vs. *CSL learner with native speaker*). The order of conditions was counter-balanced with a Latin-square pattern across pairs. In the 5th round, participants used speech to input tones. Each round presented users with phrases from 1 of 5 phrase groups selected above. By measuring subjects' improvement in tone recall for each of these phrase groups, we sought to measure the impact of the above design conditions on the learning experience.

Participants were seated in adjacent chairs facing each other. For rounds that utilized voice control and audio feedback, players were instructed to wear headphones to minimize audio interference. Each round took 8 minutes to complete.

3) *Post-test*. Participants completed a quiz identical to the pre-test after finishing the game sessions. All CSL learners and native speakers completed both tests.

We recruited 24 participants (12 CSL learner and native speaker pairs) from two local universities. Among the learners (7 female, median age 19), a majority had less than one year of formal CSL learning experience. The native speakers (7 female, median age 22.5) were mostly undergraduate or graduate students originally from China.

Participants used one of two Google Galaxy Nexus smartphones. The device has a 4.65-inch, 720 x 1280 pixel display, 1.2-GHz dual core ARM Cortex-A9 processor, and runs Android OS. Devices were connected to the Internet through 802.11g Wi-Fi.

6 Results and Discussion

We observed a difference in initial tone identification ability for learners and native speakers. During the pre-test, CSL learners correctly recognized 11.6 / 35 characters (*min* = 1, max = 25, σ = 7.9, *accuracy* = 33%[5]). In comparison, 11 / 12 native speakers scored 35 / 35, and one native speaker scored 19 / 35 (μ = 33.7, σ = 4.6, *accuracy* = 96%). The difference is statistically significant ($t = 8.38$, $p < 10^{-6}$).

After 40 minutes of gameplay, learners could recognize tones of 17.8 characters (*min* = 5, *max* = 34, σ = 10.4, *accuracy* = 51%). The average recall gain for learners was 6.2 characters (*min* = 1, *max* = 13, σ = 3.5). A pair-wise *t*-test showed this improvement is statistically significant ($F_{1,11} = 6.13$, $p < 10^{-4}$). The native speaker that did not score perfectly in the pre-test properly identified 8 more tones in the post-test.

For all experimental conditions, repeated ANOVA showed significant gain for CSL learners in recall of the tones of unfamiliar characters **(Fig. 4).** The gains were: visual feedback, 2.7 tones ($t = 6.4$, $p < 10^{-4}$); audio feedback, 2.35 tones ($t = 4.4$, $p < 0.005$); competition with AI, 2.4 tones ($t = 5.7$, $p < 10^{-4}$); learner vs. native speaker, 2.7 tones ($t = 4.5$, $p < 0.005$); voice control, 1.1 tones ($t = 3.0$, $p < 0.05$). The difference in improvement for hint feedback types (audio vs. visual) was not significant ($F_{1,11} = 0.54$, $p = 0.60$). Although learners recalled 0.3 more tones in rounds against

[5] With 5 possible tones, the chance of randomly guessing a character's tone is 20%.

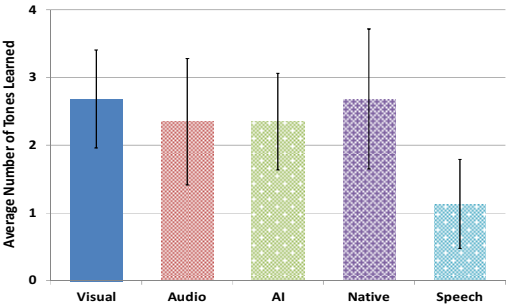


Fig. 4. Average numbers of tones learned by condition. Error bar shows one standard deviation

native speakers, this difference was not significant ($F_{1,11}$ = 0.47, p = 0.65). We attribute the lack of significance to small sample size (12 CSL learners) and the large difference in learners' initial ability (ranging from 1 to 25 out of 35 correct on the pre-test). Because of this initial variation, our current results could show a gain in recall for each condition, but could not uncover the relative strength of any one condition. We plan to address this in the near future by running classroom-level, multi-week deployments.

Although nearly all learners had lower initial tone identification accuracy than the native speakers they were paired with, 8 / 24 (33%) rounds between learners and speakers were won by learners. We attribute this to certain learners' ability to become comfortable with the control mechanics, quickly master unknown or forgotten tones, and to develop successful attacking strategies.

Participants' qualitative feedback on ToneWars was highly favorable. Ratings of user perceptions were measured on a 5-point Likert scale (1 = *strongly disagree*, 5 = *strongly agree*). CSL learners unanimously rated ToneWars' ease of use at 5, and native speakers rated it 4.75 (σ = 0.45). Learners rated ToneWars' engagement as 4.5 (σ = 0.52) and native speakers rated it 4.25 (σ = 0.75). Both native speakers (μ = 4.17, σ = 0.83) and CSL learners (μ = 4.25, σ = 0.75) enjoyed playing against a real-life partner. Compared to native speakers (μ = 3.41, σ = 1.24), learners (μ = 4.33, σ = 0.98) indicated a stronger interest to play ToneWars in their spare time.

In written comments, players were positive about ToneWars. One participant reported, "*I would love to play a game like this to help my pronunciation and tonal recognition. I am always looking for new ways to learn Chinese.*" Others enjoyed the competition. One told us that when they played against a native speaker, "*it was more competitive than playing against the computer and for me, points have the main motivation for me to focus on the game.*" As one learner expressed, the social aspect of competition could be appealing: "*Having a real life opponent is always, in my opinion, fun because of being able to make fun conversation while playing is a plus.*"

Native speakers hoped that the challenge of ToneWars could be increased by introducing more phrases per round. Several participants reported moments of frustration with input control. In some cases, tone gestures or spoken tones were misrecognized due to cases we had not foreseen (left-handedness for touch) or user skill level (spoken tones of learners with insufficient previous tone training were often

misclassified). During speech input rounds, characters spoken aloud may have interfered with the device of a player's co-located participant. In the future, we plan to improve the maturity of tone tracing and speech recognition to address these cases. We also hope to explore whether learner motivation against native speakers comes from perceived competition or from a change in game dynamics during these rounds.

7 Conclusions and Future Work

ToneWars is a group-based mobile game for CSL learners to master Mandarin tones through collaboration with native speakers. The design of ToneWars, including tone-tracing gestures, speech input, and collaboration with native speakers, is inspired by effective second-language pedagogy. Our design discoveries can be applied for researchers building mobile learning technology to aid L2 acquisition beyond just Chinese. In a 24-subject study, we confirmed ToneWars' usability and efficacy. We observed a 6.2-tone average gain in short term recall for second language learners who played around 40 minutes of ToneWars.

We plan to continue our work with ToneWars in multiple ways. We will address usability problems identified above, improving speech recognition accuracy and incorporating algorithms to support performance-based adaptive learning. We hope to extend the potential of ToneWars by adding material that complements the regular CSL curriculum and by exploring its design principles for languages other than Chinese. While our current lab study shows promising results for improving learners' short-term recall, we hope to evaluate ToneWars' feasibility and educational benefits in larger scale, longitudinal deployments.

References

1. ACTFL: Foreign Language Enrollments in K–12 Public Schools: Are Students Prepared for a Global Society? Technical Report, American Council On The Teaching of Foreign Languages (2011)
2. Bloom, B.S.: Mastery learning. In: Block, J.H. (ed.) Mastery learning: Theory and practice, pp. 47–63. Holt, Rinehart and Winston, New York (1971)
3. De Bot, K.: The psycholinguistics of the Output Hypothesis. Language Learning 46, 529–555 (1996)
4. Duolingo, `http://www.duolingo.com`
5. Edge, D., Cheng, K., Whitney, M., Qian, Y., Yan, Z., Soong, F.: Tip Tap Tones: Mobile Microtraining of Mandarin Sounds. In: ACM MobileHCI 2012, pp. 427–430. ACM, New York (2012)
6. Edge, D., Searle, E., Chiu, K., Zhao, J., Landay, J.: MicroMandarin: Mobile Language Learning in Context. In: ACM Conference on Human Factors in Computing Systems, pp. 3169–3178. ACM, New York (2011)
7. Fu, I.-P.: Student approaches to learning Chinese vocabulary. Ph.D. thesis, Virginia Polytechnic Institute and State University (2005)

8. Kam, M., Ramachandran, D., Devanathan, V., Tewari, A., Canny, J.: Localized iterative design for language learning in underdeveloped regions: The PACE framework. In: ACM Conference on Human Factors in Computing Systems, pp. 1097–1106. ACM, New York (2007)
9. Kumar, A., Reddy, P., Tewari, A., Agrawal, R., Kam, M.: Improving Literacy in Developing Countries Using Speech Recognition-Supported Games on Mobile Devices. In: ACM Conference on Human Factors in Computing Systems, pp. 1149–1158. ACM, New York (2012)
10. Tian, F., Lv, F., Wang, J., Wang, H., Luo, W., Kam, M., Setlur, V., Dai, G., Canny, J.: Let's Play Chinese Characters – Mobile Learning Approaches via Culturally Inspired Group Games. In: ACM Conference on Human Factors in Computing Systems, pp. 1603–1612. ACM, New York (2010)
11. Thorne, S., Black, R., Sykes, J.: Second Language Use, Socialization, and Learning in Internet Interest Communities and Online Gaming. The Modern Language Journal 93(s1), 802–821 (2009)
12. Hu, C., Bederson, B., Resnik, P.: Translation by Iterative Collaboration between Monolingual Users. In: Graphics Interface, pp. 39–46 (2010)
13. Lee, L.: Learners' Perspectives on Networked Collaborative Interaction with Native Speakers of Spanish in the US. Language Learning & Technology 8(1), 83–100 (2004)
14. McLaughlin, B.: Fostering second language development in young children: Principles and practices. NCRCDSLL Educational Practice Reports (1995)
15. Morett, L.M., Gibbs, R.W., MacWhinney, B.: The Role of Gesture in L2 Learning: Communication, Acquisition, & Retention. In: 34th Annual Conference of the Cognitive Science Society, pp. 773–778. Cognitive Science Society, Austin (2012)
16. Rico, J., Brewster, S.A.: Usable Gestures for Mobile Interfaces: Evaluating Social Acceptability. In: ACM Conference on Human Factors in Computing Systems, pp. 887–896. ACM, New York (2010)
17. Rivera, G., Matsuzawa, C.: Multiple-language program assessment: Learners' perspectives on first and second-year college second language programs and their implications for program improvement. Foreign Language Annals 40 (2007)
18. Wang, N.: The Structure and Meaning of Chinese Character and Word. In: Research on Chinese Cognition. Shandong Education Press (1997) (in Chinese)
19. Xing, J.: Teaching and Learning Chinese as a Foreign Language. Hong Kong University Press (2006)

Gamification of Joint Student/System Control over Problem Selection in a Linear Equation Tutor

Yanjin Long and Vincent Aleven

Human Computer Interaction Institute, Carnegie Mellon University,
5000 Forbes Avenue, Pittsburgh, PA, 15213
{ylong,aleven}@cs.cmu.edu

Abstract. Integrating gamification features in ITSs has become a popular theme in ITSs research. This work focuses on gamification of shared student/system control over problem selection in a linear equation tutor, where the system adaptively selects the problem *type* while the students select the individual problems. In a 2x2+1+1 classroom experiment with 267 middle school students, we studied the effect, on learning and enjoyment, of two ways of gamifying shared problem selection: performance-based rewards and the possibility to re-do completed problems, both common design patterns in games. We also included two ecological control conditions: a standard ITS and a popular algebra game, *DragonBox 12+*. A novel finding was that of the students who had the freedom to re-practice problems, those who were not given rewards performed significantly better on the post-tests than their counterparts who received rewards. Also, we found that the students who used the tutors learned significantly more than students who used *DragonBox 12+*. In fact, the latter students did not improve significantly from pre- to post-tests on solving linear equations. Thus, in this study the ITS was more effective than a commercial educational game, even one with great popular acclaim. The results suggest that encouraging re-practice of previously solved problems through rewards is detrimental to student learning, compared to solving new problems. It also produces design recommendations for incorporating gamification features in ITSs.

Keywords: *DragonBox*, educational games, student control, shared control, intelligent tutoring systems, algebra, classroom evaluation, rewards.

1 Introduction

In recent years, Intelligent Tutoring System (ITS) researchers have started to investigate how to integrate game elements within a tutoring environment. The goal is typically to make the system more engaging for students, while maintaining its effectiveness in supporting learning. Empirical studies have been conducted to evaluate the effects of gamifying tutors on students' learning and motivation, as well as to explore the best design to incorporate game elements in tutors. Some studies have found that game-based learning environments can significantly enhance students' learning outcomes [3, 10] and can produce the same learning effects as nongame tutors [7]. However, gamification of ITSs is not always successful. For example, one study [5]

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 378–387, 2014.

found that tutor-like assistance led to better learning and interest as compared to game-like assistance in an educational game of policy argument. Therefore, gamification of ITSs should be done with care, where possible informed by empirical studies.

Student control over problem selection may be an interesting area for gamification. *Full* student control over problem selection tends to be detrimental for learning (see e.g., [2]). However, *shared* control between student and system has shown some promise. Simple forms of shared control, in which the system and the students share the responsibilities to select problems in the system, had led to comparable learning as full system control [4, 9]. However, these simple techniques may not be as engaging as they could be, nor do they take full advantage of ITSs' ability to make good problem selection decisions. In the current work, we focus on a form of shared control in which the system selects problem types and decides when students have mastered each problem type and may go on to the next, while the student selects individual problems from a certain problem type. We try to improve on this form of shared control by adding gamification features, and investigate whether the gamified shared control leads to higher engagement and better learning.

Commercial games provide plenty of ideas for gamification of problem selection. A feature found in many popular games (e.g., *Angry Birds, DragonBox*) is the possibility to re-do problems after they have been completed. This feature is often combined with rewards (such as a number of stars) that reflect performance on the given problem. One reason players may elect to re-do a problem is to increase the rewards. According to theories of autonomy in learning [6], allowing re-practice gives students more freedom, which could possibly enhance their engagement in learning. Moreover, re-practicing could lead to more efficient acquisition of problem-solving skills, although to the best of our knowledge that has not been established definitively in the cognitive science literature. On the other hand, frequent re-practice may reduce problem variability and therefore be detrimental for learning [11]. Empirical investigation of the effectiveness of these gamification features is therefore warranted.

In the current work, we investigate the effects of gamifying shared student/system control in our linear equation tutor, *Lynnette*. We investigated two gamification features: giving students the freedom to re-practice previously completed problems (not allowed e.g., in standard Cognitive Tutors) and rewards (stars) for each problem based on students' performance. These features are similar to *Angry Birds'* or *DragonBox*' problem selection and rewards systems. We hypothesize that 1) the possibility to re-practice problems, added to shared control over problem selection will enhance students' learning and engagement; 2) rewards based on students' performance on individual problems will also lead to better learning and engagement. Consequently, we created four experimental versions of *Lynnette* to evaluate the effects of the two gamification features. Moreover, we included two ecological control conditions in the study: a standard ITS and a commercial algebra game. The standard ITS is a control version of *Lynnette* without any gamification features and with full system control over problem selection (as is common in e.g. Cognitive Tutors). The algebra game is *DragonBox*, which has attracted substantial public attention for allegedly helping young children learn algebra in a very short period of time [8, 12]. Although *DragonBox* has been the subject of at least one research study [1], we are not

aware of any studies that empirically investigated its effectiveness in teaching algebra. Given the publicity surrounding the game, it would be good to know how educationally effective and engaging it is, compared to technology proven to be effective in helping students learn (i.e., an ITS). We conducted a classroom experiment with 267 middle school students to investigate our hypotheses.

2 Methods

2.1 Lynnette and DragonBox 12+

***Lynnette* – Web-Based Linear Equation Tutor on Android Tablet.** *Lynnette* is a tutor for basic equation solving practice. It comprises five levels with increasingly difficult equations, starting with equations of the form $x + a = b$ and their variations at Level 1 and ending with equations of the form $a(bx + c) + d = e$ and their variations at Level 5. Students are required to explain some of their steps by indicating the main transformation (see Fig. 1). The problems in *Lynnette* do not require fractions and the tutor does not allow strategies that involve fractions along the way. Otherwise, it is flexible in the major and minor strategy variants that it recognizes. It also allows some suboptimal strategies, while warning students about them in the hint window (see Fig. 1), on the assumption that students can learn from seeing and being explicitly reminded of suboptimal strategies. It does not allow mathematically correct but useless transformations. *Lynnette* was designed to run on Android tablets but also runs on regular desktop computers. It was implemented as a rule-based Cognitive Tutor using the Cognitive Tutor Authoring Tools (http://ctat.pact.cs.cmu.edu/). Its cognitive model comprises 73 rules. *Lynnette* is the first CTAT-built tutor that runs on Android tablets and the first elaborate CTAT-built rule-based tutor used in classrooms.

Fig. 1. The problem solving interface of *Lynnette* on a Samsung Galaxy Tablet

DragonBox 12+. We used the Android version of *DragonBox 12+* in the study, which is one of the two *DragonBox* games that targets middle and high school algebra. It has 10 progressive chapters, each with 20 problems, covering 24 algebraic rules [13]. The two sides of the screen represent the two sides of an equation. The game provides immediate step-by-step feedback. It starts by hiding the algebraic expressions and the players have to isolate a box on one side of the screen through moving cards (Fig. 2, leftmost). It gradually transitions to algebraic problems as the students progress in the game (Fig. 2, middle and rightmost). As claimed on its official site, students can learn basic algebra in one hour with *DragonBox*.

Fig. 2. Screenshots of *DragonBox* from its official site (©WeWantToKnow)

2.2 Experimental Design, Participants, Procedure and Measurements

We conducted an experiment with a 2x2+1+1 design with a total of six conditions. The 2x2 design varies two factors: 1) whether or not the students are able to access and re-practice completed problems; and 2) whether or not the tutor shows rewards to the students. The two "+1" conditions are a popular algebra game, *DragonBox 12+*, and a standard ITS.

Table 1. Experimental conditions in the study. RePr stands for Re-Practice, NoRePr stands for no Re-Practice, Rwd stands for Rewards, and noRwd stands for no Rewards.

	RePr +Rwd	**No-RePr+Rwd**	**RePr +noRwd**	**No-RePr+noRwd**	***Dragon-Box 12+***	**Control *Lynnette***
Re-practice	Yes	No	Yes	No		
Rewards	Yes	Yes	No	No		

We created four experimental versions of *Lynnette* and a control version (as listed in Table 1). The five *Lynnette* tutors all used the same interface for problem solving, shown in Figure 1. Also, all five tutor versions employed Bayesian Knowledge Tracing and Cognitive Mastery as part of their problem selection methods. The control version used it for full system control, as is customary in Cognitive Tutors. That is, in this version the tutor always selected the next problem for the student from level 1 to level 5. The four experimental versions used Bayesian Knowledge Tracing and Cognitive Mastery for shared control. In these versions, the students also had to do the levels in order. Within a level, they could select which problem to do next. The tutor decided when a level was complete (namely, when all skills were mastered).

The system presented one or two screens in-between problems, which vary according to the two experimental factors. All four experimental tutor versions had a problem selection screen, which lists the problems within the current level. On this screen, the student selected the next problem (Fig. 3, right). In the two Re-Practice conditions, the system "recommended" problems on this screen by displaying a flag next to them. These problems had unmastered skills, according to the tutor's Bayesian Knowledge-Tracing method, and had not been practiced yet by the given student. However, students were free to select a problem with or without a flag. Also in the two Re-Practice conditions, students could select any problem available on the given level, regardless of whether they had completed them previously. By contrast, in the No Re-Practice conditions, the previously-practiced problems were grayed out so they could not be selected again. In the two Rewards conditions, students saw an additional screen between problems (Fig. 3, left), a problem summary screen showing earned stars after completing each problem, based on the number of steps, hints and errors. A trophy could be earned for perfect performance. Further, in these conditions, the problem selection screen listed the rewards earned (see Fig. 3, right). After re-practice, the number of rewards would be updated.

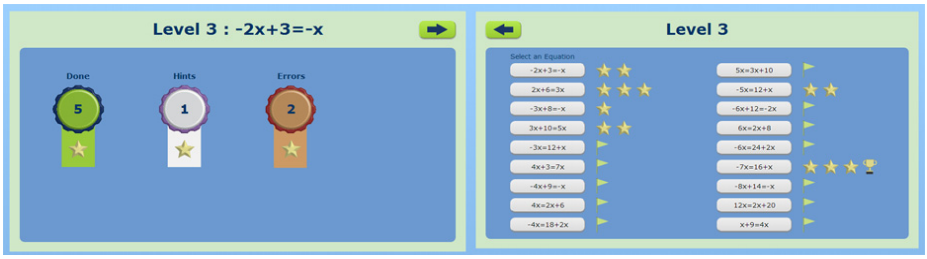


Fig. 3. Problem summary screen with rewards (left) and problem selection screen (right)

267 7th and 8th grade students participated in this study. They were from 15 classes of 3 local public middle schools, taught by 6 teachers. Students from each class were randomly assigned to one of the six conditions. All students completed a 20-minute paper pre-test on the first day of the study. They then worked for 5 42-minute class periods on consecutive school days either with one of the *Lynnette* versions or *DragonBox 12+* using Samsung Galaxy tablet PCs. All students took an immediate paper post-test after the five class periods. The pre- and post-tests were in the same format, which consisted of 6 equations that measured students' procedural skills of solving linear equations[1]. *Lynnette* only provides practice for a subset of problem types that are practiced in *DragonBox 12+*. Therefore, among the 6 equations, 4 were shared types of equations between *Lynnette* and *DragonBox 12+*, while 2 were types of equations practiced in *DragonBox 12+* only. Documentation of *DragonBox 12+* indicates that the algebraic rules that are needed to solve the 6 procedural items could

[1] The test forms also included items testing basic conceptual knowledge of algebra. However, because there was no improvement from pre-test to post-test on these items in any of the conditions (similar to what we saw in past studies), we do not report the results separately.

be practiced by Level 6 in the game [13]. We created two sets of equivalent test forms and administered them in counterbalanced order. We also included a 7-question questionnaire to measure students' enjoyment of using *Lynnette* or *DragonBox* along with the post-test. The questions were adapted from the interest/enjoyment subscale of the Intrinsic Motivation Inventory, and were all based on a 7-point Likert scale.

3 Results

A total of 190 students were present on each day of the study and completed the pre- and post-tests. Given that the sample was nested in 15 classes, 6 teachers, and 3 schools, Hierarchical Linear Modeling (HLM) was used to analyze the test data. We constructed 3-level models in which students (level 1) were nested in classes (level 2), and classes were nested in teachers (level 3; 4-level models indicated little variance on the school level, so we built 3-level models). Specifically, for the learning effects from pre- to post-tests, we used both pre- and post-test scores as dependent variables to fit this model: $score_{ij} = test_j + student(class)_i + class(teacher)_i + teacher_i$, where $score_{ij}$ was $student_{ij}$'s score on $test_j$, and $student(class)_i$, $class(teacher)_i$ and $teacher_i$ indicated the nested sources of variability in the hierarchical model. To evaluate the main effects and interaction effect across the conditions on the post-test, we modified the model and used $student_i$'s pre-test score pre_i as co-variate: $post\text{-}score_i = pre_i + tutor_j + rewards_k + re\text{-}practice_l + rewards_k{*}re\text{-}practice_l + student(class)_i + class(teacher)_i + teacher_i$, with $tutor_j$ being whether the condition learned with a tutor or *DragonBox 12+*, $rewards_k$ being whether the tutor condition received rewards, $re\text{-}practice_l$ being whether the condition allowed re-practice, and $rewards_k{*}re\text{-}practice_l$ being the interaction between the two factors. We report Cohen's *d* for effect sizes. An effect size *d* of .20 is typically deemed a small effect, .50 a medium effect, and .80 a large effect.

Table 2. Means and SDs of all conditions on pre- and post-tests for the shared procedural items, game (*DragonBox)* only procedural items, and the overall test scores

	RePr+Rwd	NoRePr+Rwd	RePr+noRwd	NoRePr+noRwd	*Dragon-Box 12+*	Control *Lynnette*
Pre-shared	.364 (.249)	.327 (.279)	.327 (.257)	.364 (.313)	.321 (.209)	.386 (.277)
Post-shared	.467 (.291)	.491 (.276)	.497 (.364)	.471 (.311)	.366 (.289)	.538 (.347)
Pre-game	.324 (.345)	.266 (.359)	.318 (.350)	.318 (.344)	.331 (.382)	.288 (.330)
Post-game	.352 (.320)	.281 (.358)	.313 (.307)	.300 (.323)	.310 (.410)	.297 (.356)
Pre-overall[2]	.439 (.178)	.413 (.142)	.403 (.183)	.477 (.172)	.422 (.133)	.418 (.155)
Post-overall	.463 (.160)	.491 (.173)	.520 (.203)	.503 (.167)	.438 (.161)	.477 (.190)

[2] Pre-overall and Post-overall include the conceptual items along with the 6 procedural items.

Learning Effects of *Lynnette* and *DragonBox*. Table 2 shows the average test scores for all conditions on the 4 shared procedural items, the 2 *DragonBox*/game only procedural items, and the overall test scores including the conceptual items. Students in the *DragonBox* condition completed an average of 140 equations in the game by the end of the 5th period, which is equivalent to finishing Level 7. Students from all five *Lynnette* conditions completed an average of 36 equations. All five *Lynnette* conditions together improved significantly on the shared procedural items ($t(300)=4.543$, $p<.001$, $d=.52$) as well as the overall test scores ($t(300)=3.305$, $p=.001$, $d=.38$), but did not improve on the game only items. The best tutor condition, RePr+noRwd also improved significantly on the shared items ($t(41)=2.392$, $p=.021$, $d=.75$), and the overall test scores ($t(41)=3.088$, $p=.004$, $d=.96$). By contrast, the *DragonBox* students did not show significant improvement on any of the three categories of test items from pre- to post-test. When comparing the post-test scores between the *Lynnette* conditions and *DragonBox*, the five *Lynnette* conditions together significantly outperformed the *DragonBox* condition on both the shared items ($t(167)=2.118$, $p=.036$, $d=.33$) and all 6 procedural items together (i.e. shared items + game-only items, $t(167)=1.986$, $p=.049$, $d=.31$). The RePr+noRwd condition also significantly outperformed the *DragonBox* condition (shared items: $t(37)=2.214$, $p=.033$, $d=.73$; all 6 procedural items: $t(37)=2.295$, $p=.027$, $d=.75$). We also compared students' post-test scores between the control *Lynnette* and the experimental *Lynnette* tutors. There were no significant differences on any of the categories of test items.

Effects of Re-Practice and Rewards. We tested the main effects and interaction of the two factors with the four experimental *Lynnette* tutors. Neither re-practice nor rewards showed a significant main effect. The interaction between the two was significant for the overall test scores ($t(104)=-2.287$, $p=.024$). Post-hoc analysis revealed that for the two Re-Practice conditions, students who did not see rewards (i.e., RePr+noRwd) performed significantly better than students who received rewards (i.e., RePr+Rwd, $t(41)=-2.311$, $p=.026$, $d=.72$). On the other hand, there was no significant difference between the two No-Re-Practice conditions (i.e., NoRePr+Rwd and NoRePr+noRwd). To explore the mechanism behind the difference between the two Re-Practice conditions, we investigated how often the students re-practiced the completed problems. Seven out of 31 (22.58%) students in RePr+noRwd re-practiced a total of 9 problems start-to-finish, whereas 16 out of 33 (48.48%) students in RePr+Rwd re-practiced 37 problems start-to-finish. We also investigated the number of times students re-started a problem they had solved before, regardless of whether they actually finished it. Specifically, we calculated the ratio of (number of re-starts)/(number of total problem visits) for each student in the two Re-Practice conditions. The average ratio was .196 (SD=.172) for RePr+Rwd and .115 (SD=.074) for RePr+noRwd, with a significant difference between the two ($t(42)=2.858$, $p=.007$, $d=.88$). In other words, students in RePr+Rwd re-started significantly more problems than students in RePr+noRwd. Moreover, the correlation between the ratio of re-starts and students' post-test performance was -.277 ($p=.028$), controlling for the overall pre-test score. The more times the students re-started problems, the less they learned.

Enjoyment. Table 3 shows the average ratings of enjoyment from the intrinsic motivation questionnaire handed out with post-test. The *DragonBox* students provided

significantly higher ratings of enjoyment while playing with the game, as compared to all the *Lynnette* conditions taken together (t(168)=-3.315, p=.001, d=.51). No significant main effects or interaction effect of re-practice and rewards were found for enjoyment among the experimental *Lynnette* tutors. The difference between the experimental *Lynnette* tutors and the control *Lynnette* was not significant either.

Table 3. Means and SDs of the enjoyment ratings across all 7 questions for all conditions

	RePr+ Rwd	NoRePr+ Rwd	RePr+ noRwd	NoRePr+ noRwd	*Dragon-Box 12+*	Control *Lynnette*
Enjoy-ment	3.815 (1.627)	3.884 (1.572)	4.166 (1.398)	4.372 (1.528)	5.099 (1.448)	4.138 (1.483)

4 Discussion and Conclusion

Gamifying ITSs to foster higher engagement and perhaps even better learning outcomes has become a popular theme in the ITS community. However, what gamification features are beneficial and how to integrate them with existing tutor features remains a challenging question. Our study found that gamification of shared student/system control was a partial success. The two gamification features held up well in the classroom but did not foster the expected higher enjoyment or learning gains. We did not find a significant difference between the experimental (gamified) *Lynnette* tutors and the control *Lynnette* with respect to enjoyment or learning. One of the gamified conditions (RePr+noRwd) had the highest learning gains, with a greater pre/post effect size (d=.96) than that for all *Lynnette* tutors (d=.38), but was not reliably better on any measure than the control tutor. Thus, gamifying tutors by incorporating common game design patterns does not automatically make them more effective. This finding is not uncommon. As discussed in the introduction, efforts at gamifying tutors frequently do not result in greater learning gains. Nonetheless, our findings may have practical value: students may have come to expect the problem selection features they know from games. Our study shows they can be added to a tutor (though with the caveat noted below) with relatively low implementation cost while maintaining the tutor's effectiveness.

An interesting finding was that the students who could re-practice completed problems and received rewards performed significantly worse than their counterparts who could re-practice problems but did not receive rewards. The same difference was not found between the two conditions that could not re-practice. To the best of our knowledge, this is a novel finding: we are not aware of studies showing a detrimental effect of re-practice in (tutored) problem solving. A possible explanation is that the urge to earn more stars pushed the students to re-practice, yet re-practicing previously-seen problems is not an optimal strategy for learning as compared to practicing new problems. (In standard ITSs, it is common practice that students practice new problems targeting the same skills, instead of re-practicing problems they have completed before.) Further data analysis supports this explanation: there were significantly more re-starts of problems in the RePr+Rwd condition and there was a significant negative

correlation between the re-start ratio and students' post-test scores. This finding affirms that performance-based rewards can influence students' study choices but it also highlights the need to ensure that students are guided in making optimal choices. Although the combination of re-practicing with performance-based rewards is a very common design pattern in games, its implementation in tutors should be handled with care. For example, instead of giving rewards for individual problems, one could consider adding to the tutor data visualizations that help students analyze and summarize their performance, and provide rewards on an aggregated level. Also, instead of allowing students to re-practice problems they have seen before, the system might afford them freedom to select remedial *new* problems to earn more rewards.

Lastly, the experiment illustrated that an ITS can help students learn more effectively than a commercial educational game, even one with high popular acclaim. The students in the tutor conditions had greater learning gains than students who worked with *DragonBox,* in spite of the fact that the *DragonBox* students solved, on average, four times as many problems. In fact, our results indicate that *DragonBox* is ineffective in helping students acquire skills in solving algebra equations, as measured by a typical test of equation solving. This test is a fair test of *DragonBox'* effectiveness; on average, the students who worked with *DragonBox* reached Level 7 in the game, and thus covered the necessary algebraic rules to solve the equations on this test. Although *DragonBox* was more engaging than the tutor, where it falls short may be in using a concrete context to hide equations during much of the game, without a clear connection to standard algebraic notation and transformation rules. To be fair, *WeWantToKnow*, the company that markets *DragonBox* has recognized the need for supplemental instruction outside of the game and provides a document that teachers can use to help transfer. It is not known how effective this additional instruction is. It is not that there is no learning in *DragonBox* - there is plenty of it, as evidenced by students' progression through the game levels. However, the learning that happens in the game does not transfer out of the game, at least not to the standard equation solving format. Much of the publicity surrounding *DragonBox* seems to have focused on progression through the game levels as an indicator of learning, perhaps because this measure is so readily observable. This, in our opinion, is a profound mistake. What matters is not within-game learning, but out-of-game transfer of learning, and the two cannot be equated. We hope that our study will contribute to more careful consideration in the popular media of out-of-game transfer of learning as a key criterion when judging the educational value of games. Incidentally, our study should not be interpreted as questioning the educational potential of games in general, just that of one game in particular. We see educational games and gamification of ITSs as promising approaches to developing effective and enjoyable advanced learning technologies.

In sum, our study represents progress in our understanding of the value of gamification in ITSs. We demonstrated ways of gamifying shared problem control in an ITS with no detrimental effects, though we would have liked to see gains at minimum in enjoyment and preferably also in learning. Further, we discovered that the combination of performance-based rewards and the freedom of re-practicing, both common game design patterns, is detrimental for learning when imported into an ITS. The comparison between the tutors and *DragonBox* affirms that an intelligent tutor can be highly effective in helping students learn. It illustrates also that an educational game can foster high enjoyment and gain great popularity without helping students learn.

We continue to see great potential for incorporating gamification features in ITSs to enhance students' learning and engagement, although as our study illustrates importing popular game design patterns into ITSs needs to be done with care. There may be no substitute for careful evaluation studies.

Acknowledgements. We thank Jonathan Sewall, Octav Popescu, Martin van Velsen, Kristen Chon, Gail Kusbit and Kate Souza for their kind help with this work. We also thank the participating teachers and students. This work is funded by an NSF grant to the Pittsburgh Science of Learning Center (NSF Award SBE083612).

References

1. Andersen, E., Gulwani, S., Popovic, Z.: A Trace-based Framework for Analyzing and Synthesizing Educational Progressions. In: Proc. of the SIGCHI Conference on Human Factors in Comp. Systems, pp. 773–782. ACM, New York (2013)
2. Atkinson, R.C.: Optimizing the Learning of a Second-Language Vocabulary. Journal of Experimental Psychology 96, 124–129 (1972)
3. Boyce, A., Barnes, T.: BeadLoom Game: Using Game Elements to Increase Motivation and Learning. In: Proc. of Foundations of Digital Games, FDG 2010, pp. 25–31 (2010)
4. Corbalan, G., Kester, L., Van Merriënboer, J.J.G.: Selecting Learning Tasks: Effects of Adaptation and Shared Control on Efficiency and Task Involvement. Contemporary Educational Psychology 33(4), 733–756 (2008)
5. Easterday, M.W., Aleven, V., Scheines, R., Carver, S.M.: Using Tutors to Improve Educational Games. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS, vol. 6738, pp. 63–71. Springer, Heidelberg (2011)
6. Grolnick, W.S., Ryan, R.M.: Autonomy in Children's Learning: An Experimental and Individual Difference Investigation. Journal of Personality and Social Psychology 52, 890–898 (1987)
7. Jackson, G., McNamara, D.: Motivation and Performance in a Game- based Intelligent Tutoring System. Journal of Educational Psychology 105(4), 1036–1049 (2013)
8. Liu, J.: DragonBox: Algebra Beats Angry Birds. Wired (2012), http://www.wired.com/geekdad/2012/06/dragonbox/all/
9. Long, Y., Aleven, V.: Supporting Students' Self-Regulated Learning with an Open Learner Model in a Linear Equation Tutor. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 219–228. Springer, Heidelberg (2013)
10. Meluso, A., Zheng, M., Spires, H.A., Lester, J.: Enhancing 5th Graders' Science Content Knowledge and Self-Efficacy through Game-based Learning. Computers and Education 59(2), 497–504 (2012)
11. Paas, F.G.W.C., Van Merriënboer, J.J.G.: Variability of Worked Examples and Transfer of Geometrical Problem-Solving Skills: A Cognitive-Load Approach. Journal of Educational Psychology 86(1), 122–133 (1994)
12. Shapiro, J.: It Only Takes about 42 Minutes To Learn Algebra with Video Games. Forbes (2013), http://www.forbes.com/sites/jordanshapiro/2013/07/01/it-only-takes-about-42-minutes-to-learn-algebra-with-video-games/
13 Where Is the Math in DragonBox 12+?, http://wewanttoknow.com/resources/DragonBox/Math_In_DragonBox.pdf

Replay Penalties in Cognitive Games

Matthew W. Easterday and I. Yelee Jo

School of Education and Social Policy, Northwestern University, Evanston IL USA

Abstract. Replay penalties that punish players by making them repeat progress are ubiquitous in video games yet noticeably absent from tutors, creating a dilemma for designers seeking to combine games and tutors to maximize interest and learning. On the one hand, replay penalties can be frustrating and waste instructional time, on the other, they may increase excitement and prevent gaming the system. This study tested the effects of replay penalties on learning and interest. In a randomized, controlled experiment with a two-group, between subjects design, 100 University students played two versions of Policy World, an educational game for teaching policy argument, with and without penalties that forced students to replay parts of the game. Results showed that replay penalties decreased learning and interest. These findings suggest a minimize penalties principle for designing cognitive games.

Keywords: intelligent tutoring, educational games, serious games, penalties.

1 Introduction

Can *cognitive games*—educational games with embedded intelligent tutoring, promote learning as effectively as tutors [1] and be as fun to play as games? Cognitive games may not be able to maximize both learning and fun—by attempting both, they might achieve neither. In this study, we examine the effect of *penalties* on learning and interest to develop empirically supported principles for designing cognitive games.

How do we design cognitive games? Unfortunately, we cannot simply add tutors to stand-alone games—tutors and games are designed differently and for different goals. As a result, designers are forced to choose which game-like and tutor-like features to use, some of which are compatible, some of which are not.

Some of these differences *are* compatible. For example, tutors often lack fantasy environments. In most tutors, a learner is more likely to find himself solving a textbook problem than battling aliens. But we can easily design a cognitive game with both a fantasy environment and intelligent tutoring. Recent studies on game-like elements in tutors have focused on compatible features that do not directly affect tutoring, like 3D graphics [2] or narrative, visual presentation, and rewards [3].

Other differences between tutors and games are *incompatible*. Tutors provide more assistance than games, and they make it easy for the learner to figure out what to do by giving scaffolding and feedback on each *step*. Imagine the first-person shooter *Halo* giving the same level of assistance: not only would it tell you whether you've hit or been hit by an enemy, it would tell you what kind of weapon to choose, which

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 388–397, 2014.

enemy to target, how to point the weapon, when to shoot, the enemy's weakness, and so on. Whereas players make their own game guides and walkthroughs for entertainment games, tutors provide these answers for free via hints. Tutors also minimize penalties—after incorrect steps, tutors often allow learners to try again. Imagine *Halo* with minimal penalties: being hit wouldn't reduce your health; after missing an enemy, the alien would patiently wait for you try again. These conflicting approaches to assistance and penalties means that it is unclear whether cognitive games can simply add tutors to normal games to maximize learning and fun—adding tutors may increase learning at the expense of fun.

Here we are interested in penalties that directly affect tutoring, specifically replay penalties, where the game punishes players by making them restart at an earlier point. Replay penalties are ubiquitous across a wide variety of single-player video games such as Angry Birds, Halo, and Tetris. Replay penalties are ubiquitous because they make single-player games fun—losing lives or progress after a mistake creates pressure to make the right choice—which increases the excitement of making the choice and the satisfaction of choosing correctly.

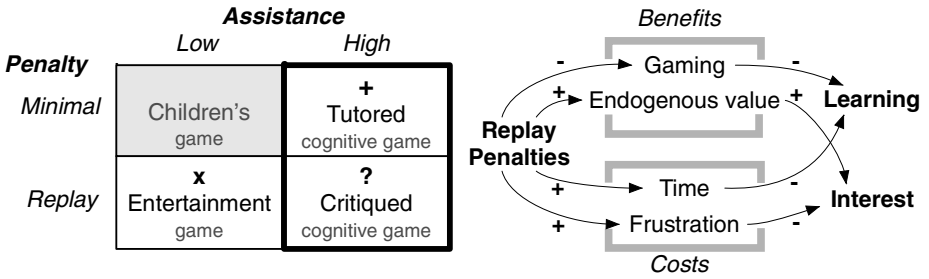


Fig. 1. Cognitive game design types (left) and possible causal effects penalties (right)

To explore the design space at the intersection of tutors and games, Easterday, Aleven, Scheines & Carver [4] compared two games: a *tutored* cognitive game with high-assistance and minimal penalties and an *entertainment* game with low-assistance and replay-penalties (Figure 1). Intuitively, we might predict a tradeoff with the tutored game better for learning and the entertainment game better for interest. In fact, the tutored game led to greater learning and competence, which in turn increased interest. So if entertainment game conventions are not effective, feedback promotes learning after all, how might a *critiqued game* with replay penalties and high feedback fare? In this study, we examine the role of replay penalties in cognitive games.

The case for replay penalties. Penalties are "rewards in reverse," such as points, resources and time that are taken away for making a mistake [5, p. 192, 6, p. 94]. Game designers consider penalties essential because they create the challenge and meaning needed to generate excitement. First, penalties *create challenge* by removing a resource needed to achieve a game goal, such as removing one of the player's limited number attempts or lives, forcing the player to replay part of the game, or reducing the player's points (needed to achieve a high score). Designers use penalties to make an

easy game more challenging to prevent boredom. Second, penalties create *endogenous value* [5, pp. 31-33] or *meaningful play* [7, pp. 353-355] by establishing the relationship between the players' actions and game outcomes—penalties and rewards communicate to the player whether her actions move her closer to, or further from the goals of the game. Third, the combination of *challenge* and *endogenous value* are necessary for generating the interest/excitement/pleasure the player experiences when overcoming a challenge to reach a meaningful goal [5, p. 192], [7, p. 346].

Penalties might also increase learning by decreasing gaming. Intelligent tutors are susceptible to the *gaming the system* phenomenon, when learners "attempt[] to succeed in an interactive learning environment by exploiting properties of the system rather than by learning the material" [8]. For example, when hints give the learner the correct answer after a given number of requests, learners often rapidly click the hint request button until they receive the answer, rather than think about the problem. Penalties that impose a cost to random guessing or hint abuse might prompt students to think about the problem.

The case against penalties. On the other hand, penalties might decrease learning by wasting instructional time. Easterday et al. [4] found that an intelligent tutor embedded in a game-like environment increased learning and interest compared to a version that provided less feedback and stronger penalties, as is more typical of games, although this game-like tutor also provided less assistance, so the causal effect of penalties was unclear.

Second, replay penalties might not be necessary for creating interest. Entertainment games designed for children such as *Lego Star Wars* are immensely popular and impose extremely minimal penalties: when a player *dies* in Lego Star Wars, he drops all his *pieces* (points and money) but immediately reappears on the screen and is given several seconds to pick up the dropped pieces. While children's games have penalties that do not affect tutoring such as losing points, they suggest that *replay* penalties may not be necessary for generating challenge and interest.

Hypotheses. In this study, we compared how two cognitive games with either *replay* or *minimal penalties* affected learning and interest. The *replay penalty* version required students to replay parts of the game after an error, while the *minimal penalty* version allowed immediate error correction. The outcome measures were *learning*, which measured the policy analysis skills taught by the game, and *interest*, as measured by the Intrinsic Motivation Inventory [9]. Assuming that penalties make games more challenging, there are several plausible hypotheses:

1. *Null*: Replay penalties have minor, floor, or ceiling effects on learning and interest.
2. *Reduced gaming*: Replay penalties increase learning by reducing gaming (caused by low levels of interest), but have little effect on low levels of interest.
3. *Tutored game*: Replay penalties decrease learning and interest, because they waste instructional time and are unnecessary for generating interest.
4. *Critiqued game*: Replay penalties increase interest by making the game more challenging and, at best, equal learning by providing identical assistance.
5. *Painful game*: Penalties decrease interest by making the game too challenging.

We predicted support for either the *null* or *painful game hypothesis* based on the motivational importance game designers place on penalties and our previous finding that a *minimal penalties* version of Policy World increased learning and aspects of

interest more than "game-like" version with minimal feedback and penalties [4]– possibly suggesting that lack of feedback in the game-like version decreased learning and masked the motivational effects of penalties.

2 Policy World

Policy World [4, 10] is a cognitive game designed to teach policy argumentation [11, 12]. In Policy World (Figure 2), the learner plays a policy analyst who must defend the public against the handsome but unscrupulous corporate lobbyist *Mr. Harding* by persuading the *Senator* to adopt policies based on evidence on topics such as carbon emissions, national health care and childhood obesity. The story employs an empowerment theme in which the young policy analyst, after typically failing an initial job interview (a disguised pre-test), is recognized as having great potential by *Ms. Cynthia Stark*, the head of a policy think-tank. The learner is guided through a grueling training by two mentor characters: *Molly*, another young but more senior analyst, and a sharp-tongued virtual *Tutor* that teaches the learner to analyze policies. At the end of the game, the player is tested in "real" senate hearings (posttests) in which the player must debate two policies with Mr. Harding to save the think tank's reputation and defend the public against Mr. Harding's corrupt agenda.

Fig. 2. Policy World screenshots of player and tutor characters

Policy World's fantasy environment follows *anime adventure/visual-novel* genre conventions that use dialogue boxes and hand drawn images of characters representing the speaker against backgrounds that display the character's location. The fantasy environment is heavily based on the game *Phoenix Wright* where the player stars as a defense attorney who "...must collect evidence, weed through inconsistent testimonies, and overcome corrupt agendas to ensure that justice prevails" [13], and which is one of Capcom's top-10 best-selling series [14]. Learners routinely comment positively on the similarities between the games.

Most Policy World levels include three broad activities: searching for policy information, analyzing that information, and debating policy recommendations against a computer opponent. During search, learners use a fake Google interface to find 3-7 newspaper-like reports, typically 3-5 paragraphs in length, containing causal claims from various sources like the New York Times, scientific journals, and bloggers that have varying levels of credibility and evidential support. At any time during search,

learners can select a report to analyze, which requires them to comprehend, evaluate, diagram, and synthesize the evidence about the causal claims in the report using causal diagramming tools. Once learners have completed searching for evidence and constructing their causal diagrammatic analysis, they proceed to the final debate phase. During debate, learners make a policy recommendation, explain how the policy will affect a desired outcome, and provide evidence for their position by citing reports. The computer opponent (either Molly or Mr. Harding depending on the level) will argue against the player, attacking his recommendations, mechanism and evidence by providing alternate recommendations mechanism and evidence.

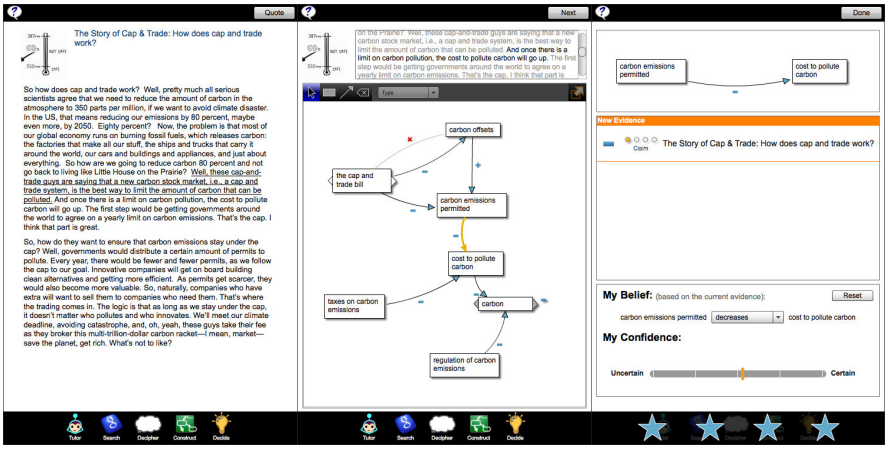


Fig. 3. Policy World comprehension, diagramming and synthesis screens

In this study, we focus on the analysis skills: comprehension, evaluation, diagramming and synthesis (Figure 3) described in [4] and repeated here for coherence:

- *Comprehend*. After selecting a report to analyze, the learner attempts to highlight a causal claim in the text such as: "the Monitoring the Future survey shows that 21 minimum drinking age laws decrease underage consumption of alcohol."
- *Evaluate*. The learner then uses combo boxes to identify the evidence type (experiment, observational study, case, or claim) and strength of the causal claim. Strength is rated on a 10-point scale labeled: none, weakest, weak, decent, strong, and strongest. The evaluation was considered correct if: (a) the evidence type is correctly specified, and (b) the strength rating roughly observes the following order taught during training: experiments > observational studies > cases > claims.
- *Diagram*. The learner next constructs a diagrammatic representation of the causal claim using boxes to represent variables and arrows to represent an increasing, decreasing, or negligible causal relationship between the two variables. The learner also "links" the causal claim in the report to the new diagram arrow which allows him to reference that report during the debate by clicking on that arrow.
- *Synthesize*. The learner then synthesizes his overall belief about the causal relationship between the two variables based on all the evidence linked to the arrows between those variables up to that point. The synthesis step requires the learner to

specify which causal relationship between the two variables is best supported by the evidence, and his confidence in that relationship on a 100 point slider from uncertain to certain. During training, a synthesis attempt is considered valid if: (a) the learner moves his belief in the direction of the evidence, assuming the learner's description of the evidence was correct, and (b) the learner's belief mirrors the overall evidence, assuming the learner's description of the evidence was correct.

Assistance. During training, errors in analysis are flagged by animated red stars and an explanation for the error. Errors in debate are also flagged and followed by Socratic questions that walk the learner through the steps involved in reading the diagram produced by analysis and citing evidence linked to the diagram.

3 Method

Design. The study used a two-group, between subjects, randomized, controlled, experimental design that compares a *replay penalties* version with a *minimal penalties* version of the game. During training, the *replay* penalties version of Policy World erased learners' progress upon making a mistake. When the learners made errors on an analysis step for a particular causal claim, they were sent back to the first analysis step. When learners received 5 debate strikes, they had to replay the level. The minimal penalties version allowed learners to correct errors with no loss of progress.

Participants. 100 university students were recruited through campus flyers and email. Students were compensated $16 for completing the study and an additional $4 for beating posttest 1 and an additional $4 for beating posttest 2.

Procedure. Students first took a pretest on either the drinking age (5 causal claims) or obesity (7 causal claims). During the pretest, students were allowed to search and analyze as many or as few reports as they liked before continuing to the debate. Students were then randomly assigned to the *replay* or *minimal penalties* training. Each group completed 3 training problems on video game violence (4 causal claims), organic foods (5 causal claims), and vaccines (4 causal claims). During training, replay penalties students received penalties for errors while minimal penalties students did not. Since it was possible that replay penalties students might take much longer on training, they were allowed to advance to the test levels after 1 hour on the training levels. After training, students completed the intrinsic motivation inventory survey [9] with sub-scales measuring perceived competence, effort, pressure, choice, value and interest. Finally students played two test levels without replay penalties or tutoring. The debate test (on cap-and-trade, with 8 causal claims) was a debate-only level that provided a completed diagram (to test hypotheses about debate skills outside the scope of this paper). Students then took a posttest identical to, and counterbalanced with, the pretest.

4 Results

Analysis 1: Do replay penalties affect learning? To examine how penalties affect learning we examined students' pre/post test analysis skill across the minimal/replay penalties groups using a two-way, repeated measures (mixed) ANOVA. Both groups

improved on all four skills. The minimal penalty group showed significantly greater improvement than the replay penalty group on comprehension, evaluation and diagramming and a (not significantly) greater improvement on synthesis, (Table 1-2).

Table 1. Both groups learned analysis but the minimal penalties group learned more

		Pretest		Posttest	
Analysis skill	**Penalties**	**M**	**SD**	**M**	**SD**
Comprehend	*Replay*	2.68	1.92	3.50	1.79
	Minimal	2.24	1.82	4.26	1.64
Evaluate	*Replay*	1.72	1.58	2.38	1.59
	Minimal	1.68	1.63	3.10	1.72
Diagram	*Replay*	2.26	1.77	3.36	1.79
	Minimal	1.94	1.68	4.08	1.68
Synthesize	*Replay*	2.76	2.19	4.00	2.06
	Minimal	2.66	2.50	4.60	2.34

Table 2. The ANOVA showed a significant increase on all analysis skills for both groups and a greater increase on 3 out of 4 skills for the minimal penalties group

	Test (pre/post)				Penalty				Test-penalty interaction			
	df	**F**	**p**	**GES**	**df**	**F**	**p**	**GES**	**df**	**F**	**p**	**GES**
Comprehend	1 98	53.4	7.5E-11 *	0.138	1 98	0.28	0.60	0.002	1 98	9.53	2.6E-03 *	0.028
Evaluation	1 98	36.4	2.9E-08 *	0.094	1 98	1.51	0.22	0.011	1 98	4.86	3.0E-02 *	0.014
Diagram	1 98	70.9	3.2E-13 *	0.183	1 98	0.48	0.49	0.003	1 98	7.31	8.1E-03 *	0.022
Synthesize	1 98	39.2	9.8E-09 *	0.110								

Analysis 2: Do penalties affect intrinsic motivation? To examine how penalties affect interest we asked students to complete the well-validated intrinsic motivation inventory [9], immediately after the three training levels and analyzed the results with pair-wise t tests. The minimal penalties group felt significantly more competent, found the game more interesting and more valuable for learning policy (Table 3).

Table 3. Replay penalties decreased perceived interest, competence and value

	Replay		Minimal						
	M	**SD**	**M**	**SD**	**t**	**df**	**p**	**ll**	**ul**
Interest	3.44	1.32	3.93	1.24	1.89	97.62	0.061 .	-0.02	0.99
Effort	4.83	1.06	4.83	1.09	-0.02	97.88	0.985	-0.43	0.42
Choice	3.41	0.82	3.50	0.87	0.57	97.59	0.567	-0.24	0.43
Competence	3.45	1.43	4.17	1.20	2.71	94.91	0.008 **	0.19	1.24
Pressure	3.74	1.64	3.74	1.06	0.01	84.13	0.988	-0.54	0.55
Value	3.88	1.56	4.41	1.34	1.80	95.91	0.075 .	-0.05	1.10

Analysis 3: How are penalties, learning and interest related? To better understand the causal relationships between penalties, training, interest and analysis, we constructed a path model using the GES algorithm implemented in Tetrad 4 [15, 16] which searched for equivalence classes of unconfounded causal models consistent

with the correlations in Table 4 and our prior knowledge that: (a) penalties were determined before any other factor, (b) training was completed next, (c) intrinsic motivation was measured next, and (d) the posttest was completed last. Figure 4 shows the model discovered by Tetrad that we consider highly plausible and shows an excellent fit to the data. A chi-squared test of the deviance of the path model from the observed values showed we cannot reject this model a significance level of .05, $\chi 2$ (53, n=100)=48.41, p>.65, (here larger p-values indicate better fit and values *above* 0.05 indicate that we can't reject the model at a significance level of .05).

Table 4. Correlations between penalties, training success, analysis skills, and IMI subscales

			Analysis skills				Intrinsic Motivation Inventory							
	Penalty	**Train**	**Compr**	**Evalua**	**Diagra**	**Synth**	**Effort**	**Interest**	**Choice**	**Compt**	**Pres**	**Val**	**M**	**SD**
Penal	1												0.5	0.5
Train	-.50 ***	1											0.7	0.2
Com	-.22 *	.41 ***	1										0.6	0.2
Evalu	-.22 *	.48 ***	.74 ***	1									0.4	0.2
Diagr	-.20 *	.37 ***	.97 ***	.71 ***	1								0.5	0.2
Synth	-.15 ·	.33 ***	.82 ***	.63 ***	.83 ***	1							0.6	0.3
Effor	.00	.09	.04	.07	.02	.02	1						4.8	1.1
Inter	-.19 ·	.43 ***	.43 ***	.40 ***	.38 ***	.35 ***	.41 ***	1					3.7	1.3
Choic	-.06	.15 ·	.06	.17 ·	.05	.12	-.07	.17 ·	1				3.5	0.8
Com	-.26 **	.54 ***	.52 ***	.55 ***	.46 ***	.45 ***	.16 ·	.55 ***	.26 **	1			3.8	1.4
Press	-.00	-.20 *	-.23 *	-.26 **	-.21 *	-.20 *	.07	-.21 *	-.25 **	-.45 ***	1		3.7	1.4
Value	-.18 ·	.37 ***	.45 ***	.40 ***	.40 ***	.43 ***	.35 ***	.78 ***	.16 ·	.54 ***	-.21 *	1	4.1	1.5

*p<.05 **p<.01 ***p<.001

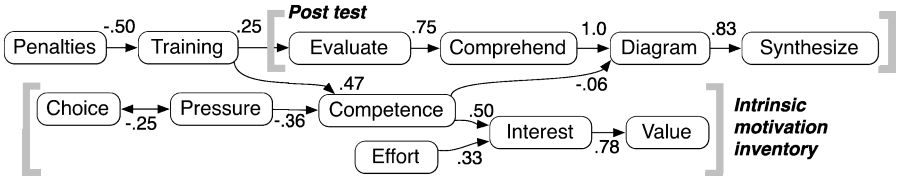


Fig. 4. Penalties decrease learning directly and by reducing perceived competence, which also decreases interest and perceived value of the game

In this model, replay penalties decreased training performance. Training performance affected posttest analysis (by increasing evaluation skills, which increased comprehension, which increased diagramming, which increased synthesis) and also influenced motivation (by increasing perceived competence, which increased interest, which increased value). Motivation in turn affected analysis by increasing diagramming. Choice was correlated with pressure, but it is not clear which caused the other.

Analysis 4: How do penalties affect training time? Students in the replay penalties version took longer to complete the training (M=20.8 min, SD=9.5) than students in the minimal penalties version (M=16.4 min, SD=4.5), t(70)=-2.96, p<.004.

5 Discussion

The results show that replay penalties decrease learning and interest in cognitive games. They do so by decreasing training performance, which directly impacts learning, and by decreasing motivational factors (specifically perceived competence which affects learning and interest and in turn value), which indirectly impact learning.

While these results may contradict our intuitions about the motivational effects of penalties, they are consistent with the effects on learning of previous work on combining tutors and games, which found that greater assistance also increased learning and motivation through similar mechanisms [4]. What is surprising is that game designers seem to so consistently and ubiquitously use a feature that seems to decrease interest across a wide variety of entertaining single-player video games.

The (apparent) contradiction is resolved by appealing to *balance*. Entertainment game designers often use (entertainment) tasks that are cognitively simple and add replay penalties to make them more challenging. Replay penalties don't create excessive frustration because players are likely to succeed if they keep trying. Educational game designers often begin with learning tasks that are cognitively complex and add assistance to make them easier. Replay penalties here make a complex task *too* frustrating. Of course, education game designers could use less assistance and easier, more gradated problems, but this would lengthen learning time.

Our intuitions about the motivational effects of games may be misleading because they are biased by our experience of players who have *voluntarily selected* to play a given game. Furthermore, entertainment games are not designed to promote learning that transfers out of the game, so there is no reason to think that cognitive games will succeed by mimicking their conventions. Entertainment games are designed to create the *illusion* of competence in a fake world, not actual competence in the real world [17].

Contribution: the *minimize penalties principle*. Thus the contribution of this work is support for a *minimize penalties principle*—that cognitive games should reduce replay penalties to increase learning and interest. Like the children's game *Lego Star Wars*, it is possible to maintain interest in cognitive games when the only penalty is a halt in progress (the most minimal possible). This leads to a design implication for educational games quite different from entertainment games: if tutoring is provided, it is better to balance a game by providing minimal penalties on a complex problem than replay penalties on a simple problem. This is the best possible result: embedding tutors in game environments increases learning and interest with *no tradeoff*.

If we are to make educational games that are effective for fun and learning, we must take advantage of what we have already learned about intelligent tutoring. While there are many proposed principles for games [5] and educational games [e.g., 18] and even some with empirical support [19], there are none that help designers resolve the conflicts that arise when applying intelligent tutoring techniques to games. Our previous work provided support for adding tutors to games (*tutoring principle*), to which we now add the *minimize penalties* principle. Future work must generalize and expand upon these principles if we are to apply intelligent tutoring research to realize the full potential of educational games.

References

1. VanLehn, K.: The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. Educational Psychologist 46(4), 197–221 (2011)
2. Lane, H.C., Hays, M.J., Auerbach, D., Core, M.G.: Investigating the Relationship Between Presence and Learning in a Serious Game. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 274–284. Springer, Heidelberg (2010)
3. Rai, D., Beck, J.E.: Math Learning Environment with Game-like Elements: An Experimental Framework. International Journal of Game-Based Learning (IJGBL) 2(2), 90–110 (2012)
4. Easterday, M.W., Aleven, V., Scheines, R., Carver, S.M.: Using Tutors to Improve Educational Games. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS (LNAI), vol. 6738, pp. 63–71. Springer, Heidelberg (2011)
5. Schell, J.: The Art of Game Design: A Book of Lenses. Morgan Kaufmann, Burlington (2008)
6. Konzack, L.: Computer Game Criticism: A Method for Computer Game Analysis. In: Mayra, F. (ed.) Proceedings of Computer Games and Digital Cultures Conference, pp. 89–100. Tampere University Press, Tampere (2002)
7. Salen, K., Zimmerman, E.: Rules of Play: Game Design Fundamentals. MIT Press, Cambridge (2004)
8. Baker, R.S.J.D., de Carvalho, A., Raspat, J., Aleven, V., Corbett, A.T., Koedinger, K.R.: Educational Software Features that Encourage and Discourage "Gaming the System". In: Dimitrova, V., Mizoguchi, R., du Boulay, B., Graesser, A. (eds.) Proceedings of the 14th International Conference on Artificial Intelligence in Education: AIED 2009, pp. 475–482. IOS Press, Amsterdam (2009)
9. University of Rochester: Intrinsic Motivation Inventory, IMI (Web page) (1994), retrieved from `http://www.psych.rochester.edu/SDT/measures/IMI_description.php`
10. Easterday, M.W.: Policy World: A Cognitive Game for Teaching Deliberation. In: Pinkwart, N., McLaren, B. (eds.) Educational Technologies for Teaching Argumentation Skills, pp. 225–276. Bentham Science Publishers, Oak Park (2012)
11. Easterday, M.W., Aleven, V., Scheines, R., Carver, S.M.: Constructing Causal Diagrams to Learn Deliberation. International Journal of Artificial Intelligence in Education 19(4), 425–445 (2009)
12. Easterday, M.W., Aleven, V., Scheines, R., Carver, S.M.: Will Google Destroy Western Democracy? Bias in Policy Problem Solving. In: Dimitrova, V., Mizoguchi, R., du Boulay, B., Graesser, A. (eds.) Proceedings of the 14th International Conference on Artificial Intelligence in Education: AIED 2009, pp. 249–256. IOS Press, Amsterdam (2009)
13. Amazon: Phoenix Wright: Ace Attorney: Video Games. Amazon (Web page) (2013), retrieved from `http://www.amazon.com/Phoenix-Wright-Ace-Attorney-Nintendo-DS/dp/B000B69E96`
14. Capcom: Total Sales Units (Web page) (2012), retrieved from `http://www.capcom.co.jp/ir/english/business/salesdata.html`
15. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search, 2nd edn. MIT Press, Cambridge (2000)
16. Tetrad: Tetrad (Computer Software) (2008), retrieved from `http://www.phil.cmu.edu/projects/tetrad/`
17. Thompson, C.: Halo 3: How Microsoft Labs Invented a New Science of Play. Wired Magazine 15(09) (2007)
18. Gee, J.P.: Learning by Design: Games as Learning Machines. E-Learning 2(1), 5–16 (2005)
19. Mayer, R.E.: Multimedia Learning and Games. In: Tobias, S., Fletcher, J.D. (eds.) Computer Games and Instruction, pp. 281–305. Information Age Publishing, Charlotte (2011)

Use of a Cognitive Simulator to Enhance Students' Mental Simulation Activities

Kazuhisa Miwa[1], Jyunya Morita[2], Hitoshi Terai[1], Nana Kanzaki[3], Kazuaki Kojima[4], Ryuichi Nakaike[5], and Hitomi Saito[6]

[1] Graduate School of Information Science, Nagoya University
miwa@is.nagoya-u.ac.jp
[2] Graduate School of Knowledge Science, Jaist
[3] Junior College, Nagoya Woman's University
[4] Learning Technology Laboratory, Teikyo University
[5] Graduate School of Education, Kyoto University
[6] Faculty of Education, Aichi University of Education

Abstract. We developed a cognitive simulator of the dual storage model of the human memory system that simulates the serial position effect of a traditional memory recall experiment. In a cognitive science class, participants learned cognitive information processing while observing the memory processes visualized by the simulator. Through the practice, we confirmed that participants learned to predict experimental results in assumed situations implying that participants successfully constructed a mental model and performed mental simulations while running the mental model in various settings. We discuss the possibility that a cognitive model can be used as a learning tool and, more specifically, as a mediator tool connecting theory and empirical data.

Keywords: mental model, cognitive model, mental simulation, cognitive science class.

1 Introduction

Scientific discovery learning is a big challenge in the ITS studies. The computer simulation method is widely used in learning contexts[6,4]. A theory-and-data correspondence is crucial in scientific discovery processes. This correspondence enhances theory-based thinking such as data interpretation and scientific explanation. In this paper, we investigate students' scientific activities in the psychology domain in contrast to previous studies that conducted their investigated in the natural sciences domains such as physics and chemistry.

Anderson proposed the theory-model-data framework for clarifying functions of computational models in cognitive science[1]. Computational models have taken a central role in the science of the human mind. In psychology, a theory is usually represented as a conceptual model; the semantic network model and the dual storage model of the human memory system are representative examples. In this paper, "theories" refers to such conceptual models. A conceptual

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 398–403, 2014.

model as a theory predicts only abstract and qualitative experimental results. Therefore, it is impossible for a theory to correspond directly with data. Computational models are embodied, as computer programs, from such conceptual models with some psychological assumptions. These models predict specific experimental results and enable researchers to verify the theories that underlie the models based on a direct comparison of results of computer simulation and human experiments. Computational models function as a strong research tool for cognitive scientists. In this paper, we investigate practical use of computational models as learning tools by utilizing their function as a mediator between theory and data. As a mediator for connecting a theory to data, computation models may contribute to the improvement of students' theory-and-data correspondence activities.

From the viewpoint of learning activities, it is important that computational models as externalized computer programs be internalized as mental models to enable students to manipulate models in their minds. Plenty of related literature has emphasized the importance of mental models in science education[3,5]. By constructing mental models, students acquire the capability to accurately predict experimental results expected to be obtained in hypothesized situations. But, as mentioned above, there is a gap between the conceptual theory and data. Therefore, it is usually impossible to predict a data pattern when only a conceptual theory is given.

In this paper, we developed a cognitive simulator that performs cognitive information processing based on the dual storage model. We report on an intelligent tutoring system mounted on the simulator and on a class practice using the tutoring system. We verify that our participants successfully improved their ability to predict experimental results, implying that they constructed more sophisticated mental models of the dual storage model and performed mental simulations while assuming various experimental settings and individual differences of simulated participants.

2 Cognitive Simulator

2.1 Dual Storage Model

The serial position effect is explained based on the dual storage model of the human memory[2]. A main concern of our practice is the distinction between short-term memory and long-term memory. Information from the outside world is temporarily stored in the iconic memory. Information selectively focused in the iconic memory is sent to the short-term memory; however, it is maintained only for about 15 to 30 seconds. Without rehearsals of the items, they are soon erased from short-term memory. Through rehearsal processes, information in the short-term memory is encoded into the long-term memory. Once information is encoded in the long-term memory, it is never forgotten. The primacy effect emerges because only words presented earlier are encoded into long-term memory through rehearsals. The recency effect appears because words from the end remain in the short-term memory and are directly retrieved from it when

asked to be reported. In contrast, words in the middle of the list have been present too long to be held in short-term memory, but not long enough to be encoded to long-term memory.

2.2 Production System Model

Our cognitive simulator was established as a production system model based on the dual storage model.

The model on the server has nine production rules: (1) *A presentation rule* presents an item and encoding it into the short-term memory; (2) *two erasing rules* erase items from the short-term memory after a time limit for holding items has passed, and erasing items from the short-term memory when the number of items has exceeded the working memory capacity; (3) *A rehearsal rule* performs rehearsals of items in the short-term memory; (4) *An encoding rule* encodes items into long-term memory when the number of rehearsals exceeds a threshold value; (5) Two *reporting rules* report items from the short-term and long-term memories when asked to report memorized items after all items have been presented; (6) Two *rules for stopping the system and increasing the time counter.*

Additionally, four parameters control the information processing of the model: (1) *Presentation interval* controls an interval between two successive item presentations; (2) *Rehearsals for encoding* specifies the number of rehearsals needed for encoding items into the long-term memory; (3) *Working memory capacity* specifies the number of items that can be simultaneously stored in the short-term memory. When the number of items exceeds this limit, the oldest item that was stored earliest is erased from the short-term memory; (4) *Holding time* specifies a time limit for holding items in the short-term memory. When no rehearsals are performed beyond the time limit, the item is erased from short-term memory.

This cognitive simulator visualizes which items are stored in the short-term and the long-term memories. Participants learn how the model works while confirming which items are rehearsed in the short-term memory and encoded into the long-term memory or which ones overflowed from the short-term memory.

Figure 1 shows computer simulation results along with results from human experiments. The human experiment data were gathered through class practice that we will report on later. A comparison of the results indicates that the model successfully duplicated the U-shaped pattern of the human experiments data.

3 Class Practice

The class practice was performed in a cognitive science class in the School of Informatics and Sciences of Nagoya University. Fifty-nine non-psychology major undergraduates participated in the class practice. Three class sessions were assigned to this practice. A summary of the sessions flow is as follows.

In the initial stage, the participants received an instructional lecture about the procedures of the memory recall experiment investigated in the practice. After the lecture, a pre-test was performed wherein an experimental sheet was given

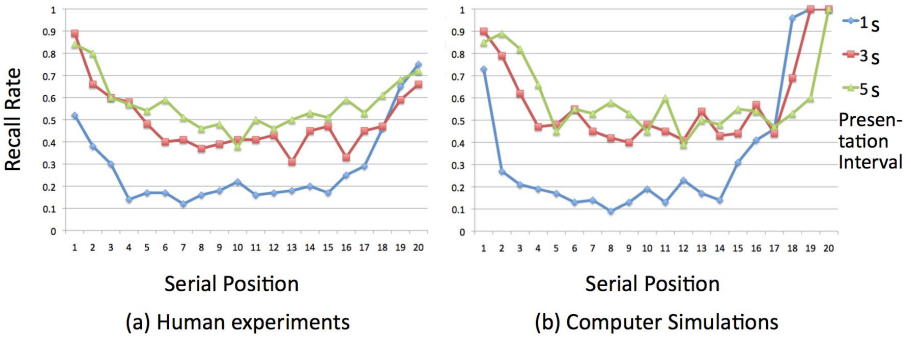


Fig. 1. The left graph shows the results of the human experiments in three settings: the presentation intervals are one, three, and five seconds. The right graph shows the results of the computer simulations also performed in three settings: the presentation intervals are short, medium, and long, corresponding to the instances of one, three, and five seconds, respectively, in the human experiments.

printed with an empty graph. The vertical axis of the graph was the recall rate and the horizontal axis was the serial position of the presented words. Participants were required to predict experimental results; specifically, they were asked to draw three lines on the empty graph corresponding to the experimental results in the three intervals of one, three, and five seconds. An identical experimental sheet was used in the middle and post-tests mentioned later.

After the pre-test, the participants joined the memory recall experiment. They were presented with a series of 20 words at intervals of one, three, and five seconds. Soon after the presentation phase, they were asked to recall the memorized items and write them on a sheet of paper. In each of the three intervals, a total of two experimental sessions were repeated. The recalled words by each participant were gathered via a web-based data collection system and analyzed using a semi-automatic analysis system. The results are shown in Figure 1. The experiment successfully demonstrated primary characteristics of the serial position effect such as the recency effect, the primacy effect, and the decrease of recall of the middle terms.

The second class session was conducted a week after the first one. In this session, the dual storage model was conceptually explained to the participants by an instructor. They were taught the fundamental functions of the components of the model such as short and long term memories and were instructed on how the model works. After the session ended, they were again required to predict experimental results with the intervals at one, three, and five seconds on another identical experimental sheet as used in the pre-test. These results were treated as a middle test.

In the third session, the participants learned how the model processes information while using our cognitive simulator. Specifically, they observed the memory process while confirming the way each presented item is stored in the short-term

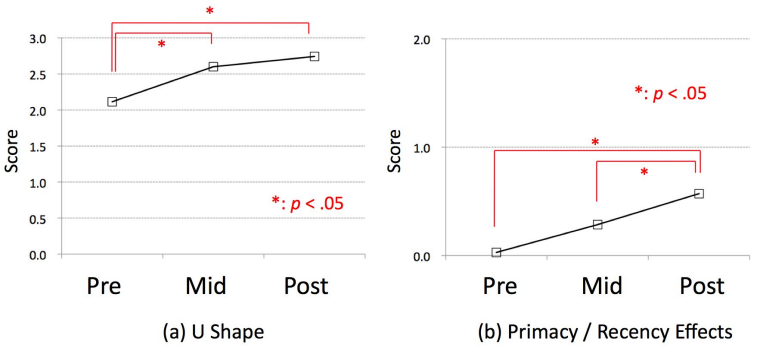


Fig. 2. Results of subjective analysis. The left graph shows the average scores from the viewpoint of the U-shaped pattern. The right graph shows the average scores from the viewpoint of the recency and primacy effects. ANOVAs revealed that the main effect of the test factor reached significance in both the U shape and the primacy/recency effects ($F(2, 68)=4.48$, $p<.05$; $F(2,68)=9.66$, $p<.01$). The results for the multiple comparisons using Ryan's method are presented in the figure.

memory, the way it overflows and is erased from the memory, or is encoded into t he long-term memory after rehearsal processes. In the simulations, the interval of word presentation was set at three seconds. After the learning phase using the cognitive simulator, they were again required to predict experimental results on another identical experimental sheet. These results were treated as a post-test. The participants observed the memory process only when the presentation interval was three seconds; therefore, they had to predict the recall performance at one and five seconds by inferring the memory process while performing mental simulations.

4 Results

We focused on whether they drew U-shape patterns. Second, we confirmed whether the recency and primacy effects were represented. For the review, we used the following criteria.

As seen in Figure 1, for all results in the three interval cases, the performance lines in the human experiments were U-shaped. We reviewed the graphs depicted by the participants and counted the number of lines represented as U-shaped patterns. A score from zero to three was assigned to each experimental sheet.

As seen in Figure 1, the impact of the primacy effect on the performance lines depended on the presentation intervals, whereas the impact of the recency effect did not. When this point is represented on the graph, one point is counted. Additionally, for the primacy effect, when the interval time was one second, the recall rate of initially presented items greatly decreased. When this point is represented, one point is added. Based on these criteria, a score from zero to two was assigned.

Figure 2 shows the average scores in the pre-, middle-, and post-tests. An improvement of the value from the pre- to middle test indicates the effects of conceptual explanations by an instructor. The improvement from the pre- to post-test indicates effects of learning experiences using the cognitive simulator along with the conceptual explanations. As for the analysis of the U-shaped pattern, the post-test score was greater than the pre-test score. However, there was also an improvement from the pre- to middle-test, but no difference between the middle- and post-tests. This indicates that a conceptual lecture was effective enough for the participants to predict that the recall rate in the middle position greatly decreased. On the contrary, for analysis of the recency and primacy effects, the score of the post-test was greater than the pre-test, but the score of the middle test was not. There was a significant increase from the middle- to post-test, indicating that learning was improved by the use of the cognitive simulator.

5 Conclusions

Our cognitive simulator that visualizes mental information processing enhances deeper understanding of the human memory system, especially from behavioral and functional points of view. Learning with such a cognitive simulator helps students construct mental models with which they can perform mental simulations. In our class practice, such a function enabled the participants to predict experimental results accurately in various hypothesized situations. These results imply a possibility that computational models can function as mediators between conceptual theories and data in scientific discovery learning in the cognitive science domain. Computational models have been used as a research tool, but our results presented an example class practice in which they can be also used as a learning tool in cognitive psychology education.

References

1. Anderson, J.R.: Rules of the mind. Lawrence Erlbaum Associates, Inc., Publishers (1993)
2. Atkinson, R.C., Shiffrin, R.M.: Human memory: A proposed system and its control processes. In: Spence, K.W., Spence, J.T. (eds.) The Psychology of Learning and Motivation: Advances in Research and Theory, vol. 2, pp. 89–105 (1968)
3. Clement, J.: Model based learning as a key research area for science education. International Journal of Science Education 22(9), 1041–1053 (2000)
4. De Jong, T., van Joolingen, W.R.: Scientific discovery learning with computer simulations of conceptual domains. Review of Educational Research 68, 179–201 (1998)
5. Gilbert, J.: Models and modelling: Routes to more authentic science education. International Journal of Science and Mathematics Education 2, 115–130 (2004)
6. Rutten, N., van Joolingen, W.R., van der Veen, J.T.: The learning effects of computer simulations in science education. Computers & Education 58, 136–153 (2012)

Towards an Ontology for Gamifying Collaborative Learning Scenarios

Geiser Chalco Challco[1], Dilvan Moreira[1], Riichiro Mizoguchi[2], and Seiji Isotani[1]

[1] University of São Paulo, ICMC, São Carlos, SP, Brazil
geiser@usp.br, {dilvan,sisotani}@icmc.usp.br
[2] Japan Institute of Science and Technology, Ishikawa, Japan
mizo@jaist.ac.jp

Abstract. Gamification is an interesting and relatively new concept. The concept of Gamification is more than just game playing; it is about introducing game design elements in a proper way to satisfy individual motivational needs according to personality traits. Researcher and Educators are currently looking at Gamification to deal with the problem of learner engagement and motivation in Collaborative Learning (CL). To address this issue, we have been developing an Ontology for Gamifying CL Scenarios (OntoGaCLeS). In this paper, we present the main ontological structure used to support the personalization of game design elements in CL contexts. To demonstrate its use, we show the personalization of a gamified CL scenario through a case study.

Keywords: gamification; ontology; collaborative learning.

1 Introduction

In the last years, many researchers have contributed to the development of the concept of gamification and its application in education [5, 8]. Deterding and colleagues define gamification as "*the use of **game design elements** in non-game contexts*" [3]. It aims to increase people's engagement and motivation through the application of game mechanics, such as point system, social connections and so on, in a situation that has other purposes than its normally expected (i.e. for entertainment). The educational benefits that a learner gets through the use of gamification depend strongly on how well game elements are connected with pedagogical approaches [8].

To support a proper design of Collaborative Learning (CL) scenarios that use game design elements, referred to as *gamified CL scenarios*, our approach has developed semantic web tools that assist the design of CL scenarios based on the principles of learning theories, instructional design and game design. In this context, this paper will describe the development of an ontology that organizes the knowledge related to CL scenarios and game elements. This ontology is called **OntoGaCLeS** - *an Ontology for Gamify Collaborative Learning Scenarios*. It has been developed using the Hozo Ontology editor [9], and it is available at *http://labcaed.no-ip.info:8003/ontogacles*.

In the following sections, we define the concepts of this ontology used to gamify a CL scenario and demonstrate how to assign proper player roles and game mechanics using through a case study.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 404–409, 2014.

2 Related Works

Despite the growing number of studies and applications of gamification in the field of education [8], there is not any ontology that enables humans and computers to find, share, and combine information related to CL scenarios and game design elements. Our work is one of the first to define player roles as a fundamental concept that can be used for the personalization of game mechanics in CL scenarios. In the literature, there are many gamification frameworks [4, 5, 6, 10, 12, 14] that can be applied in different contexts and scenarios. For the learning environments, Domínguez et al. [5] and Simões et al. [12] propose gamification frameworks that help instructional designers select proper game mechanics based in learners' individual traits. These frameworks were developed employing the relationship between game mechanics and human desired, where each game mechanics satisfies a set of human desires

3 Gamifying a Collaborative Learning Scenario

A gamified CL scenario is a CL scenario in which game design elements are applied to make the learning experience more enjoyable and meaningful. In a gamified CL scenario, the learning experience itself intends to be so enjoyable that learners will do the proposed activities even at great cost, because they are highly motivated, particularly because of the use of different game mechanics (e.g. leaderboards, point system, social connection, etc.). As motivation is the process used to allocate energy and to maximize the satisfaction of needs [11], a circular flow of "needs, behavior and satisfaction" is set in a CL scenario to gamify it, where to fulfill the learner's motivational needs, a learner must be engaged in behaviors that will lead to the satisfaction of those needs using game mechanics. In many cases the combination of different game mechanics provide the adequate environment to satisfy a person's motivational needs, called human desires by Domínguez et al. [5] and Simões et al. [12]. Thus, to support the personalization of these game mechanics in CL scenarios, our current formalization of a gamified CL scenario introduces concepts and terms shown in Figure 1, where:

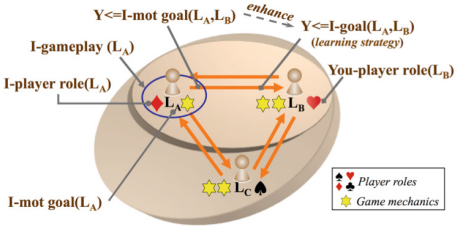


Fig. 1. Concepts and terms defined in a gamified CL scenario

I-mot goal is the individual motivation goal of person in focus (*I*). Since motivation is circular, at the end of a CL scenario, the needs of a person may change or intensify, and the level of motivation (called *current motivational stage*) will be increased. Thus, individual motivation goals will be used to represent needs that must be satisfied and motivational stage that will be achieved.

Y<=I-mot goal is the motivational strategy that represents a set of guidelines used to attain the individual motivational goals (*I-mot goal*). Vassileva [13] argues that

users can be viewed as agents who act to maximize their utility (payoff) in a world where certain behaviors have payoffs. Thus, to make people behave in particular way, the motivational strategies enhance the learning strategy (*Y<=I-goal*) in the CL scenario through the creation of proper systems of rewards (incentives).

I-player role and **You-player role** are the player roles that will be played by the person in focus (*I*) and (*You*).

I-gameplay is the gameplay strategy employed by the person in focus (*I*). The gameplay contains the definition of *game mechanics* that will be used and the behavior of how the person (*I*) and game mechanics interact during the run-time. Furthermore, the gameplay (*I-gameplay*) also is used to define the rational arrangement among player roles, motivational strategies and game mechanics.

Figure 2 (a) shows the ontological structure developed in this work to represent a gamified CL scenario. It extends Isotani et al. [7; 15] adding the *motivational strategy* and the *gameplay strategy*. The motivational strategy (*Y<=I-mot goal*) is defined as follows: the learner (*I*) who uses the strategy is the person in focus and plays the player role (*I-player role*); the other learner (*You*) who is interacting with the learner (*I*) that plays the player role (*You-player role*); and the benefits of using the strategy is represented as individual motivation goals (*I-mot goal*). The gameplay strategy (*I-gameplay*) defines: the proper game mechanics (*what use*) that can be used by learner (*I*), and each game mechanics includes a set of game dynamics (*gameplay*) in terms of game rewards (*rewards*) that will be used during the execution of the scenario.

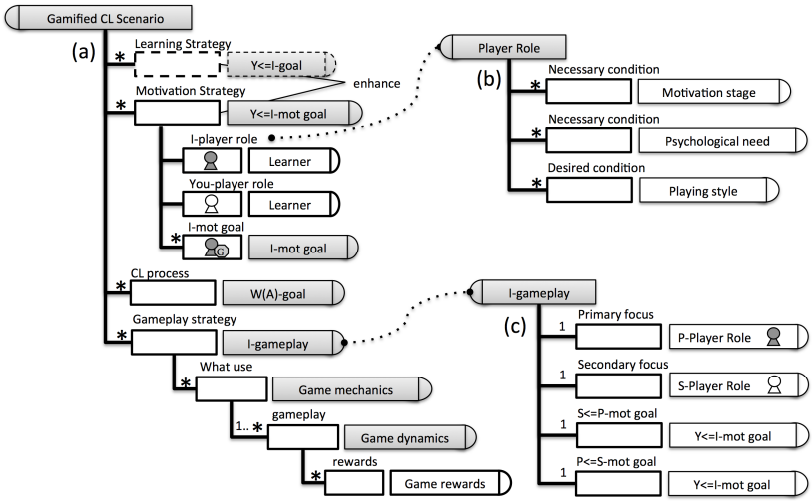


Fig. 2. Ontological structure used to define a gamified CL scenario

The Figure 2 (b) shows the ontological structure to represent the concept of player roles, where there are two types of prerequisites referred as necessary and desired conditions. The *necessary conditions* are essential for a learner to play a role and are defined as current *motivational stages* and *psychological needs*, while the *desired conditions* are those that a learner should satisfy to obtain the full benefits of playing a player role and are defined as individual traits that define a playing style.

In Figure 2 (c), the ontological structure is used to define the concept of the concept of a gameplay strategy as the rational arrangement among player roles and motivational strategies and game mechanics. This arrangement has the purpose of representing how different player roles have the potential to affect each other in a gamified CL scenario. Thus, the primary focus is a learner *P* that plays the primary player role (*P-Player role*), the secondary focus is a learner *S* that plays the secondary player role (*S-Player role*), and *S<=P-mot goal* with *P<=S-mot* goal are used to define the motivational strategies used by learner *P* to interact with learner *S*.

4 Case Study

To demonstrate the utility of our approach in the personalization of game mechanics for CL scenarios, we define eight gamified CL scenarios employing the ontological structure defined in previous section and the information shown in Table 1 and 2.

Table 1. Player roles for the case study

Player role	Necessary and desired condition		
	Psych. need	Motivation stage	Playing style (ind. trait)
networker	relatedness		interacting-orientation, users-orientation
socializer		intrinsic motivate	
exploiter	autonomy		interacting-orientation, system-orientation
free-spirit		intrinsic motivate	
consumer	mastery		acting-orientation, system-orientation
achiever		intrinsic motivate	
self-seeker	purpose		acting-orientation, users-orientation
philanthropist		intrinsic motivate	

Table 2. Gamified CL scenarios (motivation and gameplay) for the case study

Motivation strategy		Gameplay strategy	
I-Player role	Motivational goal (*I-mot goal*)	S-Player Role	Game mechanics (*what use*)
networker	satisfaction of relatedness, internalize motivation		social status, point system, and badges system
socializer	satisfaction of relatedness	socializer	social status, and social connections
exploiter	satisfaction of autonomy, internalize motivation		point system, virtual goods system, and badges system
free-spirit	satisfaction of autonomy,		unlockable system, and customization tool
consumer	satisfaction of mastery, internalize motivation		virtual goods systems
achiever	satisfaction of mastery		quests system, point system, and exclusive system
self-seeker	satisfaction of purpose, internalize motivation		leaderboard, badges system, and exclusive system.
philanthropist	satisfaction of purpose		gifting system

In our modeling, each gamified CL scenario is related with only one player role so that a set of game mechanics defined in the Table 2 will be used to satisfy the psychological motivational needs, and to internalize motivation. For example, the gamified CL scenario for consumer will be used to satisfy of mastery need and to internalize motivation because player role "*consumer*" (shown in Table 1) has as necessary conditions "mastery" and current stage of motivation "*intrinsic motivate*." The gameplay strategy for this example relates the player role "*consumer*" with game mechanics "*virtual goods system*." In our current version of our ontology, we only define one restriction for *socializer* who can only work with other *socializer*, this restriction is defined as in the gameplay strategy as *S-Player Role*.

To select proper game mechanics in a CL scenario using our developed ontology, we propose to use the next procedural steps:

1. Match the individual motivational goal for each learner by looking the *I-mot goal* in all gamified CL scenario. The result usually has more than one scenario that can help to internalize motivation and to satisfy basic needs. For example, suppose that the *I-mot goal* of a learner with identification *l*, who wants to satisfy his psychological motivational needs of mastery and autonomy, match with gamified CL scenarios for: *exploiter*, *free-spirit*, *consumer*, and *achiever*. The current level of motivation for learner *l* is "*intrinsic motivate,*" and he has personal traits of "*acting-orientation*" (preference for unilateral action) and "*system-orientation*" (preference for exploring the system).
2. Check if learners have the necessary conditions to play game roles for the CL scenarios obtained in step (1). For learner *l*, the game role *free-spirit* and *achiever* satisfy the necessary conditions because his current level of motivation is *intrinsic motivate*, and he wants to satisfy his need of *autonomy* or *mastery*.
3. Set the game roles obtained in the step (2) for each learner according priorities calculated using the desired conditions that are satisfied. Learners with all satisfied conditions have high-priority, and learners with only necessary conditions have low-priority. For learner *l*, the player role *achiever* has high-priority because he has personal traits of *acting-orientation* and *system-orientation*; thus, the role *achiever* is attribute for learner *l*.
4. Finally, we set the gameplay for learner. This task is completed through selection of proper game mechanics in gameplay (*I-gameplay*). For learner *l*, the gameplay for achiever enables to select*: quests system*, *point system*, and *exclusive system point.*

5 Conclusions and Future Research

In this paper, we presented an ontological structure that enables to represent gamified CL Scenarios. This structure allows the personalization of game mechanics through the rational arrangement between *motivational strategies* and *player roles*. To demonstrate this personalization, in the case study, we performed the organization of the knowledge related to eight scenarios that allows the selection of proper game mechanics for each learner. Next, we presented a set of procedural steps that can be used together with our modeled scenario to select proper game mechanics.

We believe that the results of this work are the first steps forward for creating new type of intelligent collaborative tools that provide assistance for development of more engaging and motivating CL scenarios. In the current version of our ontology, we did not define the game dynamics that personalize the systems of rewards in each game mechanics. Thus, our next steps will consider how this game element must be formalized according our ontology. Furthermore, it is also important to identify what is the association of game mechanics with CL interaction patterns defined in [7; 15]. Future research will also consider the inclusion of optimal flow theory [1].

Acknowledgements. We thank CNPq and CAPES for supporting this research.

References

1. Czikszentmihalyi, M.: Flow: The psychology of optimal experience. Harper (1990)
2. Deci, E.L., Ryan, R.M.: The "what" and "why" of goal pursuits: Human needs and the self-determination of behavior. Psychological Inquiry 11(4), 227–268 (2000)
3. Deterding, S., Sicart, M., Nacke, L., O'Hara, K., Dixon, D.: Gamification. Using Game-design Elements in Non-gaming Contexts. In: CHI 2011 Extended Abstracts on Human Factors in Computing Systems, pp. 2425–2428. ACM, New York (2011)
4. Dignan, A.: Game frame: Using games as a strategy for success. Simon and Schuster (2011)
5. Domínguez, A., Saenz-de Navarrete, J., Marcos, L., Fernández-Sanz, L., Pagés, C., Martínez-Herráiz, J.: Gamifying learning experiences: Practical implications and outcomes. Computers & Education 63, 380–392 (2013)
6. Duggan, K., Shoup, K.: Business Gamification for Dummies. John Wiley & Sons (2013)
7. Isotani, S., Inaba, A., Ikeda, M., Mizoguchi, M.: An ontology engineering approach to the realization of theory-driven group formation. International Journal of Computer-Supported Collaborative Learning 4(4), 445–478 (2009)
8. Kapp, K.M.: The gamification of learning and instruction: game-based methods and strategies for training and education. John Wiley & Sons (2012)
9. Kozaki, K., Kitamura, Y., Ikeda, M., Mizoguchi, R.: Hozo: An environment for building/using ontologies based on a fundamental consideration of role and relationship. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 213–218. Springer, Heidelberg (2002)
10. Nicholson, S.: A user-centered theoretical framework for meaningful gamification. Games+ Learning+ Society 8 (2012)
11. Pritchard, R., Ashwood, E.: Managing motivation: A manager's guide to diagnosing and improving motivation. CRC Press (2008)
12. Simões, J., Redondo, R.D., Vilas, A.F.: A social gamification framework for a K-6 learning platform. Computers in Human Behavior 29, 345–353 (2013)
13. Vassileva, J.: Motivating participation in social computing applications: A user modeling perspective. User Modeling and User-Adapted Interaction 22(1-2), 177–201 (2012)
14. Werbach, K., Hunter, D.: For the win: How game thinking can revolutionize your business. Wharton Digital Press (2012)
15. Isotani, S., Mizoguchi, R., Isotani, S., Capeli, O.M., Isotani, N., De Albuquerque, A.R.P.L., Bittencourt, I.I., Jaques, P.: A Semantic Web-based authoring tool to facilitate the planning of collaborative learning scenarios compliant with learning theories. Computers and Education 63, 267–284 (2013)

Serious Games Go Informal: A Museum-Centric Perspective on Intelligent Game-Based Learning

Jonathan P. Rowe*, Eleni V. Lobene, Bradford W. Mott, and James C. Lester

Department of Computer Science, North Carolina State University, Raleigh, NC 27695
{jprowe,eavagias,bwmott,lester}@ncsu.edu

Abstract. Intelligent game-based learning environments show considerable promise for creating effective and engaging learning experiences that are tailored to individuals. To date, much of the research on intelligent game-based learning environments has focused on formal education settings and training. However, intelligent game-based learning environments also offer significant potential for informal education settings, such as museums and science centers. In this paper, we describe FUTURE WORLDS, a prototype game-based learning environment for collaborative explorations of sustainability in science museums. We report findings from a study investigating the influence of individual differences on learning and engagement in FUTURE WORLDS. Results indicate that learners showed significant gains in sustainability knowledge as well as high levels of engagement. Boys were observed to actively engage with FUTURE WORLDS for significantly longer than girls, and young children engaged with the exhibit longer than older children. These findings support the promise of intelligent game-based learning environments that dynamically recognize and adapt to learners' individual differences during museum learning.

Keywords: Intelligent game-based learning environments, informal science education, individual differences, science museums, educational games.

1 Introduction

There is growing evidence suggesting that game-based learning environments are effective educational tools [1]. Intelligent game-based learning environments, which integrate rich, immersive experiences of digital games with adaptive pedagogical functionalities of intelligent tutoring systems, offer considerable promise [2–4]. To date, much of the research on intelligent game-based learning environments has focused on formal education settings [1–3] and training [4]. However, informal education settings, such as museums and science centers, stand poised to benefit as much, or perhaps even more so, from advances in intelligent tutoring and game-based learning technologies [5, 6]. While the goals of formal education and informal education settings overlap, informal science education places particular emphasis on affective

* Corresponding Author.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 410–415, 2014.

and attitudinal outcomes, which have significant implications for the design of intelligent game-based learning environments [6].

In this paper, we investigate how learners' individual differences impact learning and engagement during game-based learning, as well as how individual differences should influence the design of museum-centric intelligent game-based learning environments. To investigate these questions we have developed FUTURE WORLDS, a prototype game-based learning exhibit for sustainability education in science museums. We report findings from a museum-based study that suggest FUTURE WORLDS is effective at fostering significant gains in sustainability understanding and high levels of engagement. In addition, results indicate that the exhibit elicits more extended durations of engagement among boys and young children than girls and older children. Based on these findings, we argue that intelligent game-based learning environments in museums should incorporate automated detector models for recognizing learners' individual differences, as well as pedagogical planners that tailor problem scenarios based on these characteristics.

2 Future Worlds

FUTURE WORLDS is a prototype game-based learning environment about environmental sustainability designed for children ages 9–12 [6]. The exhibit integrates game-based learning environments, intelligent tutoring systems, and interactive tabletop displays to enable collaborative explorations of environmental sustainability. Learners solve sustainability-centered problem scenarios by investigating alternate environmental decisions in a 3D simulated environment (Fig. 1). The effects of learners' decisions are realized in real-time through vibrant 3D graphics, and they are accompanied by narrated explanations from a robot-like animated pedagogical agent.

The prototype exhibit consists of two integrated displays: a horizontally oriented Samsung SUR40 interactive tabletop, and a vertically oriented 50" high-definition television. Visitors congregate around the interactive tabletop to explore the simulation through multi-touch interactions. The vertical display provides additional space for explanations of sustainability, which are accessible to learners standing farther away from the exhibit. FUTURE WORLDS' 3D environments and sustainability simulation are built with the Unity game engine.

Building on sustainability curricula, such as *Facing the Future's Global Sustainability Resources*, the FUTURE WORLDS curriculum focuses on three integrated themes of sustainability: water, food, and energy. Visitors' objective during learning interactions with the FUTURE WORLDS exhibit is to use the interactive tabletop display

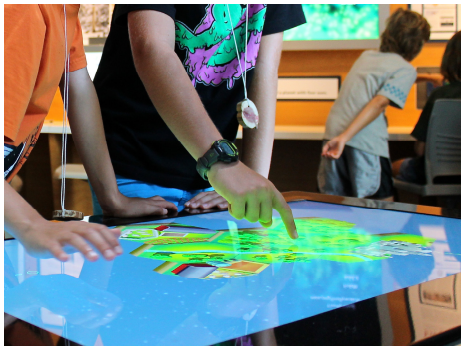


Fig. 1. Future WORLDS exhibit

to collaboratively (or individually) reconfigure an unsustainable virtual environment into a sustainable environment. The first sustainability-centered problem scenario focuses on a virtual watershed that is plagued by several examples of unsustainable farming practices. Learners explore potential solutions to the environment's sustainability challenges, such as incorporating renewable energy sources into the region's electricity portfolio, implementing organic farming practices, and instituting novel waste-to-energy technologies. As learners manipulate the environment, their choices are visualized in real-time through the environment's 3D models, textures, and animations, and the changes visibly propagate across the simulated watershed. The prototype exhibit's focus on interactivity and exploration, as well as its real-time visual feedback, are designed to foster cause-and-effect reasoning and systems thinking.

3 Museum Study

To investigate learning and engagement with a game-based learning environment in an informal setting, a study was conducted with 43 summer campers, ranging in age from 8–14, at a science museum. Participants were grouped into separate cohorts divided across three sessions (N = 14, 16, 13). Pre-test measures were administered on the first day of each cohort's week-long summer camp. The pre-test consisted of a demographic survey and three complementary measures of sustainability understanding: a personal meaning map, a sustainability identification task, and a sustainability image-sorting task.

Personal meaning maps (PMMs) consisted of a blank piece of paper with a brief set of instructions and a prompt phrase: sustainability. Participants used a pen to write or draw words, phrases, and pictures about their conceptualizations of the prompt phrase. After the learning experience, participants could revise their PMMs with a different colored pen. After the study, two raters scored each PMM based on the relevance and accuracy of each element on the page. Inter-rater reliability for the pre-test (r = .84) and post-test (r = .88) PMMs achieved high levels.

For the identification task, learners inspected an illustrated picture of an environment—depicting both sustainable and unsustainable environmental practices—and annotated the picture by circling "good" practices and crossing out "bad" practices. Participants revised their annotations during the post-test. Two raters scored the annotations using a rubric vetted by subject matter experts, and achieved high agreement on both the pre-test (r = .97) and post-test (r = .95).

For the image-sorting task, learners were given copies of ten images depicting various environmental practices (e.g., solar panels, traffic congestion). Participants organized the images into two categories of their choosing, with the goal or choosing two categories containing as similar a number of images as possible. An expert-based categorization of "sustainable" and "unsustainable" was considered the gold standard response, and this benchmark was used to grade responses.

Later in the study, several days after the pre-test, participants were given the opportunity to explore various parts of the museum, including an area where the FUTURE WORLDS exhibit was located. During these study sessions, all participants entered the

study room at the same time and were allowed to spend up to 40 minutes in the area. Learners could leave the space at anytime and were free to explore as they saw fit. However, once they left the area, learners were not permitted to return.

Including FUTURE WORLDS, there were 13 exhibits in the study area. No other museum visitors had access to the room during the study. A nearby human docent was available to resolve technical issues and answer questions. Verbal interactions between the docent and participants were otherwise kept to a minimum. Eleven of the exhibits in the study room were permanent exhibits at the museum. In addition to FUTURE WORLDS, one temporary exhibit was added to serve as a control. This temporary exhibit was the only other exhibit with a human docent, and it consisted of a white board with a sign asking learners to share the most interesting thing they learned in the citizen science area using sticky notes. None of the content in the distractor exhibits overlapped with FUTURE WORLDS.

A post-test was conducted immediately following each participant's exit of the study area, which included the same sustainability measures as the pre-test. All sessions were video recorded. Post-study analyses of the video data were conducted by two coders to determine total dwell time (time spent interacting with the exhibit to any extent) as well as time spent in each of three possible "tiers of proximity" relative to FUTURE WORLDS. Inter-rater reliability was established with a subset of the study data and then the remainder was coded independently (k = .70).

4 Empirical Findings

To investigate the efficacy of the FUTURE WORLDS exhibit, statistical analyses of the pre- and post-test measures, as well as coded video recordings, were conducted. Paired t-tests indicated that learners showed significant gains in PMM score from pre-test (M = 0.8, SD = 1.8) to post-test (M = 1.2, SD = 2.3), $t(37) = 2.5$, $p < .05$. There were also significant increases on the identification task from pre-test (M = 5.96, SD = 2.45) to post-test (M = 6.42, SD = 2.55), $t(37) = 3.28$, $p < .05$, as well as significant gains on the image sorting task from pre-test (M = 7.11, SD = 3.81) to post-test (M = 8.66, SD = 2.67), $t(37) = 2.59$, $p < .05$. Correlation analyses were conducted to investigate whether learners' individual characteristics—including age and gender—showed significant relationships with learning outcomes, but none were observed.

For each learner, total dwell time, as well as time spent in each of three proximity tiers, was determined using video recording data. Across all participants, the average dwell time at FUTURE WORLDS was 12.5 minutes. This is promising, given dwell times typically reported in other informal contexts, such as 5.03 minutes [5] and 4.9 minutes [7]. However, it should be noted that FUTURE WORLDS dwell times were recorded in a semi-controlled study setting, whereas the above cited dwell times were recorded from observations of the general public in non-controlled settings.

A two-tailed independent samples t-test revealed a significant effect of gender on dwell time, where girls (M = 8m:46s, SD = 5m:21s) spent roughly half the time as boys (M = 16m:23s, SD = 10m:33s) engaging with FUTURE WORLDS, $t(35) = 2.86$, $p < .05$. To follow up on this analysis, tier-specific dwell time was examined by gender. Results indicated that males (M = 14m:12s, SD = 7m:34s) spent significantly more

time in the first tier than females (M = 8m:56s, SD = 3m:56s), which is the tier of closest proximity to the exhibit, $t(29) = 2.27$, $p < .05$.

Additional findings about the influence of learners' individual characteristics emerged from analyses of engagement based on learner age. Results indicated that younger children spent more time at the FUTURE WORLDS exhibit than older children, $t(44) = 3.52$, $p < .01$. In fact, children under age 10 spent twice as much time (M = 20m:50s, SD = 12m:33s) as children age 10 and older (M = 9m:56s, SD = 6m:45s).

5 Discussion

Results suggest that learners' sustainability understanding improves from interactions with FUTURE WORLDS. Furthermore, evidence of extended dwell time suggests that learners are highly engaged with the exhibit. In combination, these two sets of findings suggest that learner engagement with FUTURE WORLDS is not superficial; learners are actively engaged for prolonged periods at sufficient depth to yield significant learning gains across three distinct measures of sustainability knowledge. Also, FUTURE WORLDS appears to be equally effective for boys and girls, as well as young and older children, in terms of fostering learning.

Our observation that gender and age have significant effects on dwell time point toward engagement-centric design implications for future iterations of FUTURE WORLDS and intelligent game-based learning environments in general. Regarding gender, several possible mediating factors could explain why girls spent less time with FUTURE WORLDS than boys, such as video game interest or affinity for the game's visual aesthetic style. Additional studies are necessary to isolate what design factors are responsible for the observed gender differences, and how elements of FUTURE WORLDS' design should be augmented to deliberately appeal to females to an extent that is comparable to males. Regarding age, results suggest that future iterations of FUTURE WORLDS should incorporate problem-solving scenarios that span a broader range of content and complexity. It is plausible that the implemented problem scenarios in the FUTURE WORLDS prototype were sufficiently challenging for young children but were not difficult enough for older children, and thus did not sustain their engagement for extended periods. Incorporating intelligent tutoring system capabilities to dynamically adapt the difficulty of problem scenarios to individual learners, or groups of learners, is a promising way to match scenarios' content complexity to the capabilities of diverse learners. We hypothesize that this form of adaptive pedagogical functionality will increase motivation and dwell time. However, adaptive pedagogical planning will require models for automatically detecting learners' individual characteristics as they approach and use exhibits. Administering lengthy pre-tests is not a viable design choice for most informal settings; automated detector models show more promise for diagnosing learner characteristics to inform adaptive pedagogical functionalities.

Notably, we did not find a relationship between dwell time and learning. We expect that adding curricular material beyond the prototype's current proof-of-concept scope—creating opportunities for more variance in content exposure—could reveal relationships between dwell time and learning that have not thus far been observed.

6 Conclusions and Future Work

Intelligent game-based learning environments show considerable promise for informal settings such as museums and science centers. Creating effective intelligent game-based learning environments for museums requires a grounded understanding of how learners engage with, and learn from, game-based exhibits. In this paper, we described an empirical study that found FUTURE WORLDS yields significant gains in sustainability knowledge, as well as high levels of engagement, during museum learning. Furthermore, boys were observed to actively engage with FUTURE WORLDS for significantly longer durations than girls, and young children engaged with the exhibit for longer periods than did older children. These individual differences underscore the importance of future work investigating adaptive pedagogical functionalities, as can be provided by intelligent tutoring techniques, to dynamically tailor game-based learning experiences based on gender and age, as well as automated detectors for diagnosing learners' individual and group characteristics.

Acknowledgements. This research was supported by the National Science Foundation under Grant DRL-1114655. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Clark, D., Tanner-Smith, E., Killingsworth, S., Bellamy, S.: Digital Games for Learning: A Systematic Review and Meta-Analysis (Executive Summary). SRI International, Menlo Park (2013)
2. Jackson, G.T., McNamara, D.S.: Motivation and Performance in a Game-based Intelligent Tutoring System. J. Educ. Psychol. 105, 1036–1049 (2013)
3. Shute, V.J., Ventura, M., Kim, Y.J.: Assessment and Learning of Qualitative Physics in Newton's Playground. J. Educ. Res. 106, 423–430 (2013)
4. Johnson, W.L.: Serious Use of a Serious Game for Language Learning. Int. J. Artif. Intell. Educ. 20, 175–195 (2010)
5. Lane, H.C., Cahill, C., Foutz, S., Auerbach, D., Noren, D., Lussenhop, C., Swartout, W.: The Effects of a Pedagogical Agent for Informal Science Education on Learner Behaviors and Self-efficacy. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 309–318. Springer, Heidelberg (2013)
6. Rowe, J.P., Lobene, E.V., Mott, B.W., Lester, J.C.: Play in the Museum: Designing Game-Based Learning Environments for Informal Education Settings. In: Proceedings of the 9th International Conference on Foundations of Digital Games (2014)
7. Horn, M.S., Solovey, E.T., Crouser, R.J., Jacob, R.J.K.: Comparing the use of tangible and graphical programming languages for informal science education. In: SIGCHI Conference on Human Factors in Computing Systems, pp. 975–984. ACM Press, New York (2009)

Animated Presentation of Pictorial and Concept Map Media in Biology

Whitney L. Cade, Jaclyn K. Maass, Patrick Hays, and Andrew M. Olney

University of Memphis, Department of Psychology
Institute for Intelligent Systems
Memphis, USA
{wlcade,jkmaass,dphays,aolney}@memphis.edu

Abstract. Intelligent tutoring systems are beginning to include more varied forms of media, but little is known about how to choose the appropriate media and whether or not it should be animated. This study used a 2 (animated/static) x 2 (picture/concept map) factorial design in order to evaluate the effect of animation and media type on conceptual knowledge, relational knowledge, and free recall. Learners on Mechanical Turk (N = 208) were exposed to one of four conditions in which they viewed a modified Khan Academy video on cell parts. We found that animation induced higher learning gains when it comes to relational knowledge. For conceptual knowledge, animated concept maps outperformed animated pictures while static pictures produced slightly more learning than static concept maps. Our results indicate that using animations to slowly build complexity in visual displays is particularly important when the displays have a rich structure as in concept maps.

Keywords: picture, concept map, animated media, static display, Khan Academy, Biology, link, node.

1 Introduction

As intelligent tutoring systems (ITSs) become more and more sophisticated, the types of media that can be included in such systems have become increasingly varied. In order to support the students' learning, ITSs have included static images (e.g. AutoTutor [1]), diagrams (e.g. Andes [2]), animated illustrations (e.g. Guru [3]), concept maps (e.g. Guru [3], Betty's Brain [4]), videos (e.g. Operation Aries! [5]), and other media. However, there are at this time very few rules in place to guide which media type to select and how to present it given a myriad of parameters such as the student's prior knowledge, student's spatial ability, and task demands [6]. More work is needed to understand what types of media work best under certain conditions.

Recently, the tension between static and animated images has been of particular interest. The literature on animated images demonstrates a strong division between results, where animations sometimes contribute significantly to students' learning and, at other times, they have no impact on learning whatsoever. For instance, in the document explanation literature, images animated using a technique called "sequential display"

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 416–425, 2014.

(where an image starts out blank and segments of an image appear when they become relevant in the narration) often result in better memory for the information on the image [7]. In a recent meta-analysis, animations were shown to have a $d = .37$ advantage over static images when it comes to learning [8]. However, the authors caution that this effect is not as strong under all conditions (for instance, animations had a weaker effect in Biology than they did in Chemistry), and may in fact disappear in some circumstances (such as when the animations are purely decorational). For instance, [9] found that students who viewed an ordered series of static images outperformed those who viewed an animated visual of the same dynamic process, which is one of the conditions under which animations are meant to operate best. Therefore, it seems that additional investigations must be done to discover the strengths and limitations of image animation.

However, it is not only animated pictures and illustrations that have been investigated for their efficacy. Researchers focused on concept maps, an educational device that is growing in popularity and has been incorporated into multiple ITSs, have also examined how animation can add to student learning. One of the limitations of concept maps is that they often contain no cues to guide specifically how they should be read. Eye tracking research bears this out, as gaze patterns vary largely between participants examining a concept map [10]. Therefore, animations are seen as a method of directing student attention and imposing a specified processing order. There have been two substantial investigations into concept map animation, but the results of these studies have been mixed, indicating that there may be conditions and best practice rules that guide the animation of concept maps as well. [11] found that animated concept maps resulted in better recall of the information 48 hours later over static maps or even animated text, but that animation had no effect on the ability to recall lower-level details. Recently, [12] also compared static and dynamic text and concept maps but found that animation provided no advantage for either text or concept maps. These opposing results may be due to at least one of two key differences in the experimental designs of the aforementioned studies: concept map complexity/size, where [12]'s map was more complex than [11]'s, and the use of accompanied narration, which [12] claimed counteracted the effects of animation in their study by providing too much scaffolded guidance.

While there seems to be indications that both animated concept maps and pictures can be advantageous to learning under the right conditions, very little is known about how they compare to each other. It seems intuitive to suppose that both have their own time and place in educational multimedia environments, but there are currently no rules to guide the selection of one over the other for ITS designers, and further still, there is no research to suggest whether the presence or absence of animation for either of these media forms should inform this selection decision. Currently, both the concept map animation literature and the picture animation literature focus primarily on how each media type stacks up to its own static version, as well as how it compares to and/or works alongside text (e.g. [11], [13]). How concept maps and pictures compare to each other in terms of learning, as well as how animation affects this comparison, is still an open question.

It may also be the case that it is not a simple matter of determining which media type is most effective, but rather, which type aids specific kinds of learning. For instance, one of the strengths of concept maps is that they explicitly model the relationships between concepts, which have been theoretically linked to creativity, understanding, and deep knowledge of the material [14,15]. However, both pictures and concept maps can convey conceptual knowledge, or information pertaining to the topic's main concepts, such as through picture labels or labeled nodes. To date, none of the concept mapping literature has tried to differentiate between these different knowledge types; therefore, little is known about how concept maps, especially animated concept maps, may influence memory for these kinds of information. Picture animation research has revealed that animation can have an effect on memory for different types of knowledge. [8] found that animation had the largest effect on procedural motor knowledge, followed by declarative knowledge. Others have found that the method chosen for animation, such as displaying objects that are thematically related versus spatially related, can deeply impact how the information is later recalled [16]. It may be instructive to investigate how images, animated or not, impact conceptual and relational knowledge as well, as this would allow for a direct comparison between the performance of students exposed to either concept maps or pictures.

Likewise, there also remains an open question as to how narration impacts animated concept maps. Narration is the preferred mode of information delivery when pictures, animated or not, are available, so that the student's attention is not split between the text and the picture [17]. Narration presented with animated images is also not uncommon (e.g. [13]). However, questions have been raised about whether narration washes out the effects of animation in concept maps [12]. Narration may therefore be one parameter for deciding whether or not to use an animated image or concept map, but a replication of this "washing out" should be observed before deeply exploring this parameter.

In this study, we will look at how pictures and concept maps, both animated and static, effect students' relational and conceptual knowledge learning in Biology, as well as their free recall of information. This will allow for a direct comparison between pictures and concept map media types in terms of their learning efficacy, which may help guide selection principles for their inclusion in educational multimedia environments. The visual in every condition will also be accompanied by spoken narration in order to further test [12]'s hypothesis that spoken narration removes the animation effect that had been observed by [11]. Although no advantage was found for animation in Biology visuals [8], this domain relies heavily on visual aids, and so discovering the best practices for displaying these visuals is to the advantage of both educators and ITS designers within the field of Biology.

This experiment used a Khan Academy Biology video as the basis for the educational intervention. Khan Academy is a popular online company dedicated to making short, freely available video lectures that students find easy to understand. Khan Academy videos always feature audio narration of a lesson played in synchrony with screen capture of the narrator drawing pictures or working out problems that support the lesson. Therefore, the videos produced by Khan Academy are ideal for this kind of investigation because they are ecologically valid learning videos that natively

feature picture animation and spoken narration. Khan Academy is also at the forefront of online, self-paced education, and features the kind of media which could be in ITSs due to their low production costs. This experiment seeks to use and modify these materials, which already exist in the educational world, in order to compare the learning produced by animated pictures and concept maps.

2 Methods

A 2 x 2, between-subjects experiment was conducted in order to examine the interactive effects of media animation (*animated* vs. *static*) and media type (*picture* vs. *concept map*). Participants were randomly assigned to one of these four conditions.

Participants were recruited through Mechanical Turk (MTurk), an online service offered through Amazon. MTurk allows "requesters" to put up short tasks ("HITs") to be completed by their "workers," who are then paid a small wage for satisfactorily completing the task. Requesters can also place restrictions, called "qualifications," on who can participate in their study. To ensure quality results, participants who wished to participate in the current study had to have previously completed 50 HITs and had to have at least 95% of those HITs approved by the requesters, meaning that they had done an adequate job on the task and had been paid for it. Additionally, participants in this study had to certify that they were above 18 years of age (an MTurk standard), were a native English speaker, were a United States or Canadian citizen (implemented to increase the odds of recruiting native English speakers and enforced via IP checks), had adequate hardware to complete the experiment, and did not have significant hearing impairments. Those who failed to meet these criteria were disqualified from proceeding to the experiment. Participants who completed the study were paid $1.00.

In this experiment, 214 participants completed the study, but six were disqualified due to their failure to meet the participation criteria. The average age of the participants was 35.91, with a minimum age of 18, a maximum of 72, and a median of 32.5. One hundred fourteen of the participants (54.8%) were female. Previous examinations of the Mechanical Turk workers found that workers are, on average, 31 years old, with ages ranging from 18 to 71, and 55% of workers are female [18], making our sample typical of the MTurk population with the exception that workers outside of the United States and Canada were excluded. Studies have shown that the MTurk population appears to function similarly (i.e., produce qualitatively and quantitatively similar results) to university populations and other online populations [19,20,21].

The materials for this study consisted of four edited videos which made up the stimuli, two interchangeable knowledge measures, and a brief demographics survey (portions of which are reported above). The interventions for this study are based on the "Parts of a Cell" video produced by Khan Academy. In Parts of a Cell, the narrator discusses various cellular components while drawing and labeling them on screen. The Parts of the Cell video was selected due to its straightforward nature and its popularity, as it is one of the most highly viewed videos from their Biology series. The original Parts of the Cell video was edited to shorten the overall video length from 21 minutes to 15 and to remove segments of the video where the narrator scrolls away

from the main image to illustrate some point in an aside. This edited video comprised the *animated picture* stimulus. The *animated concept map* stimulus replaced the visual portion of the edited video with an animated concept map. In the concept map version, the nodes correspond to the same labeled and drawn cell parts that appeared in the pictorial version. The concept map is composed of 18 key propositions (facts in node-link-node format) arranged in a hierarchical layout, with much of the arrangement of the map determined by the order in which information is delivered in the narration. In the *animated concept map*, propositions are added to the map generally when the proposition has been stated for the first time. Once added to the map, propositions are not removed, and the map builds in complexity until it reaches its completed state near the end of the lesson. This is the traditional method of animating concept maps [11,12]. The *static* stimuli, both *pictorial* and *concept map*, were created by taking the final, complete version of the cell picture and concept map, respectively, and using that static image as the visual for the entire video while preserving the same audio narration.

While the "smooth drawing" of the picture and the chunked "sequential display" of the concept map are not visually equivalent forms of animation, both represent the ecologically valid and traditional display methods associated with their respective media types; concept maps have long been considered "animated" if displayed one proposition at a time, while pictures lend themselves to being drawn as a form of drawing attention to and elaborating certain areas of the image (as would be seen in, for instance, expert human tutoring [22]). This experiment considers both styles of animation as roughly functionally equivalent, as both are intended to guide the student's attention to specific parts of the media.

The knowledge measures were created by first extracting the propositional facts of the ensuing lesson (e.g. "Vesicles transport proteins"). These propositions were then made into multiple choice questions by removing either the equivalent of a proposition's node (e.g. "Vesicles transport _____") or its linking phrase ("Vesicles _____ proteins."). There were 18 key propositions in the Biology lesson videos, and therefore 18 node and 18 link questions were created for the knowledge measures. The questions were then randomly sorted into Form A and Form B such that each proposition is represented only once per form, resulting in 9 node question and 9 link questions per form. Participants experienced either Form A or Form B as their pretest, and received the opposite test for their posttest (counterbalanced).

To participate, MTurk workers had to first accept the assignment on MTurk, and were then transferred to the actual experiment, which took place in Qualtrics. Once the worker consented to participate and had made the necessary certifications, he or she first took a pretest to assess his/her prior knowledge on cell parts in Biology. After completing the pretest, participants then experienced one of the four conditions (animated picture video, animated concept map video, static picture video, static concept map video). Controls were removed from the video in order to help prevent starting and stopping the lesson, and participants were instructed merely to listen attentively while the video plays without taking notes. Once the video completed, participants performed a free recall task, where they were asked to write down as much information as they could remember from the material they just saw and heard.

After the free recall task, participants took the posttest (the opposite test form from the pretest), and then filled out a brief demographics form. They were then given a password to enter into Mechanical Turk as proof of completion, for which they were then paid.

3 Results

This research seeks to investigate the effects of animation (animated versus static) and media type (picture versus concept map) on various types of learning, specifically conceptual learning, relational learning, and the general free recall of facts. This was accomplished by examining different types of questions: those questions querying the student's memory of node information (conceptual), link information (relational), and their free recall responses. Each of these research questions has been analyzed and considered separately below.

We first investigated how animation and the media type affected "link" questions, which tap into relationship knowledge. The nine multiple choice link questions from both the pre- and posttests were first scored for correctness, and then each participant's proportional learning gains score was calculated. Proportional learning gains, formulated as (Proportionalized Posttest – Proportionalized Pretest) / (1 – Proportionalized Pretest), are a useful learning gains metric because they control for prior knowledge. These were then analyzed using a 2 x 2 between-subjects analysis of variance (ANOVA). While there was not a significant main effect for media type (p = .39) or a significant animation x media type interaction (p = .645), there was a significant main effect for animation, $F(1, 204) = 4.041$, p = .046. We see that, when the media was animated (M = .542, SD = .377), participants scored significantly higher on the link questions than those in the static media conditions (M = .405, SD = .577; d = .281).

The analysis of the node questions was given a similar treatment; the scores from the nine node questions in the pre- and posttests were used to calculate a proportional learning gains score, which was then examined using a 2 x 2 between-subjects ANOVA. There was no significant main effect for animation (p = .741), but there was a marginally significant main effect for the media type, $F(1, 204) = 3.402$, p = .067, where those in the concept map condition (M = .427, SD = .39) scored higher on node questions than those in the picture condition (M = .319, SD = .452; d = .254). However, the results may be best explained by the significant animation x media type interaction, $F(1,204) = 9.021$, p = .003. When the media was animated, those in the concept map condition (M = .501, SD = .282) outperformed those in the picture condition (M = .222, SD = .537) on the conceptual node questions (d = .65). When the image was static, however, those in the picture condition (M = .414, SD = .347) learned more about concepts (nodes) than did those in the concept map condition (M = .347, SD = .468; d = .165).

The free recall was scored automatically by comparing the responses to a list of keywords created from the transcript of the audio narration. One point was awarded for each of the keywords mentioned in the free recall response (although not

for repeated mentions), and a coverage score for each person was then calculated by dividing the number of keywords mentioned by the total number of keywords on the list. This allowed us to examine their memory for technical vocabulary particular to the topic. The coverage scores were then analyzed using a 2 x 2 ANOVA to investigate the impact of animation and media type on the participants' memory for vocabulary. A covariate of the combined pretest scores for both link and node questions was also included in order to control for prior knowledge. There was no main effect for media type (p = .374), but there was a marginally significant main effect for animation, $F(1,202) = 3.524$, p = .062, where those who experienced an animated visual (M = .349, SD = .2) had better coverage of key vocabulary terms than those in the static visual conditions (M = .318, SD = .19; d = .195).

4 Discussion

In order to aid common ITS design decisions, this study sought to examine how animation, combined with picture representations and concept maps, affects memory for different types of information. The interpretation of the results is clearest when separately considering how relationships and concepts are best learned.

When it comes to knowledge of relationships, this experiment provides evidence that animation can contribute significantly to learning gains, indiscriminate of whether the image is a picture or a concept map. It seems that the action of animation, therefore, is better at guiding attention to the relationships between concepts, which included relationships such as part-of relations, properties, typology, and functional connections ("Vesicles – *transport* – proteins"). While this finding is not explicitly supported by the picture animation literature, there are some indications that it is in line with previous work. Animation has been shown to be somewhat effective in supporting declarative knowledge learning (d = .44), which would contain both concept and relationship knowledge, but it is especially effective in teaching procedural motor knowledge (d = 1.06; [8]). While procedural motor knowledge is undoubtedly also a combination of conceptual and relationship knowledge, it is mostly focused on the relational "how to" information. Therefore, it is somewhat expected that animations would aid more in teaching relationship knowledge. For concept maps, however, this is entirely new information; most recently, animation had been found to have no effect on learning [12], and there has not been an investigation on how animation would impact the learning of links or nodes. Therefore, the discovery that animation does in fact support learning with concept maps provides evidence that animated concept maps may need to be more deeply explored to understand the conditions under which they do or do not aid learners. Interestingly, although it seems intuitive that concept maps would be superior at teaching relational knowledge, no such link was found in this study, perhaps partially due to the topic (where many of the relationship are "part-of" relations, which are equivalently conveyed pictorially).

Conceptual learning is a more complex story. When the media is animated, concept maps provide superior support in teaching conceptual knowledge (operationalized by node questions). This is particularly interesting because it is not merely a case of

concept maps explicitly spelling out the concepts while the picture merely represents them pictorially. The image on the picture drawn by the narrator is also labeled, and the labels of the picture and nodes of the concept map share a high overlap (93%, with the remainder being words jotted down on the picture in an aside). Therefore, the concepts are both equally visually represented in verbal form, but the concept map has the added advantage of removing extraneous detail, which may be the key to its success. Although animated pictures have been shown to aid in teaching declarative knowledge [8], which is at least partly conceptual, this study indicates that animated concept maps may be even better for creating gains in conceptual knowledge. For the static media, the picture fared slightly better than the concept map in terms of conceptual learning, although the difference is not great. This may be because, in the absence of animation, the more detailed picture has more unique cues to encode, and so more attention is paid to the labels and concepts. Further investigation is needed to determine if there is a true advantage of static pictures over static concept maps. However, both static conditions produced higher learning gains for concepts than did the animated picture, possibly due to its overwhelming volume of information and action.

The results from the free recall analysis show a more general (albeit slighter) trend, where animation affected participants' recall of technical vocabulary, which included both conceptual and relationship knowledge (e.g. terms such as "cytosol" and "transcribe"). This effect in and of itself is not surprising given that the literature shows that animation tends to improve learning [8], but what is interesting is the lack of effect for media type. Previous analyses of free recall responses in experiments with animated or static concept maps or text have found that concept maps produce better free recalls than text [11,12]. Here, when comparing two image-based media, this effect disappears; it is possible then that animated image-based media may produce more recalled information than text, although additional research would need to be done to make this direct comparison. While the present free recall analysis is not as thorough as those typical of the concept map literature, where free recall responses are hand scored against a list of declarative knowledge statements, the free recall analysis done here does hint that animation may be useful in not just *recognition* of key terms, as may be demonstrated by the multiple choice questions, but in *recall* of information.

The pattern of results from this study implies that, generally speaking, there are conditions under which concept maps or pictures may be the preferred media, with animation being the main parameter considered in the present work. Animation in general seemed to contribute to relationship knowledge, while animated concept maps specifically were most efficacious in instilling conceptual knowledge. Although, if animation is not an option, static pictures were more effective for conceptual knowledge. This underlines two general findings. First, different types of media seem to have their own contexts in which they are most effective in improving learning, and the learning environment and knowledge goals should be addressed in order to decide on the media type. Second, animation can have different effects on different types of media and learning, and further exploration of this little studied effect is in order. There are also some other interesting implications of this study. This study demonstrated that animated concept maps are not redundant with spoken narration, which would lead to a washing out of learning differences, as [12] suggested. While

the parameters under which animation is not useful for concept maps is not yet known, narration does not seem to be one of those parameters. Additionally, it is interesting that these effects were found in the domain of Biology, which was one of the least successful domains in demonstrating differences between animation and static images. It may be the case that other domains would produce a stronger effect.

While this work fills gaps in our current knowledge of animated media, there are some limitations to this study. First, the results of this study do not take into account the effects of domain (in this case, Biology). It may be the case that certain domains or even certain properties of specific lessons are better represented with other types of media or other forms of animation. Likewise, this study also examines very specific kinds of knowledge measures, those that measure conceptual and relational knowledge, but it may be true that for other types of knowledge, such as general declarative knowledge, deep knowledge, or procedural knowledge, the results may vary. It is not the purpose of the present work to claim that one media type is superior to another in general, but rather, to relate that under the established conditions, animation and animated concept maps seem to produce larger learning gains in relational and conceptual knowledge, respectively. This work also does not explore every method of animating an image; there remains a breadth of animation methods in the existing literature to explore using this paradigm.

With the growing use of concept maps and other forms of media in ITSs, it is important that we continue to investigate the conditions under which they can be effective so that informed design decisions can be made. This will allow us to select the most effective media to use in our systems while avoiding investing in unnecessary "bells and whistles" that do not contribute to the student's experience. Future work which explores the limitations and advantages of different types of media in varying degrees of animation are necessary to contribute to the field's development.

References

1. Graesser, A.C., Lu, S., Jackson, G.T., Mitchell, H., Ventura, M., Olney, A., Louwerse, M.M.: AutoTutor: A Tutor with Dialogue in Natural Language. Behav. Res. Meth. Instum. Comput. 36, 180–193 (2004)
2. VanLehn, K., Lynch, C., Schultz, K., Shapiro, J.A., Shelby, R.H., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. IJAIED 15(3), 147–204 (2005)
3. Olney, A.M., D'Mello, S., Person, N., Cade, W., Hays, P., Williams, C., Lehman, B., Graesser, A.: A Computer Tutor that Models Expert Human Tutors. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 256–261. Springer, Heidelberg (2012)
4. Leelawong, K., Biswas, G.: Designing Learning by Teaching Agents: The Betty's Brain System. IJAIED 18(3), 181–208 (2008)
5. Butler, H.A., Forsyth, C., Halpern, D.F., Graesser, A.C., Millis, K.: Secret Agents, Alien Spies, and a Quest to Save the World: Operation ARIES! Engages Students in Scientific Reasoning and Critical Thinking. In: Miller, R.L., Rycek, R.F., Amsel, E., Kowalski, B., Beins, B., Keith, K., Peden, B. (eds.) Programs, Techniques and Opportunities, vol. 1, pp. 286–291. Society for the Teaching of Psychology, Syracuse (2011)

6. Schnotz, W., Cade, W.: Adaptive Multimedia Environments. In: Sottilare, R., Hu, X., Graesser, A. (eds.) Design Recommendations for Adaptive Intelligent Tutoring Systems: Adaptive Instructional Strategies, vol. 2. Army Research Laboratory, Adelphi, MD (in press)
7. Bétrancourt, M., Dillenbourg, P., Montarnal, C.: Computer Technologies in Powerful Learning Environments: The Case of Using Animated and Interactive Graphics for Teaching Financial Concepts. In: De Corte, E., Verschaffel, L., Entwistle, N., van Merrienböer, J. (eds.) Powerful Learning Environment: Unravelling Basic Components and Dimensions, pp. 143–157. Elsevier, Oxford (2003)
8. Höffler, T., Leutner, D.: Instructional Animation Versus Static Pictures: A Meta-analysis. Learn. Instr. 17, 722–738 (2007)
9. Lowe, R., Schnotz, W., Rasch, T.: Aligning Affordances of Graphics with Learning Task Requirements. Appl. Cognitive Psych. 25(3), 452–459 (2010)
10. Nesbit, J., Larios, H., Adesope, O.: How Students Read Concept Maps: A Study of Eye Movements. In: Montgomerie, C., Seale, J. (eds.) EDMEDIA 2007, pp. 3961–3970. AACE, Waynesville (2007)
11. Blankenship, J., Dansereau, D.: The Effect of Animated Node-Link Displays on Information Recall. J. Exp. Educ. 68(4), 293–308 (2000)
12. Adesope, O., Nesbit, J.: Animated and Static Concept Maps Enhance Learning from Spoken Narration. Learn. Instr. 27, 1–10 (2013)
13. Mayer, R., Hegerty, M., Mayer, S., Campbell, J.: When Static Media Promotes Active Learning: Annotated Illustrations Versus Narrated Animations in Multimedia Instruction. J. Exp. Psychol.-Appl. 11(4), 256–265 (2005)
14. Bloom, B.S.: Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. David McKay Co. Inc., New York (1956)
15. Novak, J.D., Cañas, A.J.: The Theory Underlying Concept Maps and How to Construct and Use Them. Technical Report, Florida Institute for Human and Machine Cognition (2008)
16. Bétrancourt, M., Bisseret, A., Faure, A.: Sequential Display of Pictures and its Effect on Mental Representations. In: Rouet, J.F., Levonen, J.J., Biardeau, A. (eds.) Multimedia Learning: Cognitive and Instructional Issues, pp. 112–118. Elsevier Science, Amsterdam (2001)
17. Sweller, J.: Cognitive Load During Problem Solving: Effects on Learning. Cognitive Sci. 12, 257–285 (1988)
18. Ross, J., Irani, I., Silberman, M.S., Zalvidar, A., Tomlinson, B.: Who are the Crowdworkers?: Shifting Demographics in Amazon Mechanical Turk. In: Mynatt, E.D., Schoner, D., Fitzpatrick, G., Hudson, S.E., Edwards, W.K., Rodden, T. (eds.) CHI EA 2010, pp. 2863–2872. ACM, New York (2010)
19. Mason, W., Suri, S.: Conducting Behavioral Research on Amazon's Mechanical Turk. Behav. Res. 44(1), 1–23 (2012)
20. Paolacci, G., Chandler, J., Ipeirotis, P.G.: Running Experiments on Amazon Mechanical Turk. Judgment and Decision Making 5, 411–419 (2010)
21. Suri, S., Watts, D.J.: Cooperation and Contagion in Web-based, Networked Public Goods Experiments. PLoS One 6(3) e16836 (2011)
22. Williams, B., Williams, C., Volgas, N., Yuan, B., Person, N.: Examining the role of gestures in expert tutoring. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 235–244. Springer, Heidelberg (2010)

Multi-methods Approach for Domain-Specific Grounding: An ITS for Connection Making in Chemistry

Martina A. Rau and Amanda L. Evenstone

Department of Educational Psychology, University of Wisconsin – Madison
{marau,alevenstone}@wisc.edu

Abstract. Making connections between graphical representations is integral to learning in science, technology, engineering, and mathematical (STEM) fields. However, students often fail to make these connections spontaneously. ITSs are suitable tools to support connection making. Yet, when designing an ITS for connection making, we need to investigate what learning processes and concepts play a role within the specific domain. We describe a multi-methods approach for grounding ITS design in the specific requirements of the target domain. Specifically, we applied this approach to an ITS for connection making in chemistry. We used a theoretical framework that describes potential target learning processes and conducted two empirical studies – using tests, eye tracking, and interviews – to investigate how these learning processes play out in the chemistry domain. We illustrate how our findings inform the design of a chemistry tutor. Initial pilot study results suggest that the ITS promotes learning processes that are productive in chemistry.

Keywords: Connection making, multiple representations, empirically grounded design, multi-methods approach, chemistry.

1 Introduction

The ability to make connections between graphical representations is integral to learning in science, technology, engineering, and mathematical (STEM) fields [1]. For instance, to learn about chemical bonding, students need to make connections between Lewis structures, ball-and-stick figures, space-filling models, and electrostatic potential maps (EPMs; see Figure 1). Connection making is a difficult task that students often do not engage in spontaneously, even though it is critical to their learning [1-2]. Hence, they need support to make these connections [3]. Recent research indicates that intelligent tutoring systems (ITSs) can be effective in supporting connection

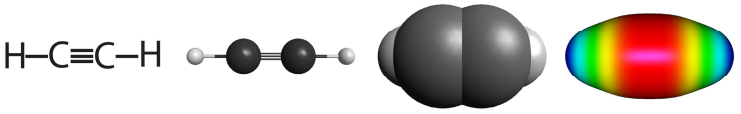


Fig. 1. Graphical representations of ethyne: Lewis, ball-and-stick, space-filling, EPM

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 426–435, 2014.

making [4]. However, to design effective connection making support, we need to investigate which specific learning processes play a role within the target domain. The goal of this paper is to describe a multi-methods approach for grounding the design of an ITS in a particular domain.

Our objective in using this approach is to develop an ITS for connection making that has the potential to significantly enhance students' learning in chemistry. ITS support for connection making is likely to enhance chemistry learning for several reasons. First, the ITS framework of learning through problem solving is in line with the chemistry education literature, which indicates that problem-solving activities can significantly enhance conceptual learning [5], especially when they include graphical representations [6]. Second, even though several educational technologies for chemistry learning exist [7-9], this research is novel because none of them provide explicit and adaptive support for connection making between graphical representations. Finally, the chemistry education literature widely acknowledges that connection making is one of the major stumbling points in chemistry education [10].

In this paper, we describe a new approach to ground the design of this ITS in the chemistry domain. Specifically, we describe how integrating multiple methods provided answers to the following questions: First, which learning processes are important in chemistry and should be supported by the ITS? Second, what problem-solving behaviors should the ITS foster? Third, which chemical bonding concepts should the ITS target? Finally, does the resulting ITS enhance productive learning processes? Even though we address these questions within the chemistry domain, we believe that our approach is a first step towards creating a principled methodology for informing the design of an ITS by the requirements of the specific target domain.

2 Domain-Specific Grounding of Connection-Making Support

2.1 Theoretical Framework

To inform the design of ITS support for connection making, we draw on a theoretical framework, which proposes that two types of connection-making abilities play a role in domain expertise [4]. *Sense-making ability* means that a student can relate aspects that correspond to one another across representations because they depict the same concept (e.g., in the example shown in Fig. 1, relating the local negative charge that results from the triple bond shown in the Lewis structure to the region of high electron density depicted by the red color in the EPM). *Perceptual fluency* is the ability to rapidly and effortlessly find representations that depict the same concept, by relying on perceptual characteristics [11] (e.g., by rapidly seeing that the representations in Fig. 1 show the same molecule based on their linear geometry). The chemistry education literature suggests that both sense-making ability [9, 12] and perceptual fluency in connection making [9-10] are important aspects of chemistry expertise.

We conducted two empirical studies that instantiate this framework for the specific domain of chemistry. Study 1 investigates whether sense-making ability and perceptual fluency are indeed separate connection-making abilities in chemistry. Study 2 investigates the domain-specific aspects of sense-making ability.

2.2 Study 1: Assessment of Sense-Making Ability and Perceptual Fluency

The chemistry education literature documents the importance of both sense-making ability and perceptual fluency in connection making [9]. Confirming the claim that these are indeed distinct abilities is a prerequisite for the design of separate activities to support each of these abilities. To address this question, we conducted an *a priori* factor analysis on an assessment of sense-making ability and perceptual fluency.

Method. Undergraduate and graduate chemistry students with varying levels of expertise were recruited through emails and fliers to take a 30-40 minute online test. 118 students started; 44 students completed the test. We consider resulting missing data to be at random because the item order was at random. The test contained one question about chemistry courses taken, 16 multiple-choice items on sense-making ability (8 on similarities, 8 on differences), and 9 multiple-choice items on perceptual fluency.

Results. We used the SPSS AMOS software to compare three models: a 1-factor model (not distinguishing sense-making ability and perceptual fluency), a 2-factor model (sense-making ability and fluency), and a 3-factor model (sense-making similarities, sense-making differences, and fluency). We excluded missing values (resulting from incomplete tests) on an item-by-item basis. To compare the fit of the tested models, we used root mean squared error (RMSE). The results show that the 3-factor model (RMSE = .072) and the 2-factor model (RMSE = .082) both yielded a better fit than the 1-factor model (RMSE = .088). Because the sense-differences and sense-similarity factors in the 3-factor model correlated highly with $r = .93$, we choose the 2-factor model for further analyses. The resulting two factors, sense-making ability and perceptual fluency, correlate moderately with $r = .62$.

A repeated measures ANOVA showed that students performed significantly better on the sense-making scale ($M = .75$; $SD = .12$) than on the fluency scale ($M = .62$; $SD = .24$; $p < .01$). To investigate the relation of these two abilities with chemistry proficiency, we conducted a regression of the number of chemistry courses taken. The number of courses taken is associated with marginally higher sense-making ability ($\beta = .22$, $p < .10$), and with significantly higher perceptual fluency ($\beta = .448$, $p < .01$).

Discussion. The finding that sense-making ability and perceptual fluency are separate skills in chemistry is in line with the chemistry education literature [9-10, 12] and supports the design of separate activities for these connection-making abilities.

The finding that students have higher sense-making ability than fluency is not surprising: it mimics a current trend in educational practice because most research on connection making focuses solely on sense-making processes [3]. Only recently has perceptual fluency gained attention in the education and psychology literature [11]. Thus, our data encourages the design of an ITS that targets perceptual fluency.

The finding that chemistry proficiency (approximated by the number of courses taken) is a better predictor of perceptual fluency than of sense-making ability is surprising. It seems that chemistry instruction does not sufficiently target the ability to make sense of connections between graphical representations. Given that students'

performance on the sense-making scale is far from perfect (M = .75; SD = .12), there is an instructional need for an intervention that targets students' sense-making ability.

2.3 Study 2: Eye Tracking and Interview Study on Sense-Making Ability

The ability to make sense of the connections between representations involves understanding similarities and differences between different graphical representations. The goal of Study 2 was to investigate the relation between students' ability to identify similarities and differences between representations and their reasoning about domain-relevant concepts. Furthermore, our goal was to identify specific concepts that students struggled with when making connections. Study 2 combined eye-tracking and interview data. This procedure allows us to investigate which visual attention patterns are associated with low and high quality connections.

Method. Twenty-six students participated in Study 2 (21 undergraduate and 5 graduate chemistry students). Sessions took place in the laboratory and lasted 30-45 min. Students were asked to describe similarities and differences between two graphical representations of the same molecule (similar to those in Fig. 1). Students performed this task on an SMI RED250 eye tracker. All verbal responses were audiotaped.

To analyze the eye-tracking data, we created areas of interest (AOIs) for each representation. We considered two measures. First, we considered frequency of switching between AOIs, which is used to indicate perceptual integration [13]. We computed AOI switches as the number of times a fixation on one AOI was followed by another. Second, we considered second-inspection durations. First inspections of an AOI is often considered to indicate initial processing of material that occurs (to a certain extent) automatically [14]. Fixations after the first inspection (i.e., when a student re-inspects an AOI) are considered to reflect intentional processing to integrate the information with other information [14]. We computed the second-inspection durations as the sum of fixation durations that occurred after the initial fixation on a given AOI.

Table 1. First level of the interview coding scheme

Code	Definition (Example)
Surface	Student makes a connection between representations, based on some conceptually irrelevant feature ("um so they're both like red on the top")
Similarities	Student refers to a structural feature of representations that depict the same concept ("the space-filling model and the EPM both in shape are very similar cause they show the electron cloud")
Differences	Student refers to a structural feature of two representations that differs between representations or to information that differs between representations
Inference	Student explains a concept that goes beyond what is depicted ("this [the EPM] just shows that on the oxygen it's more reactive because there's lone

To analyze the interview data, we applied a two-level coding scheme. First-level codes were adapted from prior research on connection making [2]. Specifically,

we distinguished connections based on surface features, similarities, or differences, and whether students made inferences about concepts not explicitly shown in the representations. Table 1 provides descriptions and examples for first-level codes. We constructed the second-level codes bottom up: by collecting concepts that were mentioned during the interview and then coding for their occurrence. Interrater reliability was good with 85% agreement for first-level codes and 72.9% for second-level codes.

Results. First, we analyzed how the eye-tracking data relates to the first-level interview codes (see Table 1). Three participants were excluded from the analysis because the eye-tracking ratio was below 85%. A regression of second-inspection durations on first-level codes showed that longer second-inspection durations were associated with significantly more surface connections ($\beta = .60$, $p < .01$), and marginally more differences ($\beta = .39$, $p = .06$). There was no association of second-inspection durations with similarities. A regression of AOI switches on first-level codes showed that more AOI switches were associated with significantly more surface connections ($\beta = .55$, $p < .01$). There was no association with similarities or differences. In turn, a regression of surface connections, similarities, and differences on inferences showed that difference utterances were associated with significantly more inferences ($\beta = .51$, $p < .01$). There were no associations between similarity or surface utterances and inferences.

Next, we analyzed the second-level interview codes. We identified concepts related to the topics of atom identity (symbol, number of electrons, CPK color coding, general identity information), molecule structure (bond angle, bond length, conformation, geometry, atomic radii, electron cloud), energy (steric interactions, relative energy), electrons (core, valence, shared, lone), atomic structure (shells, orbitals, hybridization potential, spin states), and bonding (type, electronegativity, charge distribution). To get insights into which concepts are particularly difficult for undergraduates, we compared the relative frequency of a concept being discussed by graduate versus undergraduate students. We used differences larger than 1 SD to indicate that undergraduates were unlikely to point out this difference, even though it relates to an important concept. We found that the most difficult concepts for undergraduates were CPK color coding, bond angle, atomic radii, relative energy, bonding type, and reactivity. In addition, undergraduate students were less likely use these concepts to make inferences about the behavior of electrons, atoms, and molecules to explain bonding.

Discussion. Our findings show no clear positive effects of commonly used measures of visual attention. Integrating the eye-tracking data with first-level interview codes allowed us to disambiguate the effects of eye-tracking measures on students' reasoning about domain-relevant concepts. Students who switched more frequently between representations were more likely to focus on surface-level connections. Students with longer second-inspection durations were more likely to notice surface features and differences between representations. Only difference-connections were associated with making more inferences about domain-relevant concepts.

It is surprising that we found no positive associations between noticing similarities between representations and making inferences about chemistry concepts. It may be that expertise in chemistry relies on the use of different graphical representations for

complementary purposes, rather than in using them interchangeably because they provide similar information. Indeed, this interpretation aligns with the literature on how chemistry experts use representations [15]. Consequently, we hypothesize that ITS support for connection making in chemistry should focus on how different graphical representations depict *complementary* information, rather than how they depict *similar* concepts. To do so, the ITS should help students to redirect (after initial inspection) their attention to the representations and to focus on them for a longer duration, rather than to frequently switch between different representations.

Furthermore, our findings suggest that the ITS should target the concepts of CPK color coding, bond angle, atomic radii, relative energy, bonding type, and reactivity. These concepts may be more difficult because they are more complex: they are typically used to reason about bonding phenomena that involve the interaction of one molecule with additional atoms and molecules rather than about the structure of individual atoms and molecules.

3 Design of a Chemistry Tutor for Connection Making

Study 1 encourages developing an ITS for chemistry that targets sense-making ability and perceptual fluency through separate activities. Study 2 suggests that sense-making activities should focus on differences between representations, not on similarities. Here we describe how these findings informed the design of a chemistry tutor.

3.1 Tutor Design

In line with prior research [3], *sense-making activities* are designed to help students in relating conceptually relevant aspects of different graphical representations. As Study 2 suggests, we focus on differences between representations in providing complementary information. Sense-making activities involve three parts. Consider a problem that targets one of the concepts that we found to be particularly difficult in Study 2: bonding type and electron behavior (Fig. 2). Students identify the type of bond between atoms and make inferences about how electrons are distributed across the molecule. First, they solve this problem with one representation (e.g., a Lewis structure, see Fig. 2A). Second, they solve a corresponding problem with another representation (e.g., an EPM, see Fig. 2B). Third, students are prompted to explain differences between representations (e.g., the local negative charge is shown by a larger number of electron-dots shown in Lewis structures, and by red color in EPMs; Fig. 2C).

The design of the *fluency-building activities* is based on Kellman and colleagues' perceptual learning paradigm [11]. Rather than focusing on why or how different representations correspond to one another, fluency-building support aims at helping students become faster and more efficient at extracting relevant information from the representations based on repeated experience with a large variety of problems. Thus, the fluency-building activities provide numerous practice opportunities to find corresponding graphical representations based on their perceptual properties. Fig. 3 shows two sample problems in which students have to choose a representation that show the

same molecule. Choices are designed to contrast which perceptual aspects provide relevant information. For instance, to solve the example on the left-hand side of Fig. 3, students have to attend to how EPMs depict the geometry of the molecule. To solve the example on the right-hand side, students need to attend to the lone pair in Lewis structures, which have implications for electronegativity that the EPM depicts as color. Students receive a series of these problems and are encouraged to solve them fast, by using perceptual properties and without overthinking the problem.

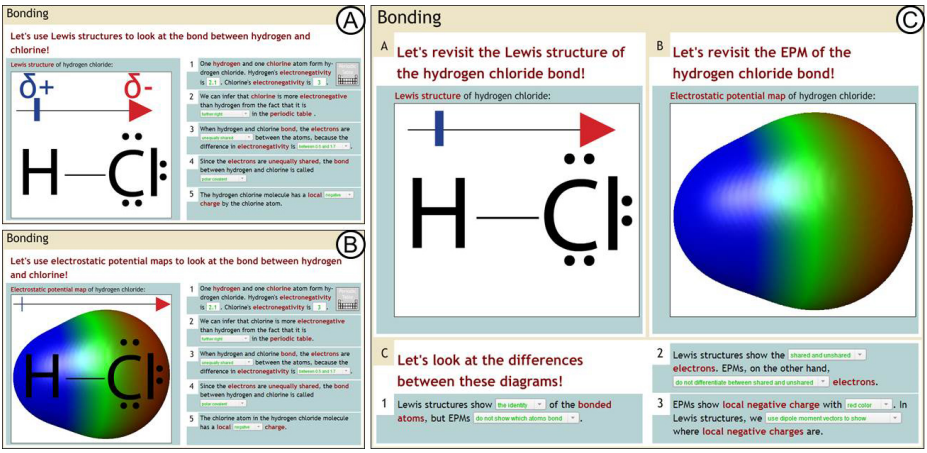


Fig. 2. Sense-making problems

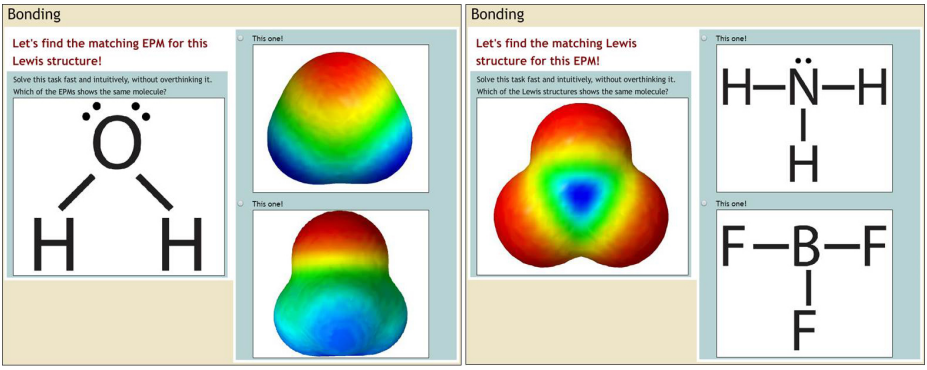


Fig. 3. Fluency-building problems.

3.2 Initial Pilot Results

We collected initial pilot data from four students who worked with a handful of sense-making and fluency-building prototypes. During the pilot sessions, we collected eye-tracking data, interview data, and tutor log data. The interview data suggests that

students like the tutor activities because they contain multiple graphical representations. For instance, one student commented, "I think it does a good job at showing multiple layouts instead of just one, so one can understand". The small sample size did not warrant a quantitative analysis of the eye-tracking data. Instead, we viewed the eye-gaze recordings and counted the number of times a student reinspected a graphical representation. For sense-making activities, this qualitative analysis suggests that impasses and reflection prompts (see Fig. 2C) are associated with subsequent reinspection of the representations. For fluency-building problems, we found that students frequently switch between representations. Finally, the log data showed that the reflection prompts (see Fig. 2C) had higher-than-average error rates. Fluency-building activities had a lower average error rate than sense-making problems.

In addition, we collected pre- and post-test data from three students in a second pilot study who worked with a fully-functioning version the ITS for one hour. We found learning gains of 16 percent points on sense-making items, 27 percent points on fluency items, and 7 percent points on transfer items about chemistry concepts.

3.3 Discussion

With respect to the sense-making activities, the pilot log data shows that sense-making prompts are challenging. This observation is in line with the finding in Study 1 that sense-making problems are difficult and further supports the conclusion that we need to support students' sense-making abilities, especially since Study 2 shows that noticing differences between representations is associated with conceptual inferences. Our qualitative analysis of the eye-tracking data suggests that impasses and prompts lead to reinspections of representations. This observation is promising because Study 2 showed that longer second-fixation durations are associated with inferences by helping students notice differences between representations. Thus, the pilot data suggests that sense-making activities enhance productive visual attention behaviors.

With respect to the fluency-building activities, further investigation is needed. The fact that the log data suggests that fluency-building activities are easier than sense-making activities stands in contrast to the finding of Study 1 that students have lower perceptual fluency than sense-making ability. On the one hand, one might conclude that the current design of the fluency-building activities enhances superficial visual processing because they are not difficult enough. On the other hand, we cannot necessarily draw the conclusion that frequent switching between representations and low error rates are associated with low learning gains, because the finding from Study 2 that frequent switching is associated with surface connections was based on an investigation of only sense-making items (not of perceptual fluency items).

Finally, pilot results on pretest to postest learning gains indicates that the ITS is effective in improving students' sense-making ability, perceptual fluency, and transfer of conceptual knowledge. An experiment testing the effectiveness of the sense-making and fluency-building components of the ITS is currently under way. Specifically, we will analyze mediation effects of eye-gaze behaviors, conceptual reasoning, and problem-solving behaviors on students' pretest to posttest learning gains.

4 Conclusion and Future Work

We described a multi-methods approach to ground the design of an ITS in the requirements specific to the target domain. Our goal in applying this approach to the chemistry domain was to inform the design of an ITS for connection making. Our empirical approach built on a theoretical framework that proposed two separate abilities: sense-making ability and perceptual fluency. We then conducted an assessment study that supports the existence of these two connection-making abilities in the chemistry domain. Even though this finding is in line with the chemistry education literature, which states that both skills are important aspects of chemistry expertise [9], our study is (to the best of our knowledge) the first to provide empirical support for this claim. Next, we conducted a study that combined eye-tracking and interview data to investigate which learning processes and concepts are most important with respect to sense-making ability. We found that making sense of differences between representations is more important than making sense of similarities between representations. Our data suggests that the visual mechanism by which students attend to differences between representations is to reinspect graphical representations rather than to frequently switch between representations (possibly among others). Furthermore, we identified several aspects of representations that undergraduates fail to identify spontaneously even though they constitute important chemistry concepts. Finally, our initial pilot results indicate that the ITS design enhances productive learning processes, that students perceive it as valuable, and that it leads to learning gains.

A limitation of the research described in the present paper is that our data are correlational in nature, but not causal. The results from Study 1 lead to the *prediction* that providing separate activities to support sense-making ability and perceptual fluency enhances students' learning in chemistry. Furthermore, the findings from Study 2 lead to the *prediction* that sense-making activities will enhance students' learning if they emphasize differences between representations rather than similarities, and if they help students to visually reinspect representations. The next step in our research is to experimentally test these predictions. We are currently conducting an experiment to evaluate the effectiveness of sense-making and fluency-building activities based on pretest to posttest learning gains, and to contrast whether (as hypothesized) students learn best when working with both sense-making and fluency-building activities, compared to working with either type of activity alone. Furthermore, we use the eye-tracking and interview measures described above to analyze whether (and how) students' visual attention patterns and connection-making utterances mediate the anticipated effects of the sense-making and fluency-building activities.

In sum, by using a multi-methods approach to ground ITS design in the specific requirements of the chemistry domain, we developed a system that appears to enhance productive learning processes and that addresses educational needs. Furthermore, this approach equips us with an initial theoretical model of how students' connection making might enhance their learning in chemistry and with a set of eye-tracking and interview measures that we can use to evaluate the effectiveness of the ITS. We conclude that our approach presents a useful methodology to identify *domain-specific* aspects that should shape the design of ITSs with multiple graphical representations.

Acknowledgements. This work was supported by the UW Madison Graduate School and the Wisconsin Center for Education Research. We thank Joe Michaelis, Abigail Dreps, Brady Cleveland, William Keesler, Taryn Gordon, and Theresa Shim for their contributions.

References

1. Ainsworth, S.: DeFT: A conceptual framework for considering learning with multiple representations. Learning and Instruction 16, 183–198 (2006)
2. Rau, M.A., Rummel, N., Aleven, V., Pacilio, L., Tunc-Pekkan, Z.: How to schedule multiple graphical representations? A classroom experiment with an intelligent tutoring system for fractions. In: Aalst, J.V., Thompson, K., Jacobson, M.J., Reimann, P. (eds.) Proceedings of the 10th International Conference of the learning Sciences (ICLS 2012) vol. 1, pp. 64-71. ISLS, Sydney, Australia (2012)
3. Bodemer, D., Faust, U.: External and mental referencing of multiple representations. Computers in Human Behavior 22, 27–42 (2006)
4. Rau, M.A., Aleven, V., Rummel, N., Rohrbach, S.: Sense Making Alone Doesn't Do It: Fluency Matters Too! ITS Support for Robust Learning with Multiple Representations. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 174–184. Springer, Heidelberg (2012)
5. Bodner, G.M., Domin, D.S.: Mental models: The role of representations in problem solving in chemistry. University Chemistry Education 4, 24–30 (2000)
6. Bowen, C.W.: Representational systems used by graduate students while problem solving in organic synthesis. Journal of Research in Science Teaching 27, 351–370 (1990)
7. Kozma, R., Russell, J.: Multimedia learning of chemistry. Cambridge handbook of multimedia learning, pp. 409-428 (2005)
8. Stieff, M.: Connected chemistry-A novel modeling environment for the chemistry classroom. Journal of Chemical Education 82, 489–493 (2005)
9. Wu, H.K., Krajcik, J.S., Soloway, E.: Promoting understanding of chemical representations: Students' use of a visualization tool in the classroom. Journal of research in science teaching 38, 821–842 (2001)
10. Gilbert, J.K., Treagust, D.F.: Towards a Coherent Model for Macro, Submicro and Symbolic Representations in Chemical Education. In: Gilbert, J.K., Treagust, D.F. (eds.) Multiple Representations in Chemical Education, pp. 333–350. Springer, Netherlands (2009)
11. Kellman, P.J., Massey, C.M., Son, J.Y.: Perceptual Learning Modules in Mathematics: Enhancing Students' Pattern Recognition, Structure Extraction, and Fluency. Topics in Cognitive Science 2, 285–305 (2009)
12. Cheng, M., Gilbert, J.K.: Towards a better utilization of diagrams in research into the use of representative levels in chemical education. In: Gilbert, J.K., Treagust, D.F. (eds.) Multiple Representations in Chemical Education, pp. 191–208. Springer, Netherlands (2009)
13. Johnson, C.I., Mayer, R.E.: An eye movement analysis of the spatial contiguity effect in multimedia learning. Journal of Experimental Psychology: Applied 18, 178–191 (2012)
14. Mason, L., Pluchino, P., Tornatora, M.C.: Effects of Picture Labeling on Science Text Processing and Learning: Evidence From Eye Movements. Reading Research Quarterly 48, 199–214 (2013)
15. Kozma, R., Russell, J.: Students becoming chemists: Developing representationl competence. In: Visualization in Science Education, pp. 121–145. Springer, Netherlands (2005)

Modeling Student Benefit from Illustrations and Graphs

Michael Lipschultz and Diane Litman

Computer Science Department, University of Pittsburgh
{lipschultz,litman}@cs.pitt.edu

Abstract. We examine a corpus of physics tutorial dialogues between a computer tutor and students. Either graphs or illustrations were displayed during the dialogues. In this work, stepwise linear regression, augmented to remove unwanted terms, is used to build models that identify situations when each graphic may aid learning. Our experimental results show that grouping students by pretest score, then by gender produces a model that significantly outperforms the baseline.

Keywords: student modeling, ITS, dialogue, graphs, illustrations, physics.

1 Introduction

One-on-one tutoring between a student and a human tutor is a very effective method of instruction [10]. Intelligent tutoring systems (ITS) have been developed to provide one-on-one tutoring, but from a computer-based tutor rather than a human tutor, and have been shown to improve student knowledge [17].

Visual representations, such as illustrations and graphs, are one method used to convey information to students thought to help them learn concepts. Illustrations use images to represent concepts [15,9], whereas graphs convey concepts primarily through such graphs as bar graphs or line graphs [15]. While much of the ITS research has made the assumption that one representation is best for everyone, differences exist between representations. Illustrations are easier for novices to interpret [12], but have surface features that may distract students [8]. Graphs can help students connect descriptions of situations to the base concepts [16], but students are more likely to make mistakes with them [14]. Researchers have thus examined the benefits of using multiple representations. Helping students become fluent in multiple representations and to be able to translate between them are beneficial [15]. Research into using multiple representations during tutoring tends to treat all students as identical; the switching of representations are on a fixed schedule [13,15]. However, research suggests that there are differences among students, leading to some representations being more beneficial than others. Student differences to consider include gender [14], spatial reasoning ability [9], and skill with domain concepts [9]. Adapting to students in other instances have had success, such as uncertainty and motivation leading to increased persistence and better learning gains [1,6].

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 436–441, 2014.

This paper explores building models to predict when illustrations and graphs benefit learning. We first describe an algorithm that constructs such models using stepwise linear regression augmented to conform to certain syntactic constraints. We then examine the models learned and find that models including both pretest score and gender when describing tutoring situations perform best.

2 Corpus

The data comes from a study comparing the effectiveness of showing illustrations versus graphs during conceptual physics tutoring with an ITS [11]. Subjects solved a physics problem in Andes [17], with the Rimac physics coach walking them through problem solving [7]. Andes presented the problem statement and a visual representing the situation described. Rimac provided instruction on solving the problem through a typed natural language dialogue. After solving the problem, subjects engaged in a reflection dialogue designed for students to reflect on concepts; it was a typed natural language discussion with a computer tutor. It began with a question on a key concept from the problem and after answering this question, the student has a discussion of the concept with the tutor. During this discussion, visuals were shown to help explain concepts.

Subjects saw only illustrations or only graphs during tutoring; the visuals presented the same information. Problems, reflection questions, and their orders, remained the same. Twenty-nine college students without college-level physics were recruited and randomly assigned one of the visuals to see. They began by filling out a background survey then completed a standard test for determining spatial reasoning ability [5]. They took a pretest to measure their incoming physics knowledge. At the end of tutoring, they took an isomorphic, counterbalanced post-test. We have 2043 data points at the utterance level.

Prior work on this corpus found differences from the pooled data using ANCOVAs [11]. This paper presents work on mining the data to learn models that can identify situations *when* illustrations or graphs were beneficial for learning.

2.1 Features

Features similar to those below have been used in previous work on tutoring systems [4,2,3] and have been found useful by cognitive science research on visual representations [14,9]. From this literature, we selected the features we could extract from the data collected during the study.

Gender – **Female** or **Male**
SpatialReason – score on the spatial reasoning test (**high**, **low**)[1]
Condition – experimental condition (**graph**, **illus**tration)
PreScore – score on pretest (**high**, **low**)
WalkThruPctCorrect – percent of correct answers in the current problem's walk through dialogue with the physics coach (**high**, **low**)

[1] Median splits were performed for ease of interpreting results.

RQPctCorrect – percent of correct answers in the current problem's prior reflection dialogue (**high**, **low**)
ProblemPctCorrect – percent of correct answers in current problem (both walk through dialogue and prior reflection dialogue(s)) (**high**, **low**)
SessionPctCorrect – percent of correct answers in session (**high**, **low**)
PctThruProblem – for each problem, how far through the dialogues (walk through and reflection) the subject has gone (**early**, **late**)
PctThruSession – how far through tutoring (# completed dialogues) (**early**, **late**)
KCusage – whether Knowledge Components (KCs) must be **stated** or **applied**
ItemDifficulty – whether the question is **easy** or **hard**, as determined by percent correct on a small pilot study using these dialogues

3 Modeling

To build an adaptive policy, we use stepwise linear regression to learn a model that explains the variance in post-test score using interactions between the features above. Standard stepwise regression produces rules that may be contradictory or non-adaptive, which are not helpful in creating an adaptive policy. We augment stepwise regression to address these additional constraints. We also constrain the syntax of the models to better describe the tutoring situation. Thus, we are trying to optimize r^2, subject to certain constraints.

The algorithm below shows how to learn an adaptive policy. Once learned, the policy can be applied at every decision point by starting at the top of the list and applying the first that applies.

1. Convert each feature into binary factors, one factor for each feature value. Each factor has a value of either 1 or 0, depending on whether the feature has that particular value for that data point.
2. Run stepwise linear regression on the data subject to syntactic constraints
   Model – Models have the form $postscore = \sum \text{terms} + prescore$. Both *postscore* and *prescore* are continuous variables. *Prescore* is included because pretest scores are often correlated with posttest scores; in this corpus it is a trend.
   Terms – Create terms by multiplying two or more factors together. Each term contains one Condition factor so that the final model learned can indicate situations when a visual helped or hindered learning. Additional factors in the term describe the situation.
3. Identify problematic term pairs. Problematic terms can be identified by:
   Contradictory pair – Two terms with opposite conditions and the other factors are identical. For example, 0.123*ConditionIsGraph* GenderIsFemale and 0.789*ConditionIsIllus*GenderIsFemale contradict each other because the first says to show graphs to females, while the second says to show illustrations.
   Non-adaptive pair – Two terms with the same factors, except one is opposite between the two terms. For example, 0.456*ConditionIsGraph* PctThruSessionIsLate and 0.123*ConditionIsGraph*PctThruSessionIsEarly are not adaptive since they say to show graphs regardless of the percent through tutoring.
4. For each problematic term pair, remove the one with the lower absolute value of the coefficient (avc)[2].

[2] We also explored removing both terms, but found that the final models did not perform as well.

Table 1. Models are compared across 10-fold cross validation according to adjusted r^2 values and their 95% confidence intervals. Italicized rows indicate results significantly better than baseline ($p < 0.05$). Underlined indicates the best result.

Model		Adj. r^2	95% CI
Baseline (Illustration)		0.1127	(0.0896, 0.1358)
1 Factor		0.0955	(0.0737, 0.1172)
2 Factors	*Gender*	*0.1788*	*(0.1428, 0.2148)*
	SpatialReason	0.1488	(0.1149, 0.1826)
	PreScore	*0.3499*	*(0.3266, 0.3732)*
	PctThruProblem	0.1007	(0.0635, 0.1378)
	PctThruSession	0.1180	(0.0851, 0.1509)
3 Factors (PreScore and ...)	*Gender*	*0.4571*	*(0.4220, 0.4922)*
	SpatialReason	*0.2817*	*(0.2367, 0.3267)*
	PctThruProblem	*0.3418*	*(0.3183, 0.3653)*
	PctThruSession	*0.3087*	*(0.2782, 0.3392)*

5. With the remaining terms, run multiple linear regression to learn the final model since the coefficient signs may change from the original model.
6. Convert the terms into rules and rank them using avc. The Condition factor indicates the visual to show and the other factors indicate the situation. For negative coefficients, use the visual opposite the one indicated by the Condition factor. Negative coefficients suggest that the visual is detrimental to learning in that situation.

4 Results

The models are compared to a baseline, which always predicts showing the same kind of graphic. We choose illustrations since they showed better learning gains. Models are each evaluated using ten-fold cross validation and are compared according to the adjusted r^2 value. The performance of the baseline can be seen in the first row of Table 1.

The "1 Factor" model contains only one factor describing the situation, plus the interaction feature Condition. As seen in Table 1, it is not significantly different than the baseline. Since all terms in this model consist of one non-Condition factor, the model can only identify situations by one feature (e.g. GenderIsFemale or PctThruSessionIsLate). This may not be enough to adequately describe situations when illustrations or graphs are more beneficial than the other; the descriptions may be too coarse-grained.

Finer-grained situation descriptions are created by adding more factors to each term. Five features were selected based on prior work suggesting a change in these features can cause large changes in models [11,9]: Gender, SpatialReason, PreScore, PctThruProblem, and PctThruSession. Five "2 Factor" models were created, one for each feature; two perform significantly better than baseline: Gender and PreScore, with PreScore significantly better than other models seen so far. Thus, keeping PreScore as the second feature, we add a third factor to this model, drawing from the same set of features. As seen in Table 1, all four "3 Factor" models

Table 2. Rules for the best 3 Factor model: PreScore*Gender

Female High Pretesters (n = 8)	Female Low Pretesters (n = 9)
1. If WalkThruPctCorrect=Low, show Graph 2. If RQPctCorrect=Low, show Graph 3. If SessionPctCorrect=High, show Illus 4. If ProblemPctCorrect=High, show Illus 5. If PctThruProblem=Early, show Graph 6. If PctThruSession=Early, show Graph	1. If SessionPctCorrect=High, show Graph 2. If PctThruSession=Early, show Illus 3. If ProblemPctCorrect=High, show Illus 4. If PctThruProblem=Early, show Illus 5. If RQPctCorrect=Low, show Illus
Male High Pretesters (n = 3)	**Male Low Pretesters (n = 9)**
1. If RQPctCorrect=Low, show Illus 2. If SessionPctCorrect=High, show Illus 3. If WalkThruPctCorrect=Low, show Illus	1. If RQPctCorrect=Low, show Illus 2. If WalkThruPctCorrect=Low, show Illus 3. If SessionPctCorrect=High, show Illus 4. If PctThruSession=Early, show Graph 5. If PctThruProblem=Early, show Graph 6. If ProblemPctCorrect=High, show Illus

perform significantly better than baseline, with PreScore*Gender performing significantly better than the rest; Table 2 has its policy.

In the model, we see differences between the partitions. Low pretesting females with a High PctSessionCorrect should be shown graphs, where as males and high pretesting females should be shown illustrations. When early in the tutoring session, low pretesting females should see illustrations whereas high pretesting females and low pretesting males should see graphs. When WalkThruPctCorrect is low, high pretesting females should see graphs whereas males should see illustrations. When RQPctCorrect is low, high pretesting females should see graphs but males and low pretesting females should see illustrations. That these differences exist in the model suggest that looking at interactions with both features improves situation description.

5 Discussion and Future Work

Prior work on this data found differences from the pooled data [11] by identifying when one group of students may benefit from one visual representation over another. This work identifies situations when one graphic might be better than the other for the same student and creates an adaptive model. In ongoing work, we have incorporated one model into a tutoring system and are evaluating its effectiveness at selecting visuals that aid learning compared to both alternating visual representations and using only one throughout tutoring.

This paper also presents a technique for mining data to create an adaptive policy when a gold standard is not available. It starts with a standard method (stepwise linear regression) and augments it to remove terms unwanted for developing adaptive systems. The method seeks to identify situations when one graphic is better than the other. Increasing situation descriptions, by adding more factors to each term, improve model performance. Many models, particularly those involving PreScore, significantly outperform the baseline. In ongoing

work, we are exploring improvements to model development, such as automatically identifying factors to add to a term to improve situational descriptions.

Acknowledgments. We thank Huy Viet Nguyen, Nathan Ong, and the Rimac team for their contributions. This research was supported by IES Grant R305A100163 to the University of Pittsburgh. The opinions expressed are those of the authors and do not represent the views of IES or the U.S. DoE.

References

1. Aist, G., Kort, B., Reilly, R., Mostow, J., Picard, R.: Experimentally augmenting an intelligent tutoring system with human-supplied capabilities: adding human-provided emotional scaffolding to an automated reading tutor that listens. In: IEEE International Conference on Multimodal Interfaces, pp. 483–490 (2002)
2. Arroyo, I., Beck, J.E., Park Woolf, B., Beal, C.R., Schultz, K.: Macroadapting animalwatch to gender and cognitive differences with respect to hint interactivity and symbolism. In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 574–583. Springer, Heidelberg (2000)
3. Chi, M., VanLehn, K., Litman, D.: Do micro-level tutorial decisions matter: Applying reinforcement learning to induce pedagogical tutorial tactics. ITS (2010)
4. D'Mello, S.K., Graesser, A.: Affect detection from human-computer dialogue with an intelligent tutoring system. In: Gratch, J., Young, M., Aylett, R., Ballin, D., Olivier, P. (eds.) IVA 2006. LNCS (LNAI), vol. 4133, pp. 54–67. Springer, Heidelberg (2006)
5. Ekstrom, R., French, J., Harman, H., Dermen, D.: Manual for kit of factor-referenced cognitive tests. Educational Testing Service, Princeton (1976)
6. Forbes-Riley, K., Litman, D.: Designing and evaluating a wizarded uncertainty-adaptive spoken dialogue tutoring system. Computer Speech & Language (2011)
7. Katz, S., Jordan, P., Litman, D., The Rimac Project Team: Rimac: A natural-language dialogue system that engages students in deep reasoning (2011)
8. Kohl, P.B., Finkelstein, N.D.: Student representational competence and self-assessment when solving physics problems. Phys. Rev. ST Phys. Educ. Res. (2005)
9. Kozhevnikov, M., Motes, M., Hegarty, M.: Spatial visualization in physics problem solving. Cognitive Science 31(4), 549–579 (2007)
10. Kulik, C., Kulik, J., Bangert-Drowns, R.: Effectiveness of mastery learning programs: A meta-analysis. Review of Educational Research 60(2), 265–299 (1990)
11. Lipschultz, M., Litman, D.: Illustrations or Graphs: Some Students Benefit From One Over the Other. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 746–749. Springer, Heidelberg (2013)
12. McDermott, L., Rosenquist, M.: vanZee, E.: Student difficulties in connecting graphs and physics: Exampies from kinematics. American Journal of Physics (1987)
13. McNeil, N.M., Fyfe, E.R.: Concreteness fading promotes transfer of mathematical knowledge. Learning and Instruction, 440–448 (2012)
14. Meltzer, D.: Relation between students problem-solving performance and representational format. American Journal of Physics 73, 463 (2005)
15. Rau, M., Aleven, V., Rummel, N.: Intelligent tutoring systems with multiple representations and self-explanation prompts support learning of fractions. AIED (2009)
16. Van Heuvelen, A., Zou, X.: Multiple representations of work–energy processes. American Journal of Physics 69, 184–194 (2001)
17. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J., Shelby, R., Treacy, D., Wintersgill, M.: The andes physics tutoring system: Lessons learned. IJAIED (2005)

Towards Assessing and Grading Learner Created Conceptual Models

Bert Bredeweg[1], Christina Th. Nicolaou[2],
Jochem Liem[1], and Constantinos P. Constantinou[2]

[1] Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands*
B.Bredeweg@uva.nl, Jochem.Liem@gmail.com
[2] Learning in Science Group, Department of Education, University of Cyprus
chr.nic@ucy.ac.cy

Abstract. Learning by creating models is an active form of learning, which is well suited to induce deep understanding of phenomena. But how to evaluated such models, and apply feedback accordingly? What makes a learner created model a good model? We present two methods to assess and grade *conceptual* models and report on the application of these to model-data obtained from learners in a summer science class.

Keywords: Conceptual models, Learning by modelling, Assessment.

1 Introduction

Learning by building models is an active form of learning during which learners create external representations in the form of models, and by doing so develop their understanding of phenomena [1,2]. A new group of tools is emerging that uses logic-based (symbolic, non-numerical) representations for expressing conceptual knowledge [3,4]. Different from numerical-based, they employ a qualitative vocabulary for users to construct their explanation of systems and how they behave. The use of graphical user interfaces has improved usability [5], and the tools are becoming common in education [6], and professional practice [7].

But how are teachers supposed to evaluate such *conceptual* models? There is a need to establish methods for analyzing learners ability to develop and deploy conceptual models [8]. This paper presents two approaches for assessing learner constructed models, and the application of these approaches in a real-word case.

2 Conceptual Models

The DynaLearn learning environment [9] enables learners to create conceptual models by working through several stages of representation from specifying and interpreting simple, static concept maps at the lowest level (level 1), to complex dynamic models with advanced representations for capturing causality at

* Co-funded by EU FP7, DynaLearn, 231526, http://www.dynalearn.eu & EU Regional Dev. Fund and Rep. of Cyprus, Didaktor/0311/92, Research Promotion FND.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 442–447, 2014.

the highest level (level 6). Consider the details in Fig. 1. This model has three **entities**, notably *First cube*, *Medium* and *Second cube*, pairwise connected by the **configurations** *Left of* and *Right of*. Both cubes are assigned the **quantity** *Temperature* and *Heat*, with a single **value** *Interval* and an unknown **direction of change** (δ). *Medium* has quantity *Flow*, which can take the values *Minus* (negative flow), *Zero* (no flow, steady), and *Plus* (positive flow). The current value and direction of change (δ) is unknown for *Flow*. The following dependencies hold. The magnitude of *Flow* is determined by the temperature difference. And *Flow* **negatively influences** ($I-$) the *Heat* of the *First cube*, and positively ($I+$) the *Heat* of the *Second cube*. Changes in heat **positively influence** ($P+$) changes in the *Temperature* quantities. Changes in temperature then **feed back** into changes of the *Flow*, positively from the *Temperature* of the *First cube* ($P+$), and negatively from the *Temperature* of the *Second cube* ($P-$). Finally, *Temperature* of the *First cube* is higher ($>$) compared to the *Temperature* of the *Second cube*, which together with the **subtraction** of the two temperatures implies that *Flow* is above zero (having the value *Plus*).

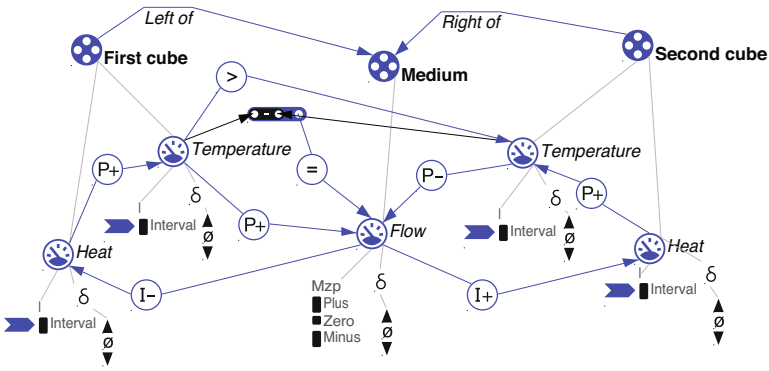


Fig. 1. Dynalearn model (level 4) showing energy exchange between two bodies

Fig. 2 shows the simulation result. The **state graph** (left top) shows the initial **scenario** and two **states**, with state 1 representing the behavior interpretation of the initial scenario, and state 2 being its successor. Second, the **inequality history** (left bottom) shows that the *Temperature* of the *First cube* is greater ($>$) compared to the *Temperature* of the *Second cube* in state 1, while in state 2 they become equal ($=$). Third, the **value history** (right top and bottom) shows the progression over states for the magnitude of quantities and their direction of change. For instance, *Flow* has value *Plus* in state 1 and decreases (arrow pointing down), and has value *Zero* in state 2 while being steady. Note that the temperatures do not change their qualitative value (magnitude). However, the *Temperature* of the *First cube* decreases in state 1, while the *Temperature* of the *Second cube* increases. In state 2 both quantities have stabilized ($\delta = 0$). Similar

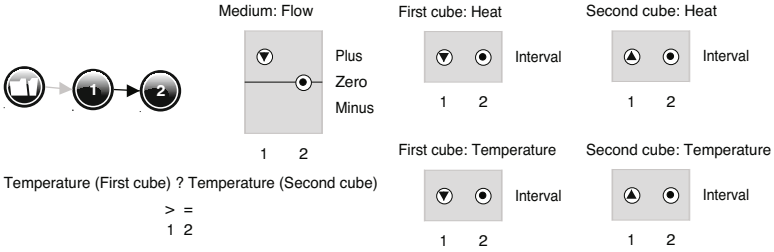


Fig. 2. Simulation results for the model shown in Fig. 1

to the temperatures, the *Heat* of the *First cube* decreases and the *Heat* of the *Second cube* increases in state 1, while both stabilize in state 2.

3 Method 1 - Modelling Practices Based Assessment

Liem [10] proposes a framework that distinguishes three categories for determining model quality. Verification based on **model errors** constitutes 50% of the metric. It is based on the different aspects that modelers need to learn to represent: Structure (10%), Quantities (5%), Quantity spaces (5%), Causality (10%), Inequalities and correspondences (10%) and Simulations (10%). The **communicative value** of the model constitutes 25% of the metric, and is based on: Quality of the layout (5%) and Documentation (20%). The **adequacy** of the model as a domain representation to fulfill a particular goal, determines the last 25% of the metric. It concerns: Correctness (10%), Completeness (10%), and Parsimony (5%). The two validation categories (communicative value and adequacy) are more subjective and rely on the expertise of the evaluator. The scores in each subcategory reflect both the things that a learner has done correctly and the errors that have been made. See for example the equation used to calculate the structure score given below. The scoring results in a number, which can be scaled to any grading system. The metrics of other subcategories are analogous.

$$\left(\frac{Entity + ConfigurationDefinitions - StructureErrors}{Entity + ConfigurationDefinitions}\right) * 100 = Score \qquad (1)$$

4 Method 2 - Criterion Referenced Artifact Analysis

This method draws from the principles of artifacts analysis [11]. Three aspects are assessed. **Representational quality** depends on the inclusion of objects, variable quantities, processes and relations. Objects (e.g. animals, plants, air) constitute the core ingredients of a model on which the rest is based. Variable quantities (e.g. size, population, velocity, temperature) are the changing features that characterize objects. Processes are mechanisms that cause change. In a model explaining thermal equilibrium, 'heat flow' is a process. Relationships

are all the inter-relationships between the other three model ingredients. These interrelationships can be causal, or none-causal. To analyse the representational feature, all relevant model ingredients are identified. The model receives score 1 for objects if none of the expected objects are represented, score 2 if only some of the objects are not included, and score 3 if all objects are represented. A similar approach is taken for the other aspects. The **interpretive function** relates to a models efficiency in providing an interpretation (score 0 if not present, 1 if the story does not cover all elements, and 2 if the story is coherent and includes all elements). A model has **predictive power** when it allows the formulation and testing of predictions for new aspects of the phenomenon it represents (score 0 when it does not, 1 when it does but results are incorrect, and 2 when it allows for correct predictions).

5 Deployment and Results

A course was given, which engaged learners in constructing and simulating conceptual models, involving 17 learners (15-17 years old). Learners worked in six groups of two or three learners. The learners were introduced to DynaLearn and the subject matter details regarding cups containing water and ice, with the goal for them to discover the difference between temperature and energy, and the proper causal model relating these quantities (see Fig. 1). Each group created a basic and a more advanced model.

Method 1. Table 1 shows the assessment of the basic models. To illustrate the scoring, consider model 1 made by group 2 (m1g2) (Fig. 3). The entities are not connected with configurations, hence two #29 structure errors (cf. [10]). As a result the equation (Eq. 1) becomes $(3 + 0 - 2)/(3 + 0) * 10$ yielding a score of 3,3. Error #20 (incorrect type of causality) is also present; the model included a $P+$ relationship from *Flow (Medium)* to *Temperature (Cube one)* which should be reversed. Similarly, the $I+$ relationship from *Temperature (Cube one)* to *Flow (Medium)* should be reversed. The result is 5, which reflects the 4 causal relationships minus the 2 incorrect divided by the total, $(4 - 2)/4 * 10$. Concerning the simulation (not shown in Fig. 3), the 10 reflects a correct

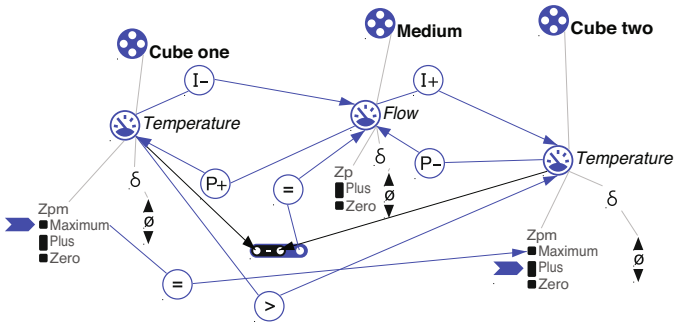


Fig. 3. Model 1 created by group 2 (m1g2)

simulation with no dead ends (error #34). There are also no mistakes concerning quantities, quantity spaces, and inequalities and correspondences (hence scores 5, 5, and 10, respectively). The model is complete (10) and parsimonious (5), but not totally correct (7.9), as the quantity space of temperature is incorrect and the model does not reach equilibrium, hence $(14-3)/14*10$, where 14 is the set of model elements and 3 the aforementioned errors. The model layout is fine but it lacks the correct documentation. The representational, interpretational, and predictive power are not well presented (4.8 in total): $((0,66+1+1+0,2)/4+0+0)/3*20$, with 0,66 for entities, 1 for quantities, 1 for processes, and 0,2 for the relations, all divided by 4, as it received 0 for the description of the other two elements.

Table 1. Score for model 1 using method 1 (m1g1 = model 1 of group 1, etc.)

Criteria	**m1g1**	**m1g2**	**m1g3**	**m1g4**	**m1g5**	**m1g6**	**Average**
Structure 10%	8.3	3.3	4.3	5	10	3.3	5.7
Quantities 5%	5	5	3.3	5	2.5	2.5	3.9
Quantities spaces 5%	5	5	5	5	5	5	5
Causality 10%	7.5	5	10	10	0	0	5.4
Ineq. & Corres. 10%	10	10	10	10	0	10	8.3
Simulations 10%	0	10	10	0	0	10	5
Total Errors (50%)	35.8	38.3	42.6	35	17.5	30.8	33.3
Correctness 10%	7.1	7.9	8.9	8.9	7.5	7.9	8.0
Completeness 10%	10	10	10	10	8.3	10	9.7
Parsimonious 5%	4.7	5	4.6	4.2	3.8	5	4.5
Total adequacy (25%)	21.8	22.9	23.4	23.1	19.6	22.9	22.3
Layout of the model 5%	5	5	5	5	5	5	5.0
Documentation 20%	18.5	4.8	9.4	5.2	12.2	4.8	9.2
Total communication (25%)	23.5	9.8	14.4	10.2	17.2	9.8	14.2
Total score	**81.1**	**71**	**80.5**	**68.3**	**54.3**	**62.5**	**69.8**

Method 2. Table 2 presents the evaluation using method 2, focussing on: representation, interpretation and prediction. Consider again model m1g2. It received full points for the represented objects (*Cube one*, *Cube two* and *Medium*) and processes (*Flow*). It received 1 point for variable quantities due to problems identified earlier concerning quantity spaces of *Temperature*, and 1 point of relationships due to the errors identified concerning $P+$ relations. The interpretation story for the model was semi-correct and received 1 point. The model explains the phenomenon by equalizing *Flow* to *Temperature difference*, correctly identifying that *Flow* negatively influences the *Temperature* of *Cube one*, and stating that a greater temperature difference results in a greater *Flow*. It received 2 points for prediction as the model allows for changing initial values and observe the results.

Table 2. Score for model 1 using method 2

Model features	**m1g1**	**m1g2**	**m1g3**	**m1g4**	**m1g5**	**m1g6**	**Average**
Representation							**1.33**
Objects	2	2	2	2	0	2	1.67
Variable properties	1	1	2	1	0	1	1.00
Processes	2	2	2	2	0	2	1.67
Relationships	1	1	2	2	0	0	1.00
Interpretation	1	1	2	2	0	0	**1.00**
Prediction	0	2	2	1	0	2	**1.17**

6 Conclusion

This paper contributes to the research area of modeling competence assessment. It describes two complementary assessment methods. The obtained data indicate that the two methods were successful in capturing the differences between learners as well as between the subcategories of each assessment criterion.

References

1. Crawford, B., Cullin, M.: Supporting prospective teachers conceptions of modelling in science. International Journal of Science Education 26(11), 1379–1402 (2004)
2. Louca, L.T., Zacharia, Z.C.: Modeling-based learning in science education: cognitive, metacognitive, social, material and epistemological contributions. Educational Review, 1–22 (2011)
3. Leelawong, K., Biswas, G.: Designing learning by teaching agents: The betty's brain system. International Journal of Artificial Intelligence in Education 18, 181–208 (2008)
4. Bredeweg, B., Linnebank, F., Bouwer, A., Liem, J.: Garp3 - workbench for qualitative modelling and simulation. Ecological informatics 4(5-6), 263–281 (2009)
5. Bouwer, A., Bredeweg, B.: Graphical means for inspecting qualitative models of system behaviour. Instructional Science 38(2), 173–208 (2010)
6. Kinnebrew, J.S., Biswas, G.: Modeling and measuring self-regulated learning in teachable agent environments. Journal of e-Learning and Knowledge Society 7(2), 19–35 (2011)
7. Bredeweg, B., Salles, P.: Qualitative models of ecological systems – editorial introduction. Ecological Informatics 4(5-6), 261–262 (2009)
8. Songer, N., Ruiz-Primo, M.A.: Assessment and science education: Our essential new priority? Journal of Research in Science Teaching 49(6), 683–690 (2012)
9. Bredeweg, B., Liem, J., Beek, W., Linnebank, F., Gracia, J., Lozano, E., Winer, M., Bhling, R., Salles, P., Noble, R., Zitek, A., Borisova, P., Mioduser, D.: Dynalearn an intelligent learning environment for learning conceptual knowledge. AI Magazine 34(4), 46–65 (2013)
10. Liem, J.: Supporting Conceptual Modelling of Dynamic Systems: A Knowledge Engineering Perspective on Qualitative Reasoning. Phd thesis, University of Amsterdam, Amsterdam, The Netherlands (2013)
11. LeCompte, M.D., Preissle, J.: Ethnography and Qualitative Design in Educational Research, 2nd edn. Academic Press, CA (1993)

StaticsTutor: Free Body Diagram Tutor for Problem Framing

Enruo Guo[1], Stephen Gilbert[2], John Jackman[2], Gloria Starns[3], Mathew Hagge[3], LeAnn Faidley[4], and Mostafa Amin-Naseri[2]

[1] Department of Computer Science
[2] Department of Industrial & Manufacturing Systems Engineering
[3] Department of Mechanical Engineering
Iowa State University, Ames, IA 50011, USA
[4] Department of Mechanical Engineering
Wartburg College, Waverly, IA 50677, USA
{enruoguo,gilbert,jkj,gkstarns,fforty,aminnas}@iastate.edu,
leann.faidley@wartburg.edu

Abstract. While intelligent tutoring systems have been designed to teach free-body diagrams, existing software often forces students to define variables and equations that may not be necessary for conceptual understanding during the problem framing stage. StaticsTutor was developed to analyze solutions from a student-drawn diagram and recognize misconceptions at the earliest stages of problem framing, without requiring numerical force values or the need to provide equilibrium equations. Preliminary results with 81 undergraduates showed that it detects several frequent misconceptions in statics and that students are interested in using it, though they have suggestions for improvement. This research offers insights in the development of a diagram-based tutor to help problem framing, which can be generalized to tutors for other forms of diagrams.

Keywords: intelligent tutoring system, statics tutor, free-body diagram.

1 Introduction

If you are one of the over 100,000 freshman engineering students in the U.S. [1], you will likely take a course in engineering mechanics called Statics. Streveler, et al.'s [2] elegant overview of conceptual learning within engineering notes that statics, along with thermal science and electrical circuits, is one of the most difficult domains for students. Chi [3] and Reiner, et al. [4] address the question of why some misconceptions are particularly prevalent and difficult to correct. Their results suggest that particularly problematic misconceptions may be based on metaphors to physical phenomena that are similar but not quite right, e.g., thinking of electrical current as a fluid. Or, difficult misconceptions are based on phenomena with unobservable components or relationships. Computer simulations are mentioned as a potential solution. This paper describes the StaticsTutor, an attempt to provide students not only with the simulation, but also with interactive feedback that directly addresses conceptual challenges.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 448–455, 2014.

StaticsTutor was developed as part of a problem framing research project funded by the National Science Foundation (EEC-1025133). This paper describes an initial assessment of the StaticsTutor's usability and an exploration of the dynamics of its usage with 81 engineering undergraduates.

A student assigned a homework problem using StaticsTutor sees a problem statement and a descriptive figure for the statics problem shown on the left side of a web-based interface (see Figure 1). The student begins by first drawing the overall problem information in a drawing area to the right of the problem statement, i.e., a beam, its support forces, external forces, and/or moments. The drawing software offers a typical drawing palette but has been customized for engineering. It includes tools to create force vectors, moments, labels, coordinate systems, etc.

At any point while working on the problem, the student may click a button labeled, "Check Free Body Diagram," which invokes the tutor. The tutor provides feedback in a popup window based on comparing the student's diagram with a known solution to this problem and with a set of general rules about free body diagrams that apply across problems. The student receives feedback at two conceptual levels. First, are all the pieces of the diagram present? Second, are there any likely misconceptions?

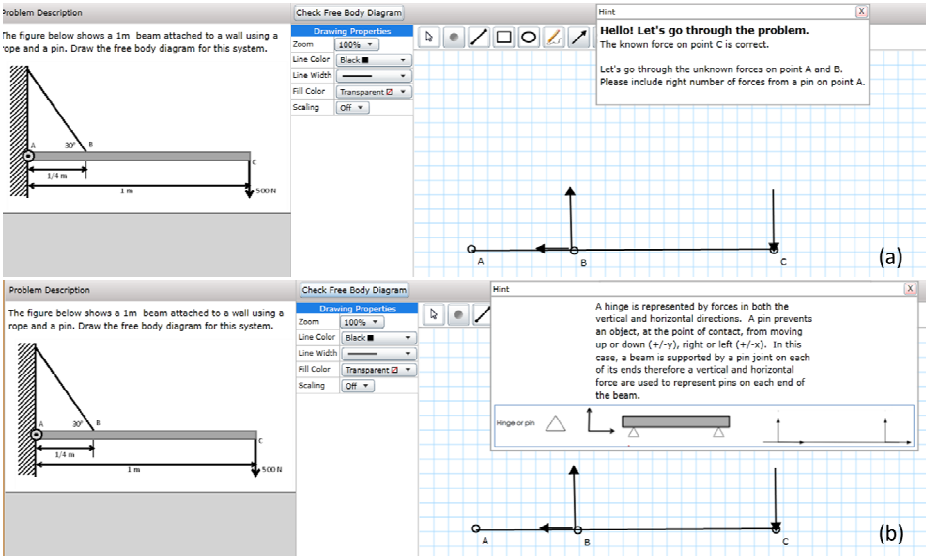


Fig. 1. A rope and pin problem. (a) A pop-up window summarizes problem-specific feedback. (b) A pop-up window is triggered by a potential misconception and gives an example on how a pin/hinge reacts in a system and how the reaction forces should be represented.

The student's tutoring experience described above raises the following research questions. 1) How is StaticsTutor different than existing tutors for science and engineering diagrams? 2) How do students engage with StaticsTutor, and do they find it usable and useful? 3) Does StaticsTutor help prevent misconceptions on future problems in the real world? This paper focuses on the first question.

2 Related Work

It is well established that engineering problem solving skills are critical for students to become practicing engineers. The most important stage of problem solving is problem framing, which occurs at the onset of problem solving in which students structure the problem using reasoning and metacognitive skills [5, 6]. When they know to do so, students typically attempt framing as the first step of the problem solving process [7, 8], and Voss & Post [9] found that early framing leads to better success in later stages of the problem.

A model-tracing tutor can be successful for well-defined problem-solving procedures, but recognizing the student's model for solving an open-ended engineering challenge is a work in progress. In the last a few decades, intelligent tutoring systems (ITSs) have steadily improved to make content more accessible to the average students [10]. Tutors have been used both in class and for homework in mathematics, physics, computer programming, and other subjects [11, 12]. While these systems have been successful, very few explicitly tutor on underlying concepts, focusing instead on helping students master the procedural skills. There have been exceptions, notably the effort with Andes to address conceptual problems [13]. StaticsTutor was designed with the intent to distract students as little as possible with numerical values and focus on the concepts of statics: equilibrium, resolution of forces into their orthogonal components, and summation of moments about any moment axis consistently using the rules of vector multiplication.

In the domain of intelligent tutors, we can expect more personalized feedback and conceptual teaching. Evaluations have shown that students who did their homework on Andes [12] learned significantly more than students who did the homework on paper. Whereas most tutoring systems have students enter only the answer to a problem, Andes has students enter several intermediate steps, such as drawing vectors, drawing coordinate systems, defining variables and writing equations while providing feedback after each step. When students ask for help in the middle of problem solving, Andes gives hints on what is wrong or on what kind of step to take next.

There are some other existing diagram-based tutoring systems. COLLECT-UML [14] supports individual and collaborative learning of UML class diagrams. EER-Tutor [15] helps learning and practicing principles of Enhanced Entity-relationship modelling. Free-Body-Diagram Assistant [16] provides students opportunities to construct FBDs for the human body and receive constructive feedback in biomechanics. CogSketch [17], which is a sketch-based educational software application, has demonstrated the powerful ability to understand sketched shapes and recognize them even after rotation or change of position. Labeling provides a rapid way to match instructor and student components. However, in engineering statics, many problems ask the student to define one or more forces without requiring specific labels on those forces, so matching by labels has some limitations. Mechanix [18] is a free-body diagram tutoring system based on free-hand drawing recognition. A checklist area is shown with specific instructions to guide students in order to finish the problem, However, a typing mode is still needed to check the value of each force. Unlike the existing free body diagram tutors, StaticsTutor addresses conceptual understanding at the problem framing stage.

3 StaticsTutor Interface and Architecture

The tutor uses a web-based drawing interface, XDraw, developed internally by author Jackman using the Microsoft Silverlight framework. A backend database saves students diagrams. StaticsTutor communicates with XDraw via a TCP socket between the two servers. Currently, authoring is based on xml files on the tutor server. In the future, a GUI will be implemented to serve as an authoring tool and interface. XDraw supports basic drawing objects such as points, lines, rectangles and vectors as well as free-hand drawing. A coordinate system is an object that is defined by the drawing tool as well. Students can rotate the coordinate system to facilitate solving the problem so that the angle of the forces would be adjusted based on the new axes.

The drawing can be designated as scaled or un-scaled. A scaled drawing allows the student to set up a grid scale for distance and the magnitude and units of force vectors. Also, a measuring tool is provided to measure distance between any two points. The scaled mode can be applied to vector magnitude check and distance measurement. The un-scaled mode provides more flexibility in creating a free-body diagram at the problem framing stage. For this study, students performed problems using the un-scaled mode.

The overall architecture of StaticsTutor includes 5 parts: domain model, expert solution, evaluation sequence, domain-wide check functions and student model. It has been used for a tutor in thermo-dynamics courses as well, the ThermoCycleTutor [19, 20], which indicates the generality and feasibility to different domains.

The problem solution, created by an expert instructor, contains a correct diagram and appropriate force values. The tutor can designate the student correct if the student's diagram is conceptually equivalent to the problem solution even if not identical. E.g., for a pin, the force could be pulling from below or pushing from above and both would be correct. Similarly, point B in Figure 1 must be between A and C, and not too close to either side, but its exact position is not important. The problem solution contains the number of forces reacting at each point, and the angle of each force. Magnitudes of forces are not represented because this approach is focused on conceptual structure. The tutor can accept all the possible correct angles. A tolerance value for the angles can be added by the solution author to make the tutor more accepting of student vectors that are not exact, e.g., 10 degrees. Also, the tutor gives students the freedom to draw either resultant forces or resolved forces in a selected orthogonal coordinate system.

StaticsTutor's problem evaluation sequence evaluates the free-body diagram in a number of steps inspired by the three overall stages of problem framing defined by several authors [21, 22]. First, the learner defines the stated problem. Second, the learner reflects on the stated problem, which involves a) a review of his or her personal assumptions about the problem situation, b) identification of a clear interpretation of the problem prior to considering the possible solution and c) identification of preexisting solutions embedded in the initial problem situation. Third, the learner reframes the problem if necessary.

In the example problem in this paper (Figure 1), the StaticsTutor problem evaluation sequence contains the following eight steps. A beam (1 meter) is attached to a

wall using a pin (point A) and a rope (point B). An external force (500 Newtons) acting at point C is pushing downward on the beam. The weight of the beam can be neglected since the problem statement does not indicate otherwise. Initially, the tutor takes three steps to consider if the student has clear recognition of the stated problem, which contains check of whether all non-force components are present, and a check of the given forces: Step_1 “if a beam is present and point A, B and C are present and in correct relative location,” Step_2 “if the number of forces at point C is correct,” and Step_3 “if the angle of force at point C is correct.” These three steps correspond to the first stage of problem framing, defining the problem. Each step functions by calling one or more of the domain-wide check functions. E.g., a check function that accepts a diagram point and the number of expected forces at that point per the problem solution might look like `numForcesCorrect(point, expectedNumForces)` and return a True or False. More detail on check functions provided below.

The tutor's next steps (4-7) focus on stage two of problem framing, whether the student has a clear interpretation of the problem. It contains Step_4 “if the number of forces at point A is correct,” Step_5 “if the angle of the forces at point A are correct,” Step_6 “if the number of forces at point B is correct,” and Step_7 “if the angle of the forces at point B are correct.” If any of the angles of the student diagram do not match the problem solution, the tutor will offer problem-specific feedback, such as, "Please check the angle of the force at point A." Also, the tutor evaluates the student's interpretation of the problem by evaluating the diagram components with a pool of domain-wide misconceptions. If the pin (point A), for example, does not have both its vertical and horizontal components represented, then the student may not understand that a pin exerts forces in both directions, and would receive the misconception feedback for pins. There are similar misconception feedback checks for ropes, hinges, and rollers. Domain-wide misconception feedback contains text and pictorial explanations created by co-authors Starns and Faidley, who teach engineering statics, and has been used in multiple problems.

The last stage is to check student’s reframing of the problem. To check for reframing, StaticsTutor looks for extra information that may have drawn initially before the student recognized the type of problem appropriately. Step_8 checks for anomalies such as a force that is not associated with any point or line, or an extra point that is not needed. Each of the eight steps needed to be satisfied for the problem to be complete.

Note that the eight steps of the problem evaluation sequence described above were specific to the particular problem posed, although the evaluation functions they used were the domain-wide check functions. Therefore, problem authors can change the evaluation sequence based on the needs of the problem or based on different pedagogical preferences, though most likely the sequence will still correspond to the three stages of problem framing. For example, the check for extra information might be removed if the instructor is not interested in this sort of evaluation. Also, some experts do not require students to draw the beam if the beam’s weight is negligible, in which case they might chose to remove the beam check.

It is important to describe the check functions in more detail because they support StaticsTutor's approach that is agnostic of force values, enabling the more generic

problem framing analysis. A pool of domain-wide check functions is available to the problem evaluation sequence. This pool contains general checks applicable to all problems, such as check the angle of a force, check the number of forces attached to a point. Check functions were designed in three levels to handle forces. First, a force can be directly accessed via its label, if it has been predefined by the problem statement, i.e., F_1, and the student has labeled it. Second, a force can be indirectly accessed via its contacting point, e.g., a pin, from a heuristic spatial relation check of its head and tail within a tolerance. However, if its head and tail are touching two different points, this force would be assigned to both points. StaticsTutor cannot resolve this ambiguity issue for now, which also would be difficult to resolve for a human instructor. In most situations, the name of the contacting point is given in the problem statement. Students have the flexibility to draw forces attached to contacting points and name the forces to their liking. By this means, StaticsTutor does not need to enforce labeling of the forces in order to ensure a match with the expert solution. Third, a force can be attached to other objects, e.g., a line, where it might represent the weight of a beam. StaticsTutor gives point association a higher priority than other object association. So a force is considered as *object-associated* only if it is not attached to any point.

Lastly, a model of the student is constructed for the purposes of tracking student performance, recording the student's misunderstandings and facilitating the instructor's analysis. The student model contains: 1) the series of student drawings, each of which is automatically saved when the student sends a request for feedback; 2) the feedback message generated by the tutor; and 3) answers to a post-survey that is administered using third-party software.

4 Preliminary Evaluation

StaticsTutor was tested on 81 engineering undergraduates in Fall 2013 who were enrolled in first-year mechanical engineering courses. Each student completed a statics problem with StaticsTutor. While a complete analysis of results is beyond the scope of this paper, it is worth noting that the categories of errors (Table 1) correspond with common misconceptions elicited from the instructor about hinge and rope elements, which validates our tutor design. 24% of students solved the problem completely before their first request for feedback and the remaining students' had request counts with a mean of 6.8 and median of 4. Their times to complete the problems were similarly distributed with a maximum time of 44 minutes, mean of 7.8 minutes, and median of 3.8 minutes. In total, 79% of students completed the problem, despite their initial misconceptions.

Table 1. Categories of Tutor Feedback Across 714 Requests (n=81; students could err in multiple categories)

Basics missing	Forces at C missing	Hinge & Rope	Rope Issue	Hinge Issue	Extraneous Info	Fully Correct
25.5%	2.8%	15.7%	16.1%	18.9%	1.4%	19.6%

5 Conclusions and Future Work

As most existing ITSs require students to define variables and equilibrium equations that may not be necessary during the problem framing stage, this initial investigation of StaticsTutor provides a guideline on how ITSs could help students at the initial stage of problem solving at a conceptual level, with little distraction on problem-specific input. This architecture could easily apply to domains beyond diagram tutors. Its evaluation architecture is based on the three subskills in problem framing: 1) defining, 2) reflecting and 3) reframing the problem. It can evaluate a free-body diagram without requiring labeling of the forces, and can detect misconceptions based on each problem component. The architecture also allows students the option to either draw resultant forces or decompose them using a specified coordinate system. Future work will integrate the ability to customize feedback based on previous errors made by an individual student, as well as evaluate the impact of StaticsTutor on classroom assessments, a statics concept inventory, and other learning measures.

References

1. National Science Foundation, National Center for Science and Engineering Statistics (NSF/NCSES): Women, Minorities, and Persons with Disabilities in Science and Engineering: 2013. Arlington, VA (2013)
2. Streveler, R.A., Litzinger, T.A., Miller, R.L., Steif, P.S.: Learning Conceptual Knowledge in the Engineering Sciences: Overview and Future Research Directions. Journal of Engineering Education 97, 279–294 (2008)
3. Chi, M.T.H.: Commonsense Conceptions of Emergent Processes: Why Some Misconceptions Are Robust. Journal of the Learning Sciences 14, 161–199 (2005)
4. Reiner, M., Slotta, J.D., Chi, M.T.H., Resnick, L.B.: Naive Physics Reasoning: A Commitment to Substance-Based Conceptions. Cognition and Instruction 18, 1–34 (2000)
5. Diefes-Dux, H.A., Salim, A.: Problem Formulation during Model-Eliciting Activities: Characterization of First-Year Students' Responses. In: Proceedings of the Research in Engineering Education Symposium 2009, Palm Cove, QLD (2009)
6. Redish, E.F., Smith, K.A.: Looking beyond content: Skill development for engineers. Journal of Engineering Education 97, 295–307 (2008)
7. Liikkanen, L.A., Perttula, M.: Exploring problem decomposition in conceptual design among novice designers. Design Studies 30, 38–59 (2009)
8. Litzinger, T., Lattuca, L.R., Hadgraft, R., Newstetter, W.: Engineering education and the development of expertise. Journal of Engineering Education 100, 123–150 (2011)
9. Voss, J.F., Post, T.A.: On the solving of ill-structured problems. In: Chi, M.T.H., Glaser, R., Farr, M.J. (eds.) The Nature of Expertise, pp. 261–285. Lawrence Erlbaum Associates, Hillsdale (1988)
10. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. International Journal for Artificial Intelligence in Education 8, 30–43 (1997)
11. Corbett, A.T., Koedinger, K., Hadley, W.S.: Cognitive tutors: From the research classroom to all classrooms. In: Goodman, P.S. (ed.) Technology Enhanced Learning: Opportunities for Change, pp. 235–263. Lawrence Erlbaum Associates, Mahwah (2001)

12. Vanlehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes Physics Tutoring System: Lessons Learned. Int. J. Artif. Intell. Ed. 15, 147–204 (2005)
13. Rosé, C.P., Jordan, P., Ringenberg, M., Siler, S., VanLehn, K., Weinstein, A.: Interactive conceptual tutoring in Atlas-Andes. In: Proceedings of AI in Education 2001 Conference, pp. 151–153 (2001)
14. Baghaei, N., Mitrović, A.: COLLECT-UML: Supporting Individual and Collaborative Learning of UML Class Diagrams in a Constraint-Based Intelligent Tutoring System. In: Khosla, R., Howlett, R.J., Jain, L.C. (eds.) KES 2005. LNCS (LNAI), vol. 3684, pp. 458–464. Springer, Heidelberg (2005)
15. Zakharov, K., Mitrovic, A., Ohlsson, S.: Feedback Micro-engineering in EER-Tutor. In: Proceedings of the 2005 Conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology, pp. 718–725. IOS Press (2005)
16. Roselli, R.J., Howard, L., Cinnamon, B., Brophy, S., Norris, P., Rothney, M., Eggers, D.: Integration of an interactive free body diagram assistant with a courseware authoring package and an experimental learning management system. Proceedings of the American Society for Engineering Education (2003)
17. Forbus, K., Usher, J., Lovett, A., Lockwood, K., Wetzel, J.: CogSketch: Sketch Understanding for Cognitive Science Research and for Education. Topics in Cognitive Science 3, 648–666 (2011)
18. Valentine, S., Vides, F., Lucchese, G., Turner, D.: Mechanix: A Sketch-Based Tutoring System for Statics Courses. IAAI, AAAI (2012)
19. Amin-Naseri, M., Guo, E., Gilbert, S., Jackman, J., Hagge, M., Starns, G., Faidly, L.: Authoring a Thermodynamics Cycle Tutor Using GIFT. In: AIED 2013 Workshops Proceedings, vol. 7, p. 45 (2013)
20. Jackman, J., Gilbert, S., Starns, G., Hagge, M., Faidly, L.: Problem Framing Behavior in Statics and Thermodynamics. In: Proceedings of the 2013 Annual ASEE Conference, Atlanta, GA (2013)
21. Cuban, L.: Problem-finding: Problem-based learning project. Stanford University, School of Education (1990)
22. Leithwood, K.A., Stager, M.: Expertise in Principals' Problem Solving. Educational Administration Quarterly 25, 126–161 (1989)

Are Automatically Identified Reading Strategies Reliable Predictors of Comprehension?

Mihai Dascalu[1], Philippe Dessus[2], Maryse Bianco[2], and Stefan Trausan-Matu[1]

[1] University Politehnica of Bucharest, Computer Science Department, Romania
{mihai.dascalu,stefan.trausan}@cs.pub.ro
[2] LSE, Univ. Grenoble Alpes, France
{philippe.dessus,maryse.bianco}@upmf-grenoble.fr

Abstract. In order to build coherent textual representations, readers use cognitive procedures and processes referred to as reading strategies; these specific procedures can be elicited through self-explanations in order to improve understanding. In addition, when faced with comprehension difficulties, learners can invoke regulation processes, also part of reading strategies, for facilitating the understanding of a text. Starting from these observations, several automated techniques have been developed in order to support learners in terms of efficiency and focus on the actual comprehension of the learning material. Our aim is to go one step further and determine how automatically identified reading strategies employed by pupils with age between 8 and 11 years can be related to their overall level of understanding. Multiple classifiers based on Support Vector Machines are built using the strategies' identification heuristics in order to create an integrated model capable of predicting the learner's comprehension level.

Keywords: Self-Explanations, Reading Strategies, Comprehension Prediction, Identification Heuristics, Support Vector Machines.

1 Introduction

In order to build textual coherence and to achieve a consistent representation of the discourse, readers need to transcend beyond what is explicitly expressed by employing cognitive procedures and processes, referred to as reading strategies. Those procedures are elicited through self-explanations [1]. Research on reading comprehension has shown that expert readers use specific strategies to on-line monitor their reading, thus being able to know at every moment their level of understanding. Moreover, when faced with a difficulty, learners can call upon regulation procedures, also part of reading strategies [2]. In this context, psychological and pedagogical research has revealed that people tend to understand better a text if they try to explain themselves what they have read [3]. Starting from these observations, techniques such as *SERT* (Self-Explanation Reading Training) [4], were developed to support students better understand texts.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 456–465, 2014.

Reading strategies have been extensively studied with adolescent and adult readers using the think-aloud procedure that engages the readers to self-explain what they understood so far at specific breakpoints while reading. Our study is focused on comprehension assessment for an audience more rarely studied, primary pupils, whose guidance plays a central role. As previous research suggests, self-regulation can be enhanced through the use of metacognitive reading strategies [5]. Pupils tend to better understand a given text by employing these specific mechanisms [6]. Also, this paper represents a continuation of previous research [7, 8], with a refined set of heuristics for best matching human annotations, accompanied by a prediction mechanism based on Support Vector Machines [9] in order to estimate pupil's comprehension level of a given text.

The following section presents an overview of the evaluation of reading strategies, their categorization, and other similar automated systems that have been developed to identify the employed reading strategies. The third section is centered on the description of the used heuristics, while the fourth section introduces the classifier that combines the previously identified reading strategies and predicts the learner's comprehension level. Afterwards, the fifth section encompasses the performed validations for testing the system's accuracy, while the last section is focused on conclusions and future improvements.

2 Overview of the Assessment of Reading Strategies

Expert readers frequently make use of four types of reading strategies in order to achieve a deep understanding from the texts they read [4]. *Paraphrasing* enables readers to express what they understood from the explicit content of the text and can be considered the first and essential step in order to achieve a coherent representation. *Text-based inferences*, consisting predominantly of causal and bridging strategies, build explicit relationships between two or more textual segments of the initial text. On the other hand, *knowledge-based inferences* create relationships between the information from the text and the reader's personal knowledge and are essential to create the situation model [10]. *Control strategies* refer to the actual monitoring process, when readers explicitly express what they have or have not understood.

Nevertheless, if we want students to be assisted while reading, one human expert (e.g., a teacher) can take care only after a small number of them, which makes it impossible for such training techniques to be used on a large scale. For example, this is one of the major problems of MOOCs (Massively Online Open Courses) in which, due to the previous constraints, assistance is frequently provided by peer students, increasing nevertheless the risk of making mistakes [11, 12]. Moreover, assessing the content of a verbalization is a demanding and a subjectivity-laden activity, which can be assisted by computer-based techniques. These are the main motives behind the idea of using a computer program instead of, or as support for, a human tutor.

Initial experiments were conducted by McNamara and her colleagues [13] and *iSTART* [14] can be considered the first implemented system that addresses self-explanations [15]. It has various modules that explain the *SERT* method to the

students, one that shows them how to use those techniques using a virtual student, and another training module that asks students to read texts and give verbalizations, evaluates them and provides an appropriate feedback. *iSTART* divides verbalizations into four main categories: irrelevant, paraphrases, verbalizations that use knowledge previously found in the text and verbalizations which use external knowledge from the students' experience. It is easier to automatically identify paraphrases and irrelevant explanations, but it is more difficult to identify and evaluate verbalizations that contain information coming from students' experience [16].

We conducted an experiment [6] for analyzing the control and the regulation of comprehension through reading strategies. Pupils (3^{rd}–5^{th} grade, 8–11 years old) were given the task to read aloud two French stories and were asked at predefined moments to self-explain their impressions and thoughts about the reading materials. The self-explanations were coded according to McNamara's [4] scheme. The results of this study support the view that pupil's self-explanations are an adequate way to access to their reading strategies. The sole exception consists of prediction strategies, which were scarcely used in comparison to McNamara's participants, perhaps due to the age of the pupils. Initial and partial automated results based on the previous study were presented in [7], and we present in this paper data from a larger sample, using fine-tuned heuristics and an automatic classifier for predicting comprehension.

3 Reading Strategies Identification Heuristics

In terms of reading strategies, our aim was to create automated extraction methods designed to support tutors at identifying various strategies employed by pupils that are best aligned with the annotation categories: 1/ *paraphrasing*, 2/ *text-based inferences* consisting of *causality* and *bridging*, 3/ *knowledge-based inferences* or elaboration and 4/ monitoring or *control* [6]. A clear demarcation between causal inferences and bridging had to be established within our automated system due to underlying approaches and computational complexity, although causal inferences can be considered a particular case of bridging, as well as a reference resolution. In addition, we have tested various methods of identifying reading strategies and we will focus solely on presenting the refined heuristics that provided in the end the best overall human–machine correlations.

In ascending order of complexity, the simplest strategies to identify are *causality*, with markers like "*parce que*" (because), "*pour*" (for), "*donc*" (thus), "*alors*" (then), "*à cause de*" (because of) and *control*, with markers like "*je me souviens*" (I remember), "*je crois*" (I believe that), "*j'ai rien compris*" (I haven't understood anything) for which cue phrases based on pattern matching techniques have been used. As particular refinement for causality, all occurrences of the keywords at the beginning of a verbalization have been discarded because the strategy needs to create an inferential link between two adjacent textual segments, out of which the first is lacking since it is the beginning of a verbalization. In this particular case, the use of causality patterns indicates a lacunar pupil formulation frequently observed at their age. In terms of control, besides the verification of specific cue phrases, we added a

check to verify whether the pattern exists in the sentences within the original text, in which case we would be dealing with a paraphrase rather than a control statement.

As a second stage of complexity, *paraphrases*, that in the manual annotation were considered mere repetitions of the same semantic propositions by human raters, were automatically identified through lexical similarities. More specifically, words from the verbalization were considered as paraphrased words if they had identical lemmas or stems, or were synonyms extracted from lexicalized ontologies – *WordNet* [16] or *WOLF* [17] – with words from the initial text. Adjacent words from pupil's self-explanations, identified as paraphrased concepts were grouped into paraphrase segments in order to highlight contiguous zones highly referential to the initial text. In addition, if more than a predefined percentage of relevant words from a sentence from the initial text are paraphrased within the verbalization, that specific sentence is tagged as a paraphrasing segment. The previous percentage was empirically set after performing multiple iterations with incremental values, whereas relevant words are obtained after stop words elimination and after selecting solely dictionary words.

In the end, the strategies most difficult to identify are *knowledge inference* and *bridging*, for which semantic similarities have to be computed. An inferred concept is a non-paraphrased word for which the following three semantic distances were computed: the highest similarity to another word from the initial text (expressed in terms of semantic distances in ontologies, Latent Semantic Analysis and Latent Dirichlet Allocation) [7] and the relevance of both words to the textual fragments in-between consecutive self-explanations expressed as semantic cohesion. The latter distances had to be taken into consideration for better weighting the importance of each concept, with respect to the whole text. In the end, for classifying a word as inferred or not, a weighted sum of the previous semantic similarities is computed and compared to a minimum imposed threshold which was experimentally set at 0.4 for maximizing the precision of the knowledge inference mechanism.

As bridging consists of creating connections between different textual segments from the initial text, cohesion was measured between the verbalization and each sentence from the referenced reading material [7]. Semantic similarity was measured in-between the current verbalization and the two previous textual blocks from the initial text. In order to relate to the overall cohesion between the verbalizations and what was initially stated within the reading material, the imposed similarity threshold for tagging a sentence as being a bridged element uses a cohesion value that exceeds the mean plus standard deviation of all previous similarity measures performed on all self-explanations of a given pupil. Similarly to paraphrases and for best adapting to the manual annotation process, adjacent sentences from the initial text tagged as being bridged within the verbalization are grouped into a bridging segment. Moreover, if a sentence is considered to be a paraphrasing segment due to a high density of paraphrased words, that sentence is not taken into consideration while defining the final bridging segments. To better highlight the identification mechanisms, Fig. 1 depicts with bold bridged sentences from the initial text with verbalization 2 that exceed the identified threshold and that are not marked as paraphrases. In the end, four bridged segments are automatically determined: A3, B1 together with B2 due to adjacency within the same paragraph, C1 and C3 from the later textual block.

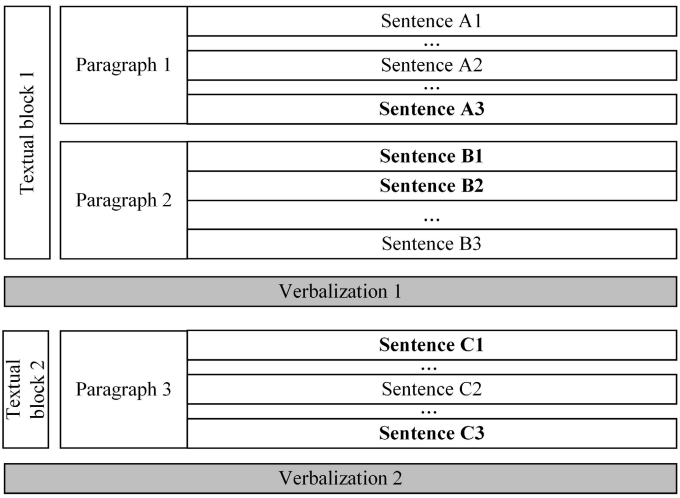


Fig. 1. Bridging identification

4 Combining Automatically Identified Reading Strategies for Predicting Comprehension

All the previous reading strategies and their corresponding identification heuristics can be viewed as attributes that describe the learner's comprehension level. In order to *predict* the comprehension level of each learner based on the used reading strategies, post-tests were administered to each pupil and comprehension scores were manually determined using these tests. Therefore, we found it appropriate to use a classifier that accepts as inputs the number of used reading strategies and predicts a comprehension class depicting the reader's understanding level expressed as a comprehension level class estimate.

Similar to the textual complexity problem for which Support Vector Machines (SVMs) [9] have been proven to be the most relevant [18], we trained multiple SVMs for determining the appropriate comprehension class. A one-versus-all approach implementing the winner-takes-all strategy is used to deal with the problem of multiple SVM returning 1 for a specific text (the classifier with the highest output function assigns the class). As specific optimizations, an RBF kernel with degree 3 was selected and a Grid Search method [19] was enforced to increase the effectiveness of the SVM through the parameter selection process for the Gaussian kernel. Exponentially growing sequences for C and γ were used, and each combination of parameter choices was checked using the testing corpora; in the end, the parameters that generated the best precision were selected.

5 Validations of the Identification Heuristics and of the Comprehension Prediction Model

We ran an experiment with 82 pupils with age between 8 and 11 years, uniformly distributed in terms of their age, who had each to read aloud two French stories of about 450 words (*The Cloud Swallower* and *Matilda*). During their lecture, pupils had to stop in-between at five, respectively six predefined markers, and explain what they understood up to that moment. Their explanations were first recorded and transcribed, then annotated by two human experts (PhD in linguistics and in psychology), and in the end categorized according to the imposed annotation scheme. Nevertheless, when looking at manual assessments, discrepancies between evaluators were identified due to different understandings and perceptions of pupil's intentions, expressed within their self-explanations; all disagreements were solved individually by mediation for each self-explanation. In addition, predefined rules and patterns were used to perform automatic cleaning in order to process the phonetic-like transcribed verbalizations.

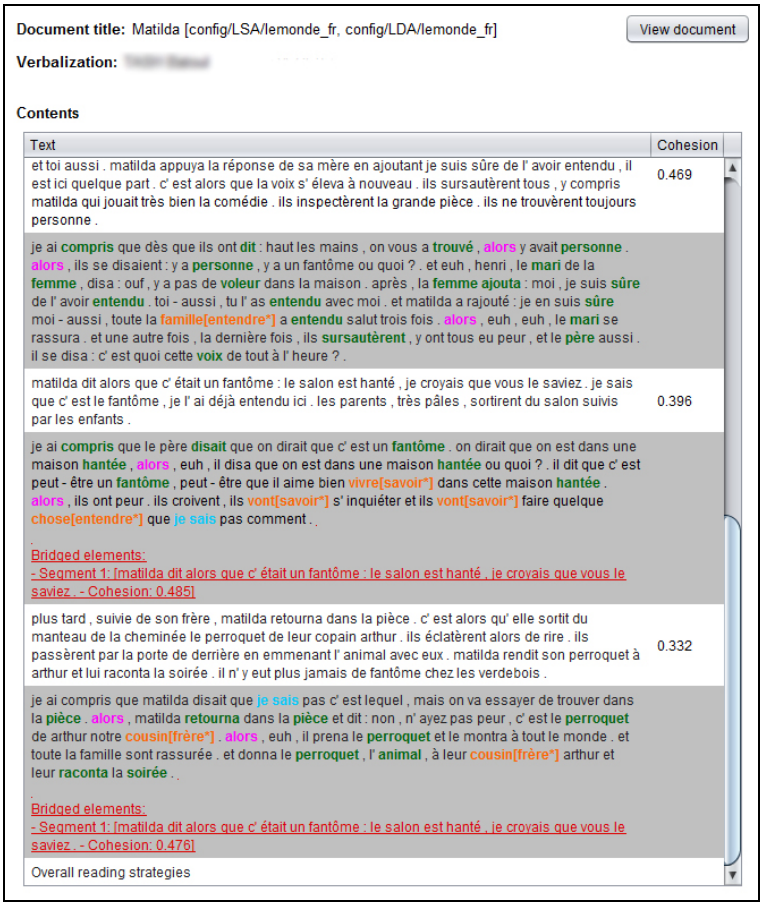


Fig. 2. Visualization of automatically identified reading strategies

Fig. 2 depicts the main interface of our developed system in which the grey sections represent the pupil's self-explanations, whereas the white blocks represent paragraphs from the read story. All strategies are highlighted within the self-explanation with a specific color encoding: control (cyan blue), causality (purple), paraphrasing (green), inferred concept [*] (yellow) and bridging (red) with a clear demarcation of the textual segments from the reading material comprising of inter-linked cohesive sentences. In addition, Fig. 2 also depicts in the last column the cohesion measures normalized in [0; 1] with previous paragraphs from the story.

Three variables were required for fine-tuning the higher-level reading strategies: bridging requires a *minimum semantic cohesion* ($Min_{coh\_bridging}$) and a *maximum percentage* of words for not considering a sentence as paraphrased ($Max_{paraphrase}$), while knowledge inference uses only a *minimum similarity threshold* (Min_{sim_KI}). Our system automatically determines the most suitable values for maximizing the overall Pearson correlations and *F1*-scores as measures of outputs' correctness with regards to the manual annotations (see Table 1). As expected, paraphrases, control and causality occurrences were much easier to identify than information coming from pupils' experience [20]. Moreover, our experiments demonstrate that although the variables for the two texts have similar optimal values, there are rather high fluctuations in the accuracy of the reading strategies' identification, therefore highlighting the specificities of each text and the intrinsic subjectivity of the analysis.

Table 1. Accuracy of the automatically identified reading strategies

Statistic measure	Paraphrasing	Text-based Inference (causality and bridging)	Knowledge-based Inference	Control
Text 1: *The Cloud Swallower*				
$Min_{coh\_bridging}$ = .40; $Max_{paraphrase}$ = 60%; Min_{sim_KI} = .33				
Pearson correlation	.64	.55	.41	.84
Precision	.64	.79	.50	.76
Recall	.99	.83	.94	.63
F1 score	.78	.81	.65	.68
Text 2: *Matilda*				
$Min_{coh\_bridging}$ = .45; $Max_{paraphrase}$ = 65%; Min_{sim_KI} = .33				
Pearson correlation	.56	.69	.48	.90
Precision	.73	.71	.34	.86
Recall	.99	.94	.97	.70
F1 score	.84	.81	.50	.77
All verbalizations together, from both texts				
$Min_{coh\_bridging}$ = .4; $Max_{paraphrase}$ = 65%; Min_{sim_KI} = .33				
Pearson correlation	.64	.60	.35	.89
Precision	.69	.74	.47	.83
Recall	.99	.90	.87	.68
F1 score	.81	.81	.61	.74

After fine-tuning the identification heuristics, we opted to create three comprehension classes for predicting the learner's comprehension level with a distribution of 30%, 40% and 30% of all pupil scores sorted in ascending order and to

apply 3-fold cross-validations for the SVM training process. The resulting average agreement between automatic predictions and the class assigned from the post-test scores was approximately .78 in most runs (see Table 2). Due to a rather limited corpus, the prediction accuracy oscillates between different training sessions, with a minimum of .66. We also noticed a rather small differentiation between the first and the second class, as well as conflicting instances of pupils with a high number of used reading strategies, but pertaining to opposite comprehension classes. The previous contradictions in terms of the number of used reading strategies in opposition to pupils' comprehension levels, corroborated with rather small differentiations between adjacent classes, led to a rather low prediction accuracy of the second class.

Table 2. Comprehension prediction based solely on the four automatically identified reading strategies

Verbalizations pertaining to	Agreement – Class 1 –	Agreement – Class 2 –	Agreement – Class 3 –	Average agreement
Text 1: *The Cloud Swallower*	1	.33	.67	.67
Text 2: *Matilda*	.67	.33	1	.67
Both texts	1	.33	1	.78

Nevertheless, results are encouraging based on the limited number of training instances, the reduced number of classification attributes and the fact that a lot of noise existed within the transcriptions. From this point, it becomes clear that external factors should be enforced in order to increase the accuracy of the prediction and to create a more comprehensive view, as the diversity and the richness of the strategies a reader carries out depend on many factors, either personal (proficiency, level of knowledge, motivation), or external (textual complexity).

In order to prove the feasibility of the previous statements, we added a simple factor already computed during the identification process: the average value of cohesion between each verbalization and the corresponding paragraphs from the initial text. This measure emphasizes the link between what was initially stated and the learner's understanding or personal perspective. As expected, the results from Table 3 highlight an increase in the overall prediction accuracy.

Table 3. Comprehension prediction based on the four heuristics plus the average cohesion value added as an attribute for classification

Verbalizations pertaining to	Agreement – class 1 –	Agreement – class 2 –	Agreement – class 3 –	Average agreement
Text 1: *The Cloud Swallower*	1	.33	1	.78
Text 2: *Matilda*	1	.67	.67	.78
Both texts	1	.67	1	.89

In the end, notable improvements in terms of the initial experiments presented in [7] can be observed: 1/ the use of 8 times more participants, each self-explaining two texts instead of only one; 2/ an important increase in the identification accuracy for

paraphrases, knowledge and text-based inferences; 3/ although bridging taken individually has still a low correlation which indicates that the human annotated bridging strategy is not aligned with the identification heuristics, the use of the new class of text-based inferences demonstrates that the integrated perspective of bridging and causality taken together is more cognitively relevant and representative with regards to the manual annotations; 4/ the use of Support Vector Machines for predicting the learner's comprehension level.

6 Conclusion and Future Research Directions

Our aim consists of supporting tutors and our approach emphasizes the benefits of a regularized and deterministic process of identification as a viable alternative to the subjectivity-laden task of manual annotation. Moreover, the performed validations confirm that reading strategies are related to the pupil's comprehension level, but also highlight the need to add more factors, potentially inspired from textual complexity measures [21, 22] or essay scoring techniques [23] in order to increase the accuracy of the predictions.

As the comprehension scores are not global, but related to the read texts subject to expressing one's meta-cognitions, we can state that reading strategies can be used to predict comprehension based on the overall experimental settings. Our next aim consists of deploying and using our system in classroom settings to analyze student's reading strategies and to infer possible comprehension problems in near realtime.

Acknowledgements. This research was partially supported by an *Agence Nationale de la Recherche* (DEVCOMP) grant and by the 264207 ERRIC–Empowering Romanian Research on Intelligent Information Technologies/FP7-REGPOT-2010-1 project. We would also like to thank Aurélie Nardy and Françoise Toffa who helped us to gather experimental data, and the teachers and pupils who participated in our experiments.

References

1. Millis, K., Magliano, J.P.: Assessing comprehension processes during reading. In: Sabatini, J.P., Albro, E.R., O'Reilly, T. (eds.) Assessing Reading in the 21st Century, pp. 35–53. Rowman & Littlefield Publishing, Lanham (2012)
2. McNamara, D.S., Magliano, J.P.: Self-explanation and metacognition. In: Hacher, J.D., Dunlosky, J., Graesser, A.C. (eds.) Handbook of Metacognition in Education, pp. 60–81. Erlbaum, Mahwah (2009)
3. McNamara, D.S., Scott, J.L.: Training reading strategies. In: 21th Annual Meeting of the Cognitive Science Society (CogSci 1999), pp. 387–392. Erlbaum, Hillsdale (1999)
4. McNamara, D.S.: SERT: Self-Explanation Reading Training. Discourse Processes 38, 1–30 (2004)
5. Nash-Ditzel, S.: Metacognitive Reading Strategies Can Improve Self-Regulation. Journal of College Reading and Learning 40(2), 45–63 (2010)

6. Nardy, A., Bianco, M., Toffa, F., Rémond, M., Dessus, P.: Contrôle et régulation de la compréhension: l'acquisition de stratégies de 8 à 11 ans. In: David, J., Royer, C. (eds.) L'apprentissage de la lecture, p. 16. Peter Lang, Bern-Paris (in press)
7. Dascalu, M., Dessus, P., Trausan-Matu, Ş., Bianco, M., Nardy, A.: ReaderBench, an Environment for Analyzing Text Complexity and Reading Strategies. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 379–388. Springer, Heidelberg (2013)
8. Dascalu, M.: Analyzing Discourse and Text Complexity for Learning and Collaborating, Studies in Computational Intelligence, vol. 534. Springer, Switzerland (2014)
9. Cortes, C., Vapnik, V.N.: Support-Vector Networks. Machine Learning 20(3), 273–297 (1995)
10. van Dijk, T.A., Kintsch, W.: Strategies of discourse comprehension. Academic Press, New York (1983)
11. Piech, C., Huang, J., Chen, Z., Do, C., Koller, D.: Tuned models of peer assessment in MOOCs. In: Int. Conf. Educational Data Mining (EDM 2013). International Educational Data Mining Society, Memphis (2013)
12. Goldin, I.M.: Acounting for peer reviewer bias with Bayesian models. In: The Proceedings of the Workshop on Intelligent Support for Learning Groups at the 11th Int. Conf. on Intelligent Tutoring Systems (ITS 2012), Chania, Grece (2012)
13. O'Reilly, T.P., Sinclair, G.P., McNamara, D.S.: iSTART: A Web-based Reading Strategy Intervention that Improves Students' Science Comprehension. In: CELDA 2004, p. 8. IADIS Press, Lisbon (2004)
14. McNamara, D.S., Boonthum, C., Levinstein, I.B.: Evaluating self-explanations in iSTART: Comparing word-based and LSA algorithms. In: Landauer, T.K., et al. (eds.) Handbook of Latent Semantic Analysis, pp. 227–241. Erlbaum, Mahwah (2007)
15. Jackson, G.T., Guess, R.H., McNamara, D.S.: Assessing cognitively complex strategy use in an untrained domain. In: 31st Annual Meeting of the Cognitive Science Society (CogSci 2009), pp. 2164–2169. Cognitive Science Society, Amsterdam (2009)
16. Miller, G.A.: WordNet: A Lexical Database for English. Communications of the ACM 38(11), 39–41 (1995)
17. Sagot, B.: WordNet Libre du Francais, WOLF (2008), http://alpage.inria.fr/~sagot/wolf.html
18. François, T., Miltsakaki, E.: Do NLP and machine learning improve traditional readability formulas? In: PITR2012, vol. 2012, pp. 49–57. ACL, Montreal (2012)
19. Bergstra, J., Bengio, Y.: Random Search for Hyper-Parameter Optimization. The Journal of Machine Learning Research 13, 281–305 (2012)
20. Graesser, A.C., Singer, M., Trabasso, T.: Constructing inferences during narrative text comprehension. Psychological Review 101(3), 371–395 (1994)
21. Graesser, A.C., McNamara, D.S., Kulikowich, J.: Coh-Metrix: Providing multilevel analyses of text characteristics. Educational Researcher 40(5), 223–234 (2011)
22. Nelson, J., Perfetti, C., Liben, D., Liben, M.: Measures of text difficulty: Testing their predictive value for grade levels and student performance. Council of Chief State School Officers, Washington, DC (2012)
23. Todd, R.W., Khongput, S., Darasawang, P.: Coherence, cohesion and comments on students' academic essays. Assessing Writing 12, 10–25 (2007)

Modeling Strategy Use in an Intelligent Tutoring System: Implications for Strategic Flexibility

Caitlin Tenison[1] and Christopher J. MacLellan[2]

[1] Psychology Department
Carnegie Mellon University Pittsburgh, PA 15213, USA
[2] Human-Computer Interaction Institute
Carnegie Mellon University Pittsburgh, PA 15213, USA
ctenison@andrew.cmu.edu, cmaclell@cs.cmu.edu

Abstract. Education research has identified strategic flexibility as an important aspect of math proficiency and learning. This aspect of student learning has been largely ignored by Intelligent Tutoring Systems (ITSs). In the current study, we demonstrate how Hidden Markov Modeling can be used to identify groups of students who use similar strategies during tutoring and relate these findings to a measure of strategic flexibility. We use these results to explore how strategy use is expressed in an ITS and consider how tutoring systems could integrate a measure of strategy use to improve learning.

1 Introduction

Strategic flexibility in arithmetic problem solving is both an important reflection of knowledge [1,2] and a recognized predictor for future learning [3,4]. The National Mathematics Advisory Panel [5] lists flexibility along with accuracy and speed of problem solving as the core defining features of a student's math proficiency. Despite the importance placed on flexible problem-solving, work on Intelligent Tutoring Systems (ITSs) has traditionally focused on the accurate completion of problem steps rather than on the strategies used by students to complete them. The current study identifies differences in strategy use within an ITS, relates these differences to a pencil and paper measure of strategic flexibility, and explores how ITSs may be designed to support strategic flexibility.

Strategic flexibility refers to a student's knowledge of multiple strategies and their ability to choose the best of those strategies for a given problem [6,4]. Measures of strategic flexibility correlate with both the student's procedural and conceptual knowledge [1,7,4]. Schneider et al. [2] found a bidirectional relationship between procedural and conceptual knowledge and hypothesized that these two types of knowledge improve strategic flexibility in an iterative fashion. Students who have high strategic flexibility are more likely to adapt their strategies, transferring their knowledge to solve new problems [8,4]. Conversely, students who lack strategic flexibility struggle to solve more difficult or unfamiliar problems that require the use of different strategies [6]. An active area of

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 466–475, 2014.

research concerns developing pedagogical methods to foster strategic flexibility and understanding the impact of strategic flexibility on problem solving.

Studies of ITSs have shown their effectiveness for developing students' procedural and conceptual knowledge in math problem solving [9,10,11]. Little research, however, has explored the impact of ITSs on strategic flexibility. Unlike the traditional studies of strategic flexibility, which use pencil and paper, ITSs confine students to working within the structure of the interface. This raises the question of if and how different strategies present themselves in rigid tutoring interfaces. One study focuses on the effects of allowing strategic flexibility within a tutor [12]. Measuring the number of times students made variations from the main strategy path, Waalkens et al. found that students did not take advantage of the flexibility permitted by the interface. Without instructing students to solve problems using multiple strategies, the likelihood of a student using a divergent strategy is low. Previous research on strategic flexibility has found that although students may be aware of many strategies, they often limit their choice to the most efficient one when problem solving [6]. Acknowledging these earlier findings, we identify two ways to improve research on strategic flexibility within ITSs. First, researchers must actively encourage the use of different strategies if they wish to observe student's strategic flexibility. Second, as we will discuss, researchers should use more sophisticated methods for modeling strategies and detecting how they are used. Introducing these changes to ITS research of strategic flexibility will make it possible for researchers to explore the effect of strategically focused interventions on how well students solve future problems.

Outside of the math domain there has been some work on developing more complex methods for assessing strategy from the choices students make in ITSs. Piech et al.[13] looked at differences in the paths that introductory programmers take when completing programming assignments and found that strategic differences in two homework assignments at the beginning of the semester predicted students' midterm grades. These sequences or paths can be seen as a reflection of the strategies that a student employs during problem solving. Additionally, the area of research on meta-cognitive hint seeking identifies the strategies students use when unable to solve problems [14,15]. Using a model of hint seeking, Roll et al. [16] found that recognizing and intervening when students are using bad hint seeking strategies improves learning. Work from these different areas demonstrates that strategy use can be identified within a tutoring system, and also suggests that these results can identify opportunities for tutoring.

1.1 The Current Study

The current study bridges research on strategic flexibility in mathematics and work identifying strategy use in ITSs. With evidence supporting the value of strategic flexibility in math, it is important that tutoring systems develop approaches for understanding and supporting strategic flexibility. In the current study, we use Hidden Markov Modeling (HMM) to cluster participants into strategically distinct groups. We present evidence supporting the hypothesis that these groups differ on a measure of strategic flexibility collected using the pencil

and paper test developed by Rittle-Johnson and Star [7]. Furthermore, we use this HMM method to explore how flexibility presents itself in a math ITS as differences in tutor behavior. We conclude with recommendations concerning how ITSs can be built to measure and encourage flexible strategy use.

2 Materials and Methods

We used an observational design in which all participants completed the same tutor curriculum. Students completed a 20-minute pencil and paper math test to assess proficiency with algebraic problem solving and strategic flexibility. A week after taking this test, students spent an hour and a half in the school computer lab working with an algebra ITS.

2.1 Participants

There were 112 eighth and ninth grade Algebra I students (72 eigth grade; 57 females; mean age 13.6, SD 1.2) who took a math test and participated in the tutoring. There were seven eighth grade classes (3 advanced and 4 regular) and four ninth grade classes (2 regular and 2 remedial). All students attended the same large, urban public school. The school consisted of 60.6% Caucasian, 33.5% African American, and 1.3% Asian. Approximately 52% of students qualified for free or reduced lunch.

There were four teachers whose classes participated in this study. Each teacher taught the same grade and advancement level to students in their class. All four teachers used the same algebra curricula, which was supplemented with work on the ALEKS online math program. All classes had previously covered the distributive property and solving multi-step equations. Human subjects' approval and consent from the school was obtained prior to conducting the study.

2.2 Materials

Assessment. We used a modified version of the Rittle-Johnson and Star [7] assessment, which assesses mathematical knowledge (both conceptual and procedural) and strategic flexiblity for one- and two-step algebra equation problem

Table 1. An example of the two strategies for solving the two problem types. Both strategies are correct, but the green strategies are those biased by the tutor.

Divide Problem	Divide Problem	Multiply Problem	Multiply Problem
Distribute Strategy	Both Sides Strategy	Distribute Strategy	Both Sides Strategy
$2(3x+5)=6$	$2(3x+5)=6$	$\frac{(8y-4)}{2}=6$	$\frac{(8y-4)}{2}=6$
$2*3x+2*5=6$	$\frac{2(3x+5)}{2}=\frac{6}{2}$	$\frac{(8y)}{2}-\frac{4}{2}=6$	$2*\frac{(8y-4)}{2}=6*2$

solving. We implemented only the procedural skill and strategic flexibility portion of the assessment. The test of strategic flexibility assessed three features; the ability to generate, recognize, and evaluate multiple strategies. Although we followed the organization of this assessment, we modified many of the math problems tested in this assessment in order to better accommodate an older student population and focus more directly on the problems being studied within the tutoring system. The student's procedural accuracy was calculated by percentage of problems correctly solved.

Problem Types. Work on strategic flexibility has found that while students may be aware of many strategies they will often learn to use the most efficient strategy for a specific problem [4,6]. To encourage the use of different strategies we used 6 variations of the linear equation. Students saw 6 examples of each type of problem. For this study we collapsed the 6 problem types into two categories; "divide problems," in which the problem can be solved by dividing both sides by a coefficient, and "multiply problems," in which the problem can be solved by multiplying both sides by a coefficient (Table 1). While both of these problem types can be solved using either of two correct strategies, "distribute" or "both sides", we took several actions to bias strategy use. First, the tutor hints recommended different strategies for the two problem types. Table 1 displays the recommended strategies in green. Second, distribute required fewer tutor actions than the both sides strategy for divide problems, while the both strategies required the same number of actions for multiply problems. Finally, if students chose their strategy to avoid large fractions, this would promote the use of the multiply strategy for the multiply problems. This last decision-making heuristic applies more for problems that require student calculation.

Intelligent Tutoring System. We used a modified version of Cognitive Tutor [11]. The tutoring interface directs students to select the step the computer should take to solve the problem. Students can choose between actions to "transform" or "solve" the problem. The transform command directed the computer to take actions to change the structure of the equation. Table 1 shows how the transformation of "distribute," "multiply both sides" or "divide both sides" would change the equation. Solve actions instruct the computer to perform various calculations, such as combining like terms. This is an important feature of the tutor because students must indicate each step taken to solve the problem rather than combining steps in their calculations.

2.3 Data Analysis

With strategic flexibility so closely related to procedural skill [1,2], the inclusion of off-task strategy paths would bias our clustering method to cluster based on students' use of off-task strategies. Because we are interested if the correct strategies students learn to use are related to their strategic flexibility we restrict our analysis to on-task actions only. For each student we recorded the choices made when faced with the problems shown in Table 1.

We based our analysis on the HMM clustering method described by Smyth [17] and Piech et al. [13]. Detailed justification of this method can be found in these two papers, so we will only summarize the steps of the analysis. Using the on-task student actions, we fit an HMM model to each student's action sequence; the model had a hidden state for every possible action, where each state had 100% probability of emitting the action represented by that state (this is essentially a Markov Chain). After fitting the model to a student's actions, we modified the transition probabilities between hidden states so that none of the transition probabilities were equal to zero. We did this by replacing all zero probability transitions with a small constant (1×10^{-10}) and renormalizing the transition probabilities. Next, we calculated the log probability that this model fit each of the other student's action sequences individually. After doing this for all students, we calculated the distance between two subjects sequences as the average of the log probability of one subject's model predicting the other subject's data with the log probability of the other subject's model predicting the first subject's data. Finally, we used these pairwise distances to cluster the students. We used the k-mediods clustering algorithm [18] to ensure clusters of similar size and to deemphasize the effect of outliers. We determined the number of clusters by fitting models with 2 through 10 clusters and evaluating the log probability with leave-one-subject-out cross-validation. We found four clusters best fit the data. The parameters of these 4 clusters were next used to initialize a composite HMM model, which had additional latent states for each cluster that transitioned only to their respective smaller HMMs. After initializing this new HMM using the individually computed transition probabilities, we retrained the composite HMM using all the data. The resulting estimates are considered better than those generated by separately training the model on the four smaller HMMs. This method is particularly useful for building descriptive rather than predictive models of the data [17] and is thus useful for understanding the common strategy choices displayed during tutoring.

3 Results

3.1 Flexibility Score

Our study replicates the results of Rittle-Johnson and Star [1]. We found a significant correlation between strategic flexibility and procedural knowledge $r(110) = 0.73$, $p < 0.001$.

3.2 HMM Clustering

We clustered students based on their correct problem solving paths using the method described in section 2.3. We best fit a four cluster model. We found a significant difference in the average strategic flexibility scores of students in the different clusters, $f(3, 107) = 10.1$, $p < 0.001$. Figure 1 shows the mean and standard errors for the four clusters. Post hoc comparisons using the Tukey HSD test indicated that the only significant ($p < 0.05$) differences were between the 2nd and 3rd cluster and the 3rd and 4th cluster.

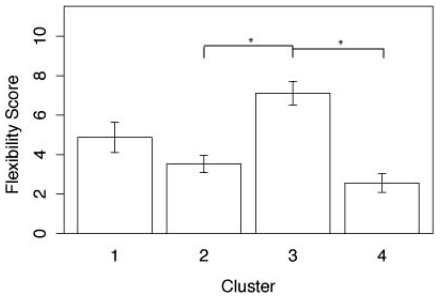


Fig. 1. The mean flexibility of the four identified clusters with standard error

3.3 Exploring Strategic Flexibility in Tutor Behavior

We used the transition probabilities generated from the HMM training to explore the different strategies learned. The four clusters of students showed distinct patterns of strategy use. Figure 2 shows the predominate strategies used by the four groups. In cluster 2 ($n = 25$) and cluster 3 ($n = 54$), students alternate between the tutored strategies. In cluster 1 ($n = 16$) and cluster 4 ($n = 16$), on the other hand, students use the same strategy for both problem types. Students in cluster 1 distribute on both problems, whereas students in cluster 4 apply either divide or multiply to both sides of the equation in order to eliminate the coefficient.

We next wanted to test if the strategies presented by each group of students were learned over the course of tutoring. To see if students increased their use of either of the dominant strategies of each cluster (displayed in Figure 2) we fit each cluster's data to an additive factor model (AFM) [19]. For this paper we are interested in only using this model descriptively to observe if students are increasing their use of the dominant strategies. We found that as students gained practice with the problems they increased their use of the dominant strategies, these slopes are displayed in Table 2.

Table 2. The coefficient for the learning rate of each cluster's dominant strategy by problem type, as computed by the Additive Factors Model

Cluster	Divide Problems	Multiply Problems
1	.21	.05
2	.05	.04
3	.13	.02
4	.14	.02

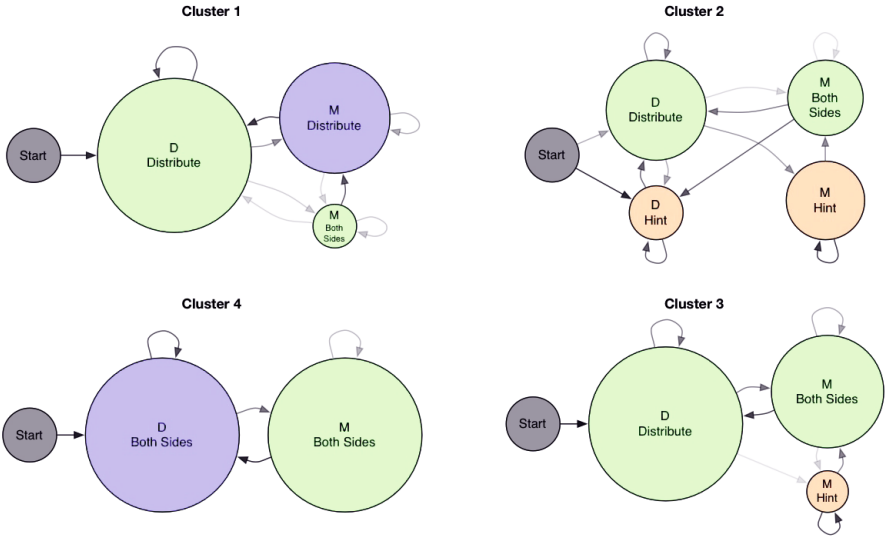


Fig. 2. The student behavior for each cluster. The Divide (D) and Multiply (M) labels represent the problem type and the Distribute and Both Sides labels denote the strategy taken. The node size denotes the number of times it was visited. Green nodes are strategies that were suggested by the tutor, blue nodes are (valid) untaught strategies, and orange nodes are hints. Arrow gradients denote transition probability.

4 Discussion

The ability to flexibly solve math problems is a valued measure of proficiency and important to future learning [5,8]. Few studies have investigated how strategic flexibility is displayed in ITSs, and perhaps as a result no studies have directly measured math flexibility using an ITS. The current study uses a method previously employed in a different domain [13] to identify groups of students in a math ITS that differ in how they apply strategies to solving equations.

First, our findings from the behavioral pre-test replicated results from Rittle-Johnson and Star [1,7], showing a correlation between strategic flexibility and procedural knowledge. Understanding how strategic flexibility is represented during tutoring is less clear. The Rittle-Johnson and Star measure of strategic flexibility directs students to generate multiple solution paths; however, in a tutoring setting students are not prompted to choose multiple strategies. Studies of strategic flexibility have observed that when students are not asked to generate multiple strategies they will often use only one strategy to solve problems [4,6]. This makes it difficult to observe flexible strategy use indirectly. To combat this challenge, our study used two sets of problems, which despite being solvable by the same two methods, were set up to favor different methods. As previous research would suggest, students remain relatively consistent within problem types; however, between problem types students changed strategies. When students appropriately adjust

their strategy this suggests that these students are more strategically flexible and would be better able to adapt and transfer their knowledge to similar, but novel problems– testing this hypothesis is one possible direction for future work.

In the current study we used a variation of the HMM clustering method used by Smyth [17] and Piech et al. [13] to cluster participants according to their correct action sequences. This clustering method, while conducted without information about a student's strategic flexibility measure, distinguished groups that significantly differed in strategic flexibility. This distinction suggests that the strategies students choose are reliant on their strategic flexibility. The lack of a significant difference between clusters 1 and 3 suggests that students may be less likely to use the optimal strategy on multiply problems. This can be explained by the lack of a direct benefit for using the both sides strategy on the multiply problems, as discussed in section 2.2. We ran this study to develop a method for identifying strategy use in math problem solving and while experimental manipulations must be done to learn more about the underlying causes of strategic flexibility in the tutor, the exploratory analysis from our work sets forth multiple areas of potential research.

The HMM clustering provides some insight into how the strategic flexibility scores translate into strategy use during tutoring. Students in cluster 3 show flexibility in their ability to switch between the distribute and the both sides strategy. This is echoed in their high strategic flexibility scores. Students in cluster 2 show a similar pattern, however their strategy path also shows that these students are reliant on the hints that direct them towards these strategies. The positive slopes from the AFM, indicate that while these students are learning to apply these strategies more often over the course of tutoring, this is at a slow rate. Students in cluster 1 generally use the distributive strategy to solve both problem types. Although there is some use of the both sides strategy, users of this strategy are seen returning to the distribute strategy. The positive learning gains reported from the AFM indicate that over the course of tutoring students become more rigid in their use of the distribute strategy on both problems. Students in cluster 4 use the both sides strategy to solve these problems and increase in their use of this strategy over tutoring. The behavior of cluster 4 scoring the lowest on the strategic flexibility and only using the both sides' strategy suggests that these students may not recognize distribute' as a potential strategy for solving these problems and could benefit most from an intervention.

As an observational study, we are limited in the causal claims we can make about the relationship between strategic flexibility and strategy use in the tutor. However, the methods used in this study can be applied in future experiments. First, this study demonstrates a means of observing strategy use by designing problems to specifically favor some strategies over others. Future studies could use a model of students' decision-making heuristics (i.e., a model of how they decide between possible next actions, such as avoiding fractions) to identify problems that favor different strategies. These problems could be used to triangulate the strategies that students know and do not know.

The current study used a version of the tutor in which students had to explicitly select each transformation of the problem. This tutoring format lends itself well to the method of strategy detection that we use, however, it may not apply to other tutors. Future work should extend the model to contexts in which students have more freedom in making multiple transformations in a single step, such as is seen in Waalkens et al. [12]. We expect different interfaces may foster strategic flexibility to different levels.

Two questions of great interest remain present in the field: How best can strategic flexibility be improved and to what extent does that improvement influence learning? While these questions are outside the scope of the current study, the ability to identify students based on their strategy use can establish the effects of an experimental intervention on strategy use and help identify individual differences in response to intervention. Paired with AFM, this strategy clustering method can identify the strategies students are using and if these strategies are increasing in use. Just as work by Yudelson and Koedinger [20] has demonstrated learning gains with an improved model of procedural skills, so should researchers investigate how modeling strategic flexibility impacts learning.

In conclusion, we investigated the relationship between strategic flexibility and the strategies students used in an ITS using HMM clustering. We discovered that students could be clustered into four distinct strategy groups which differed on their average flexibility scores. A closer look at the strategies used by the four groups showed that students converged in their use of these strategies over the course of tutoring. An exploration of the different strategies suggested multiple explanations for students' strategy use including learning from tutor hints, gaps in knowledge, and decision making heuristics favoring different strategies. This study is an first step in integrating models of strategic flexibility into ITSs.

Acknowledgments. This work was supported in part by a Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (#R305B090023) and by the Pittsburgh Science of Learning Center (NSF #SBE-0836012). We would like to thank Carnegie Learning, Inc., for providing the Cognitive Tutor data supporting this analysis. All opinions expressed in this article are those of the authors and do not necessarily reflect the position of the sponsoring agency.

References

1. Rittle-Johnson, B., Star, J.R.: Does comparing solution methods facilitate conceptual and procedural knowledge? An experimental study on learning to solve equations. Journal of Educational Psychology 99(3), 561–574 (2007)
2. Schneider, M., Rittle-Johnson, B., Star, J.R.: Relations among conceptual knowledge, procedural knowledge, and procedural flexibility in two samples differing in prior knowledge. Developmental Psychology 47(6), 1525–1538 (2011)
3. Alibali, M., Goldin-Meadow, S.: Gesture-Speech Mismatch and Mechanisms of Learning: What the Hands Reveal about a Child's State of Mind. Cognitive Psychology 25, 468–523 (1993)

4. Blöte, A.W., Van der Burg, E., Klein, A.S.: Students' flexibility in solving two-digit addition and subtraction problems: Instruction effects. Journal of Educational Psychology 93(3), 627 (2001)
5. National Mathematics Advisory Panel: The Final Report of the National Mathematics Advisory Panel. Technical report, U.S. Department of Education (2008)
6. Newton, K.J., Star, J.R., Lynch, K.: Understanding the Development of Flexibility in Struggling Algebra Students. Mathematical Thinking and Learning 12(4), 282–305 (2010)
7. Rittle-Johnson, B., Star, J.R.: Compared with what? The effects of different comparisons on conceptual knowledge and procedural flexibility for equation solving. Journal of Educational Psychology 101(3), 529–544 (2009)
8. Carpenter, T.P., Franke, M.L., Jacobs, V.R., Fennema, E., Empson, S.B.: A Longitudinal Study of Invention and Understanding in Children's Multidigit Addition and Subtraction. Journal for Research in Mathematics Education 29, 3–20 (1998)
9. Beal, C.R., Walles, R., Arroyo, I., Woolf, B.P.: On-line Tutoring for Math Achievement Testing: A Controlled Evaluation. Journal of Interactive Online Learning 6, 1–13 (2007)
10. Koedinger, K.R., Anderson, J.R.: Intelligent Tutoring Goes To School in the Big City. International journal of Artificial Intelligence in Education 8, 1–14 (1997)
11. Ritter, S., Anderson, J.R., Koedinger, K.R., Corbett, A.T.: Cognitive Tutor: Applied research in mathematics education. Psychonomic Bulletin & Review 14(2), 249–255 (2007)
12. Waalkens, M., Aleven, V., Taatgen, N.: Computers & Education. Computers & Education 60(1), 159–171 (2013)
13. Piech, C., Sahami, M., Koller, D., Cooper, S., Blikstein, P.: Modeling How Students Learn to Program. In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, pp. 153–160. ACM (2012)
14. Baker, R.S., Corbett, A.T., Koedinger, K.R.: Detecting student misuse of intelligent tutoring systems. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 531–540. Springer, Heidelberg (2004)
15. Shih, B., Koedinger, K.R., Scheines, R.: A response time model for bottom-out hints as worked examples. Handbook of Educational Data Mining, 201–212 (2011)
16. Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R.: Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. Learning and Instruction 21(2), 267–280 (2011)
17. Smyth, P.: Clustering sequences with hidden Markov models. In: Mozer, M.C., Jordan, M.I., Petsche, T. (eds.) Advances in neural information processing systems, pp. 648–654. Citeseer (1997)
18. Kaufman, L., Rousseeuw, P.J.: Finding groups in data: an introduction to cluster analysis. Wiley Series in Probability and Statistics, vol. 344. John Wiley and Sons, Inc., New Jersey (1990)
19. Cen, H.: Generalized Learning Factors Analysis: Improving cognitive Models with Machine Learning. PhD thesis, Carnegie Mellon University (2009)
20. Yudelson, M.V., Koedinger, K.R.: Estimating the benefits of student model improvements on a substantive scale. In: D'Mello, S.K., Calvo, R.A., Olney, A. (eds.) Educational Data Mining, Memphis, TN (2013)

Assessing Student Performance in a Computational-Thinking Based Science Learning Environment

Satabdi Basu, John S. Kinnebrew, and Gautam Biswas

Institute for Software Integrated Systems, Vanderbilt University, TN, USA
{satabdi.basu,john.s.kinnebrew,gautam.biswas}@vanderbilt.edu

Abstract. Computational Thinking (CT) can effectively promote science learning, but K-12 curricula lack efforts to integrate CT with science. In this paper, we present a generic CT assessment scheme and propose metrics for evaluating correctness of computational and domain-specific constructs in computational models that students construct in CTSiM – a learning environment that combines CT with middle school science. We report a teacher-led, multi-domain classroom study using CTSiM and use our metrics to study how students' model evolution relates to their pre-post learning gains. Our results lay the framework for online evaluation and scaffolding of students in CTSiM.

Keywords: Computational Thinking, Science education, CT Assessments, Computational Modeling, Agent-based modeling and Simulations, Scaffolding.

1 Introduction

Computational Thinking (CT) encompasses the representational practices and behaviors involved in formulating and solving problems and designing systems by drawing on computer science concepts like abstraction, decomposition, recursion, and simulation [6]. CT can play an important role in K-12 STEM education because computational modeling is an effective approach for learning challenging science and math concepts [3, 6]. Despite these known synergies between CT and science education, efforts to integrate them in the K-12 curricula and develop relevant CT-based assessments are lacking [2].

Several CT-based environments focus on domain-independent game design activities, and assessments typically measure use of different computational constructs over time [2, 4]. Frequent use of CT constructs is favored, but their effects on final artifacts (e.g., games designed) and the relation between final artifacts and pre-defined learning goals are rarely considered. Some interventions also include system-dependent post-assessments, which hinder generalization and make learning gains hard to ascertain [5]. The few efforts to integrate CT and science learning have primarily used external pre-post assessments to measure changes in students' attitude or awareness about CT, rather than proficiency in CT skills and science concepts [3].

In this paper, we present initial steps toward a more systematic assessment of CT-based science learning. We present a recent 6th-grade classroom study with

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 476–481, 2014.

CTSiM – a CT-based science learning environment. Pre- and post-tests assess gains in students' science and CT knowledge. We also evaluate students' computational models and their model evolution trajectories with respect to an 'expert' model, and then investigate their relationship to pre-post learning gains across different modeling activities.

2 The CTSiM Learning Environment and Learning Activities

In CTSiM, students first construct a conceptual model and then design a corresponding computational model for a given science phenomena. CTSiM employs an agent-based-modeling approach. Conceptual modeling involves identifying the relevant agents with appropriate properties and behaviors described as *sense-act* processes that capture the properties sensed and acted upon by the behaviors. For example, a fish agent's 'feed' behavior senses the fish's 'hunger' property and acts upon its 'energy' property. The student models *how* this behavior is enacted by constructing a computational model in the 'Construction world' (see Fig. 1) by selecting from a library of visual primitives that includes domain-specific (e.g., 'speed') and domain-general (e.g., conditionals and loops) primitives. The relevant domain-specific primitives for a behavior are made available only if the student has correctly conceptualized the relevant properties of the sense-act model of the behavior.

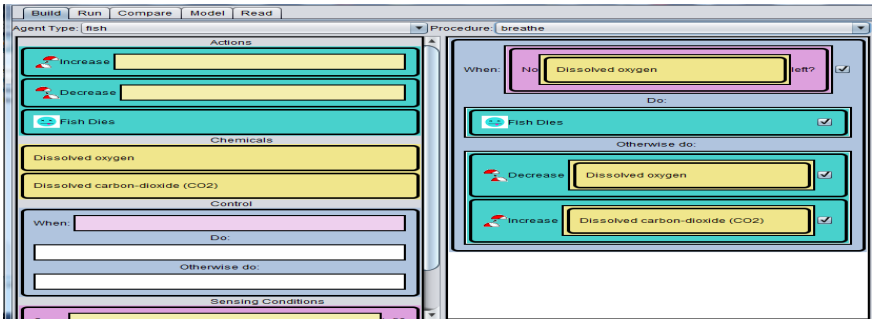


Fig. 1. The Construction world with a 'fish-breathe' procedure in a fish-tank unit

Students can simulate their computational models and use step-by-step highlighting to trace the model execution. Students can also verify the correctness of their models in the 'Envisionment world' using a side-by-side comparison of their model behaviors with an 'expert' simulation (see Figure 2). Identifying differences helps students refine and correct their conceptual and computational models. Reference domain information is also provided through hypermedia resources.

Currently, CTSiM comprises four primary modeling activities. (*Activity 1*): Students generate algorithms to draw simple shapes to explore the relations among acceleration, speed, and distance. They start by modeling shapes like squares with equal-length segments, implying constant speed. Then, they modify their algorithms

to generate spirals, where each line segment is longer (or shorter) than the previous one, to model acceleration; (*Activity 2*): Students model a rollercoaster car as it traverses segments of a track: (1) up (pulled by a motor) at constant speed, (2) down (accelerating), (3) flat (cruising), and (4) up again (decelerating). An expert simulation helps students understand "*correct*" system behavior and build models to match these behavior; (*Activity 3*): Students model part of a closed fish tank system - a macro-level semi-stable model involving the food chain, respiration, and reproduction processes of fish and duckweed, and the macro-level elements of the waste cycle. The non-sustainability of the model (the fish and the duckweed gradually die off) encourages students to reflect on the probable cause (toxicity from increasing fish waste), prompting the transition to Activity 4; (*Activity 4*): Students introduce Nitrosomonas and Nitrobacter bacteria to model the waste cycle, which convert the ammonia in the toxic fish waste to nutrients (nitrates) for the duckweed. The simulations with plots of chemical concentrations helps students understand the interdependence and balance among the agents in the fish tank ecosystem.

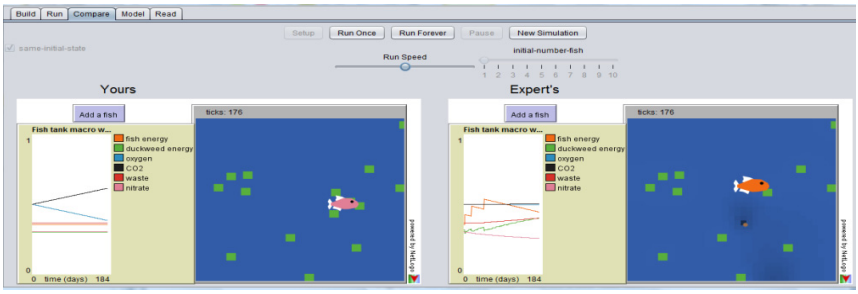


Fig. 2. The Envisionment world for a fish-tank unit

3 Method

We conducted a 2-week classroom study with 25 6th-grade, middle Tennessee students. The study was run daily during the 50-minute science period and was led by the science teacher, who had no significant prior experience with programming and was introduced to CTSiM during two 90-minute training sessions before the study. During the intervention, he alternated between teaching using CTSiM and having the students work individually to build their models using CTSiM. On Day 1, students took pre-tests for both the units. They worked on Modeling Activity 1 from days 2-4, and Activity 2 on days 5 and 6, then took the Kinematics post-test on day 7. Students then worked on the Ecology unit Activity 3 from days 8-10, and Activity 4 on days 11and 12. All students took the Ecology post-test on day 13. Student actions on the CTSiM system were continually logged as events for subsequent analysis.

We designed pre-post assessments to measure both science content and CT skills. The Kinematics domain questions tested the concepts of acceleration, speed, and distance and their relations, including the generation and interpretation of speed-time graphs. Ecology domain questions focused on students' understanding of the role of the species in a fish-tank ecosystem and their interdependence. CT skills were

assessed by asking students to construct algorithms for scenarios using computational and scenario-specific constructs primitives specified in the questions. This tested students' abilities to interpret given abstractions to generate meaningful algorithms, and their understanding of programming constructs like conditionals, loops, and variables.

The computational models that students generated for each activity were evaluated by comparing them against the expert model for that activity. We developed a *vector-distance model accuracy metric* [1] for measuring the difference between a student's model and the expert model; a distance of 0 implying a perfect match, i.e., the student's model contained all the primitives in the expert model and no extraneous primitives. The distance measure is based on the bag-of-words metric with each agent-procedure represented by the set of primitives they contain. Equation 1 defines our *correctness* measure as a fraction of the expert primitives in the student model.

The *incorrectness* measure captures extraneous primitives used in the student models (Equation 2). The *vector distance (accuracy) metric* (Equation 3) combines the (correctness, incorrectness) measures, calculating the model's vector distance to the expert model represented as the point (1,0). By labeling primitives as computational (e.g., 'repeat') or domain (e.g. 'speed'), we calculated separate computational and domain vector distances. We applied this metric to evaluate all but the Activity 1 shapes, which did not have one particular correct expert model.

$$Correctness = \frac{\sum_{each\ procedure} |user \cap expert|}{\sum_{each\ procedure} |expert|} \quad (1)$$

$$Incorrectness = \frac{\sum_{each\ procedure} (|user| - |user \cap expert|)}{\sum_{each\ procedure} |expert|} \quad (2)$$

$$Distance\ (Accuracy) = \sqrt{incorrectness^2 + (correctness - 1)^2} \quad (3)$$

4 Results

We report results for 22 out of the 25 students who participated in the study because one student was absent for the Ecology post-test and two others were outliers in terms of their inactivity (low number of edits to their models (≤ 5)) in at least one or the activities. The intervention produced significant learning gains overall, and for both domains and CT skills, measured separately [1]. All learning gains were significant at the $p<0.0001$ level and effect sizes were high (in the range of 0.4-0.7).

We also studied the modeling performance across activities, and found that on the average, students made more edits to their fish-macro models [mean = 146.5 (sd = 64.8)] than the fish-micro models [85.36(29.8)], and made the least number of edits in the Rollercoaster (RC) activity [49.9(16.24)]. The significant jump in the number of edits from RC to the fish-macro unit ($p<0.0001$) was expected because of the increased complexity and size of the expert model. We also calculated an edit '*effectiveness*' measure. This was the proportion of a student's model edits that improved the model accuracy. Overall, the effectiveness of students' edits decreased significantly ($p<0.0001$) from the RC [.7(.1)] to the macro activity [.58(.09)], and then increased significantly ($p<0.0001$) in the micro activity [.7(.09)]. A similar trend can

be seen for the final models where the final model accuracy decreased from the RC [.24(.16)] to the fish-macro model [.5(.27)] and then increased for the micro model [.3(.27)]. The trend persists for both domain and computational aspects of modeling performance.

We hypothesized that the changes in edit-effectiveness and model-accuracy across activities could be linked to the challenges students faced in the corresponding activities. For example, the RC activity involved modeling a single agent and a single procedure. However, the fish-macro activity introduced new domain content, multiple agents and multiple procedures for each agent. In a previous study, we coded student activity videos for the number of challenges faced, which confirmed the increase in challenges from the RC to the fish-macro activity and the decrease from the macro to micro activity [1]. While no quantitative conclusions can be drawn across the two studies, the basic modeling activities were the same, so we believe that the changes in the *vector-distance* and *effectiveness* metrics are likely the result of a corresponding change in the number of challenges faced across the activities.

Table 1. Edit effectiveness predicts final model accuracy

Correlations	Rollercoaster	Fish-tank macro	Fish-tank micro
Overall	R=-0.73, p<0.0005	R=-0.58, p<0.005	R=-0.69, p<0.0005
Domain	R=-0.45, p<0.05	R=-0.53, p<0.05	R=-0.48, p<0.05
CT	R=-0.76, p<0.0001	R=-0.71, p<0.0005	R=-0.82, p<0.0001

Table 1 shows, not surprisingly, that effectiveness of students' edits is a strong predictor of their final model accuracies. However, effectiveness was not a good predictor of pre-post gains. All correlations were below 0.3 and were not significant at the $p<0.05$ level. This generally agrees with previous results in which students' final model accuracy was not predictive of learning gains, except in the fish-tank micro unit, where fewer challenges were experienced by the students [1].

Table 2. Effects of different model-edit consistencies on final model and pre-post gains

	Final model distance			Pre-post gains		
	Less consistent	Highly consistent	t-test	Less consistent	Highly consistent	t-test
Roller-coaster	.29(.1), n=11	.2(.2), n=11	t=1.33, p>0.05	14.4(23.4), n=11	28.0(10.6), n=11	t=1.76, p>0.05
Fish-tank macro	.66(.2), n=8	.41(.3), n=14	t=2.21, p<0.05	46.1(.3), n=8	34.4(.21), n=14	t=1.21, p>0.05
Fish-tank micro	.56(.2), n=6	.20(.2), n=16	t=3.45, p<0.005	21.1(.1), n=6	44.2(.22), n=16	t=2.45, p<0.05

Qualitative examination of students' model evolutions indicated variations in modeling consistency, so we implemented a measure of *edit consistency in model improvement* as the coefficient of determination (R^2) from a linear regression on a student's model accuracy over time. We split students into two groups by their edit

consistency (above or below the median consistency value of 0.9) across all activities. We then compared the final model distances and pre-post gains across these '*high consistency*' and '*low-consistency*' groups (see Table 2). Final model accuracy and pre-post gains are generally higher in the *high-consistency* group, although the fish-macro activity deviates from this trend. This may be because the Ecology post-test was taken after the fish-micro activity and effects of edit-consistency in the macro unit became secondary to that of the micro unit.

5 Discussion and Conclusions

This paper studies student performance in a synergistic CT-based science learning environment based on students' pre-post tests, their computational models, and their model evolution across units. We designed system-independent pre-post assessments for science and CT and developed *vector-distance, effectiveness,* and *consistency* measures to characterize student models. Using these assessments, we show that students gained significantly on both science and CT content in Kinematics and Ecology. When the modeling activities were less complex, students' model edits were more effective and consistent, and their final models were more accurate. Students with more effective edits tended to have more accurate final models, but effectiveness was a weak predictor of learning gains. Students with more consistent edits also had more accurate final models, but they were also likely to have high learning gains for most activities. We believe that our vector-distance metric can help with online evaluation of students' models, providing opportunities for scaffolding and guidance in CTSiM.

Acknowledgements. This work was supported by NSF Cyberlearning (#1237350).

References

1. Basu, S., Dukeman, A., Kinnebrew, J., Biswas, G., Sengupta, P.: Investigating student generated computational models of science. In: Proceedings of the 11th International Conference of the Learning Sciences, Boulder, CO, USA (2014)
2. Grover, S., Pea, R.: Computational Thinking in K–12: A Review of the State of the Field. Educational Researcher 42(1), 38–43 (2013)
3. Hambrusch, S., Hoffmann, C., Korb, J.T., Haugan, M., Hosking, A.L.: A multidisciplinary approach towards computational thinking for science majors. In: Proceedings of the 40th ACM Technical Symposium on Computer Science Education (SIGCSE 2009), pp. 183–187. ACM, New York (2009)
4. Sengupta, P., Kinnebrew, J.S., Basu, S., Biswas, G., Clark, D.: Integrating Computational Thinking with K-12 Science Education Using Agent-based Computation: A Theoretical Framework. Education and Information Technologies 18(2), 351–380 (2013)
5. Werner, L., Denner, J., Campe, S., Kawamoto, D.C.: The Fairy Performance Assessment: Measuring computational thinking in middle school. In: Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, pp. 215–220. ACM (2012)
6. Wing, J.M.: Computational Thinking: What and Why? Link Magazine (2010)

A Student Model for Teaching Natural Deduction Based on a Prover That Mimics Student Reasoning

João Carlos Gluz[1], Fabiane Penteado[1], Marcel Mossmann[1], Lucas Gomes[1], and Rosa Vicari[2]

[1] Post-Graduation Program in Applied Computer Science (PIPCA) – UNISINOS – Brazil
jcgluz@unisinos.br, {fabiane.penteado,maca871, lucas.gomes.canoas}@gmail.com
[2] Interdisciplinary Center for Educational Technologies (CINTED) – UFRGS – Brazil
rosa@inf.ufrgs.br

Abstract. Logic is a fundamental discipline for Computer Science, and Engineering students. However, despite its importance, there are several problems with the teaching of this discipline in graduate courses. Trying to improve this situation, we designed, and developed a new tutoring system for Logic, called Heraclito. This system implements a dynamic and adaptive student model, which is able to automatically solve the problems presented to students in a way similar to the employed by teachers, and, at the same time, is able to follow, and adapt itself to the form of reasoning used by students. The paper presents the main components of Heraclito's student model, including the formal definition its similarity measurement function, and the similarity experiments conducted with Logic proofs generated by this system.

Keywords: Logic Tutors, Automatic Provers, Natural Deduction, Intelligent Tutoring Systems, Learning Objects.

1 Introduction

Despite the importance of Logic for the development of analysis, formalization, and troubleshooting skills, there are several problems with the teaching of this discipline in graduate courses. In practice, the difficulties begin when concepts such as formulas, inference rules, and formal proofs begin to be taught. This was our main motivation to conceive a new tutoring system for Logic called Heraclito, intended to help the teaching of Natural Deduction in Propositional Logic (NDPL). Heraclito belongs to the class of intelligent tutoring system for Logic intended to teach deduction in Propositional Logic. In this category, the work that stands out most is the environment Logic-ITA[11], which is an intelligent tutoring system with a fairly traditional architecture for teaching deduction. The KRRT[1], P-Logic Tutor[7], and the Hint Factory[9] method to automatic generation of hints for a logic tutor, are also important examples of this type of system. More recent work has explored the use of probabilistic models (Markov models) to infer the main properties of the student model[2]. Heraclito differs from these systems in several aspects. Its approach to teach Logic, based on

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 482–489, 2014.

Socio-Historical theory[10], is not the common way to design Logic tutoring systems. There are relatively few works on tutoring systems explicitly based on socio-historical concepts. To date, ECOLAB[6], and the collaborative learning system of Chiru, and Trausan-Matu [4], are the most outstanding examples.

Other advance of Heraclito, which is the focus of this paper, resides in its student model. Different than Logic-ITA[11], Heraclito can analyze the logical validity of current proof step, and identify the tactical role this step assumes in the overall proof strategy: if it is useful, or not for the strategy. Different than Hint Factory[9], Heraclito's automatic hints are based on this prover, and not on previous teacher's experience represented by a bayesian probabilistic model. To assure the credibility of these hints, we take an indirect way, measuring objectively the similarity degree achieved by proofs automatically generated by the prover, when compared to proofs made by Logic teachers.

2 NDPL Proofs and the Proof Similarity Measurement

Formulas of Propositional Logic (PL) form the main interaction language between Heraclito, and students. A PL *well-formed formula* (or simply a *formula*) ϕ is a simple proposition A, B, C, ..., or is formed by the combination of simple propositions by logical conjunction ($\phi \wedge \psi$), disjunction ($\phi \vee \psi$), negation ($\neg\phi$), the conditional ($\phi \rightarrow \psi$), and bi-conditional ($\phi \leftrightarrow \psi$) operators. An argument $\phi_1, \phi_2, ..., \phi_n \vdash \psi$ is formed by a set of hypothesis or premises, an $\phi_1, \phi_2, ... , \phi_n$ a conclusion ψ. The argument is *valid* if, and only if, the conclusion is true, when all hypothesis are true. The deductive system used by Heraclito is very similar to the deductive system presented in [3]. It uses the introduction, and elimination rules for the five logical operators. There are two hypothetical rules: the *Reductio Ad Absurdum* (*RAA*) and the *Proof of the Conditional* (*PC*). Heraclito also supports derived rules like *Modus Tollens*, Disjunctive Syllogism, Hypothetical Syllogism, Constructive Dilemma, Exportation and Inconsistency.

The similarity measurement used by Heraclito was based on Jaccard index[8], extended to measure the similarity of the main proof, and then generalized to handle subproofs. A proof P is a finite set of steps $P=\{p_1, p_2, ..., p_n\}$ with each step indexed by its line number. A *fragment* $FP \subseteq P$ is a subset of a proof P, which contains some proof-steps of P, but which do not necessarily start in 1, or go continually till n. Each proof-step p_i is a quadruple $p_i=<l,\phi,r,rfs>$, where: l is the subproof level, ϕ is the formula, r the inference rule, and rfs is the ordered list of references. The functions $l(p_i)$, $\phi(p_i)$, $r(p_i)$, and $rfs(p_i)$ return the value of these components for the proof step p_i. A proof P is a *well-formed proof* if for all $p_i \in P$, the formula $\phi(p_i)$ is a well-formed formula resulting from the correct application of the deduction rule $r(p_i)$ over referenced steps $rfs(p_i)$.

The measurement of the syntactical similarity of two well-formed proofs, P and Q, is based on how much steps of these two proofs *match*. A match between two pairs of proof-steps $p_i \in P$, and $q_j \in Q$ occur if both formulas, and deduction rules of these steps are equal, i.e., $\phi(p_i)=\phi(q_j)$, and $r(p_i)=r(q_j)$. To measure the syntactical similarity index of two formulas, is necessary to abstract all line numbers, references, and subproof

levels of the steps of some proof. This is made through the operator $\delta(P)=\{<\phi(p_i),r(p_i)> \mid p_i \in P\}$. Now, if P and Q are well-formed proofs, which do not contains subproofs it is possible to calculate the Jaccard index $J(A,B)$ using $\delta(P)$ and $\delta(Q)$, as follows: $J(\delta(P), \delta(Q))=|\delta(P)\cap\delta(Q)| / |\delta(P)\cup\delta(Q)|$.

This provides the basic similarity index between proofs P and Q. However, the restriction on subproofs is much strong and unnecessary. To solve this, we created the *general similarity degree of proofs* $\Delta(P,Q)$, which measure the similarity on unrestricted well-formed proofs, based on a recursive process that follows the structure of the subproofs, measuring the similarity of the subproofs, and then summing up, and normalizing these indexes. To implement the normalization we departed from the purely set-theoretic approach of Jaccard index, and used recursive counting functions.

Some auxiliary operations and classes were used to define $\Delta(P,Q)$. The operation $P_{i..j}=\{p_k \in P \mid I <= k <= x\ j\}$ selects a contiguous part of some proof P, from step i to j. The operation $P^n=\{p_k \in P \mid l(p_k)=n\}$ selects steps on particular level n. The operation $P_R=\{p_k \in P \mid r(p_k) \in R\}$ where R is a set of deduction rules, selects the proof-steps of proof P which used the rules in R. The operation $P \| Q=\{<p_i, \ q_j> \mid p_i \in P\ ,\ q_j \in Q\ ,\ \phi(p_i)=\phi(q_j)$ and $r(p_i)=r(q_j)\}$ makes the *match pairing* of P, and Q proofs (or proof fragments).

Note that, because this operation return pairs of proof steps, it possible for distinct pairs to have the same formula and rule, but have distinct line numbers if they have different references, or proof-levels. So, to extract the sets of unique pair matches resulting from $P \| Q$ we define the class $UP(P \| Q)$, as follows: $UP(P \| Q)=\{U \subseteq P \| Q \mid$ for all $<p, q>, <r, s> \in U\ ,\ p=r$ and $j=l$ if and only if $q=s\}$. Now, the function that counts the proof-step matches that occur in proof-level n and above of proof fragments P and Q, is defined as follows:

$$matches^n(P,Q)= \quad |\delta(P)\cap\delta(Q)| \quad + \quad \underset{UP(P^n_{\{RAA,PC\}} \| Q^n_{\{RAA,PC\}})}{\textbf{\textit{Max}}} \quad (spmatches^n_S(P,Q)) \qquad (1)$$

The term $|\delta(P^n)\cap\delta(Q^n)|$, counts the set of distinct matching steps contained in the level n of P and Q. The maximization right term discovers the maximum count of matching steps in the matching subproofs of P and Q. The $P^n_{\{RAA,PC\}} \| Q^n_{\{RAA,PC\}}$ operation will identify the set of matching subproofs of P and Q. If eventually two subproofs were duplicated in the same level of some of these proofs, then $UP(P^n_{\{RAA,PC\}} \| Q^n_{\{RAA,PC\}})$ will pick up only the sets of unique combinations of matching proofs, and then only the combination that returns the maximum matching count, will be used as the value of the right term. To count the matching proof steps of these combinations of matching subproofs, it is used the function $spmatches^n_S(P,Q)$ defined recursively for a set of matching subproofs S, and for a proof-level n, as:

$$spmatches^n_S(P,Q) = \begin{cases} matches +^{n+1}(P_{i..j},Q_{k..l}) + spmatches^n_{S\setminus<p,q>}(P,Q) \\ \qquad \text{if exists } <p,q> \in S \text{ such that} \\ \qquad\qquad rfs(p)=<i,j> \text{ and } rfs(q)=<k,l> \\ 0, \quad \text{otherwise} \end{cases} \qquad (2)$$

The left term of the sum, when S is not empty, counts the set of matching steps of some matching subproof pair $\langle p_i, q_j \rangle$ using the function $matches^n(P,Q)$ previously defined, only that, in this case, this function is applied to the next level $n+1$ of the proof. The right term of the sum is simply the recursive application of *spmatches()* for the remaining set of matching subproof pairs, without $\langle p,q \rangle$. Now, using $matches^n(P,Q)$, is possible to define the general syntactical similarity index $\Delta(P,Q)$ as follows: $\Delta(P,Q)=2\times matches^0(P,Q) / |P|+|Q|$.

3 The Student Model on Heraclito System

The Heraclito system is a component system of MILOS infrastructure[5]. Heraclito was designed as a multiagent system composed of three pedagogical agents that run on a server, and a set of interactive Heraclito Learning Objects (HLO). The HLO are Java applications, or Android tablets apps. They are active, and interactive objects built over a *proof edito*r, and combined with additional multimedia material about NDPL contents, and exercises. Heraclito has three pedagogical agents: the *Mediator*, the *Specialist*, and the *Student Profile* agents. *Mediator* agent selects pedagogical strategies and controls the tutoring process, based on the current status of the student model. This model emerges from the interplay among the proof editor, and these agents. This resulted in a model, which contains, besides the HLO the student is working, the current argument (exercise), and the partial proof being worked by the student, and diagnostic information provided by *Specialist* agent, which the identification of what part of the proof is being worked by the student (proof's premises, main part, or end of the proof), the percentile of the proof completed by the student, the current diagnose about the resolution process.

The focus of this paper is the *Specialist* agent, which represents the role of the teacher as a specialist in Logic. This agent follows the resolution process of the student, analyzing whether last step of the partial proof is on the right way to finish it. Its main responsibility is to estimate the system's (teacher's) degree of confidence that student will complete problem solving task, diagnosing if the last step produced by the student is: (1) *useful*: if the formula contained in the last step effectively contributes to the proof of the argument, the step is said to be useful; (2) *harmful*: if the hypothetical RAA or PC rule used in the last step will eventually prevents the completing of the proof for the argument, then the step is classified as harmful; (3) *redundant*: the last step is not harmful, but it is unnecessary to complete the proof. To find out which category the last step belongs, the *Specialist* uses a NDPL prover to complete the partial proof (see Fig. 1).

First it uses the prover to complete the proof, starting from the last step. If the completion is not possible, then the last step prevented the proof of the argument, and must be considered harmful. In NDPL this only can occur if the step starts a RAA or a PC subproof, which cannot be proved with current premises. To check if the last step is redundant *Specialist* invokes again the prover, but this time from one step *before* last step. If the last step does not belong to the completed proof, then this step is not considered necessary by proof strategies incorporated in the prover, and is classified as redundant. Otherwise, it is considered useful. After identifying the category of the last step, the *Specialist* passes this information to *Mediator*.

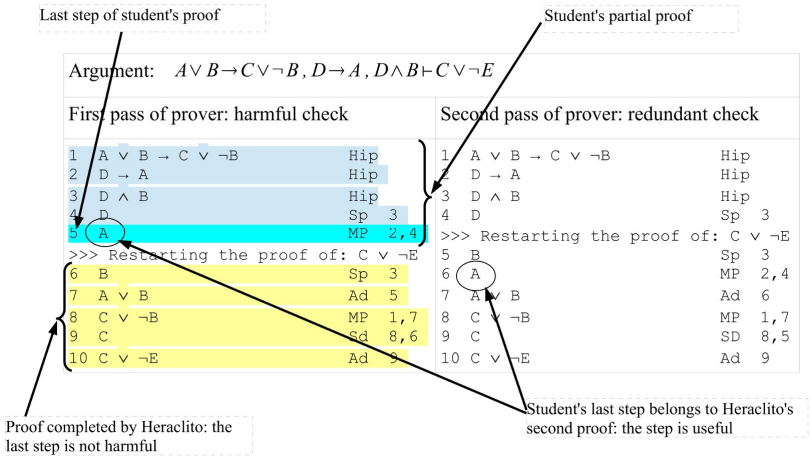


Fig. 1. Specialist agent analysis of a proof-step

The NDPL prover was developed in Prolog. It uses a mixed forward/backward chaining proof method, which tries to mimic students', and teachers'. The method is divided in three phases applied recursively: (1) premises expansion (forward chaining), (2) target checking, and (3) new target selection (backward chaining). The forward chaining phase (1) adds new formulas to the list of premises, through the application of NDPL elimination rules over formulas in the current list of premises. Fig. 2 shows the code for application of *Modus Ponens* rule. This phase implements a search process over elimination rules, trying to find the best rule to apply in each step. The ordering of `expand_premises()` clauses (see Fig. 2) program provide the basic heuristic used to imitate known teacher's resolution processes.

Phase (2) only checks if the target formula *P* of the argument already appears in the expanded list. If this is true then this recursive proving cycle ends. Otherwise, the prover pass to backward chaining phase (3), where the structure of the target formula is matched to NDPL introduction rules to select new targets to prove. Fig. 3 shows the code for disjunction introduction rule: to prove $P \vee Q$, first try to prove *P*, and if this does not work, try to prove *Q*.

The natural deduction proof is obtained by annotating in the output proof list (*ProofOut*), the rules applied in proving phases. To finish the proof is necessary to reverse its order, set its line numbers, proof levels, and references used by each rule, and eliminate unused steps eventually generated by the prover. The partial prover applies the same proving method as the full prover, after it has rehearsed or "re-proved" the part of the proof already made by the students. The basic "re-proving" is made by including all steps already made by the student in the initial premises list. However, if some RAA or PC subproof is found, then a new "re-proving" recursive procedure is started to keep the state variables of the prover updated, because the partial proof of the student could finish just in the middle of one of these subproofs. This allows the prover to recall the strategies used to reach to the current point of the proof, because the set of premises also provide heuristics used to select these strategies. Thus, the partial prover

can follow the reasoning used by the student, if there are corresponding strategies incorporated in its knowledge base. To do so, it incorporates several proving strategies, in addition to those which mimics known teacher's proofs.

```
% Modus Ponens: P, P -> Q :-- Q
expand_premises( Premises, NewPremises1, ProofIn,  ProofOut1) :-
  member( P -> Q, Premises),
  member( P1, Premises),
  \+member( Q, Premises),
  check_antecedent_disjunction(P1,P,ProofIn,ProofOut),
  add(Q, Premises, NewPremises),
  !,
  expand_premises( NewPremises, NewPremises1, [step(0, 0, Q, mp, []) | ProofOut], Proo

expand_premises( Premises, NewPremises1, ProofIn,  ProofOut1) :-
  member( P -> Q, Premises),
  check_antecedent_conjunction( P, Premises,ProofIn,ProofOut),
  \+member( Q, Premises),
  add(Q, Premises, NewPremises),
  !,
  expand_premises( NewPremises, NewPremises1, [step(0, 0, Q, mp, []) | ProofOut], Proo
```

Fig. 2. Application of *Modus Ponens* rule in phase (2) of NDPL prover

```
% Disjunction introduction: P :-- P v Q; Q :-- P v Q
retarget_prove( Premises, P v Q, ProofIn, [step(0, 0, P v Q, adic, []) | ProofOut]) :-
  is_proving(adic, Premises, P),
  check_prove( Premises, P, ProofIn, ProofOut),
  is_proved(adic, Premises, P).

retarget_prove( Premises, P v Q, ProofIn, [step(0, 0, P v Q, adic, []) | ProofOut]) :-
  is_proving(adic, Premises, Q),
  check_prove( Premises, Q, ProofIn, ProofOut),
  is_proved(adic, Premises, Q).
```

Fig. 3. Application of disjunction introduction in phase (3) of NDPL prover

4 Similarity Experiment

The similarity experiment measured the degree of similarity of the proofs generated by the *Specialist* agent, when compared to a set of proofs produced by a group of four teachers of Logic. The comparing proofs where collected from the teacher's solutions for an exercise book in Logic made in 2009. This was two years before the start of Heraclito's project, so the proofs have no relation with this system. A total of 51 well-formed proofs were selected from the exercise book, the only condition was that these proofs used only the NDPL rules used by Heraclito. Then, for each teacher's proof P, it was calculated the similarity index $\Delta(P,Q)$ in relation to the Heraclito corresponding proof Q. The average similarity degree was high, achieving 84.3%. A subset of 33 proofs achieved a 100% degree value, being complete matches between teacher's solutions, and Heraclito's generated proofs. From the remaining proofs, 5 achieved a high degree of similarity of 80% or more. Fig. 4 shows a typical example of a high similarity index proofs.

Argument: $S \leftrightarrow P, \neg P \vdash \neg S$ **, with similarity index** $\Delta(P, Q)$ = 85,7%

Teacher's proof		**Heraclito's Proof**	
1 S ↔ P	Hip	1 ¬P	Hip
2 ¬P	Hip	2 S ↔ P	Hip
3 |S	Hip-RAA	3 S → P	-Eq 2
4 |S → P	-Eq 1	4 | S	Hip-RAA
5 |P	MP 3,4	5 | P	MP 3,4
6 |P ∧ ¬P	Cj 2,5	6 | P ∧ ¬P	Cj 5,1
7 ¬S	RAA 3-6	7 ¬S	RAA 4-6

Fig. 4. Example of proofs with high similarity index

In this case, the difference was due to the decision on when to apply elimination rules. Heraclito's prover strategy is to apply these rules as early as possible. The teacher strategy was to apply elimination rules when necessary. These different strategies generally cause no differences in the similarity index, because they are kept in the same proof level, and the similarity index does not takes into consideration the ordering of steps. However, in this proof the steps resulting from elimination rules were put on different levels causing a measurable difference on $\Delta(P,Q)$.

Argument: $P \rightarrow Q \vdash \neg P \vee Q$ **, with similarity index** $\Delta(P, Q)$ = 32%

Teacher's proof		**Heraclito's Proof**	
1 P → Q	Hip	1 P → Q	Hip
2 |¬(¬P ∨ Q)	Hip-RAA	2 | ¬(¬P ∨ Q)	Hip-RAA
3 ||P	Hip-RAA	3 || P → Q	Hip-RAA
4 ||Q	MP 1,3	4 ||| P	Hip-RAA
5 ||¬P ∨ Q	Ad 4	5 ||| Q	MP 1,4
6 ||(¬P ∨ Q) ∧ ¬(¬P ∨ Q)	Cj 2,5	6 ||| ¬P ∨ Q	Ad 5
7 |¬P	RAA 3-6	7 ||| ¬P ∨ Q ∧ ¬(¬P ∨ Q)	Cj 6,2
8 |¬P ∨ Q	Ad 7	8 || ¬P	RAA 4-7
9 |(¬P ∨ Q) ^ ¬(¬P ∨ Q)	Cj 2,8	9 || ¬P ∨ Q	Ad 8
10 ¬¬(¬P ∨ Q)	RAA 2-9	10 || ¬P ∨ Q ∧ ¬(¬P ∨ Q)	Cj 9,2
11 ¬P ∨ Q	DN 10	11 | ¬(P → Q)	RAA 3-10
		12 | (P → Q)∧ ¬(P → Q)	Cj 1,11
		13 ¬¬(¬P ∨ Q))	RAA 2-12
		14 ¬P ∨ Q	DN 13

Fig. 5. Example of proofs with low similarity index

A subset of 6 proofs achieved a medium similarity degree, from 60% to less than 80%. The remaining 7 proofs achieved only a low similarity index. Fig. 5 shows an example of this situation. In this case, even considering that the general strategy is the same in both proofs, being based on the use of *Reduction Ad Absurdum*, Heraclito prover used more steps to reach the same conclusion, resulting in a low similarity degree. Note that, because of the adaptive and dynamic behavior of the student model, this kind of difference is not a big issue for the tutoring system as a whole. If some student is really following the strategy outlined in the left side proof in Fig. 5, then the corresponding step 3 will be included as part of the partial proof being created by this student. With this, the prover has enough information to finish the overall proof following the same strategy.

5 Conclusions and Future Works

Heraclito is being experimentally used for almost two years to help the teaching of Logic. Until now the results are encouraging, providing favorable evidences that Heraclito is on the right way to become a good tool to help students learn Logic. Its prover has the ability to follow the reasoning used by the student, being able to restart the proof exactly at the point where the student left off. This ability is the basis for the student model, which emerges from the interactions between Heraclito's agents and the student. This is an important contribution of Heraclito, because what emerges from these interactions is a truly dynamic and adaptive student model that tries to adapt itself to the way that the student is trying to solve the exercise, not forgetting the "right" way to solve it.

From now on, we pretend to use Heraclito as an effective tool to teach Logic in our Universities. We are developing a production version of this system able to support up to 500 students in several different classes. To accomplish this task we are designing and developing the scalability of Heraclito, through the distribution of the agents of this system for operation on a cluster of machines.

References

1. Alonso, J.A., Aranda, G.A., Martn–Mateos, F.J.: KRRT: Knowledge Representation and Reasoning Tutor System. In: Moreno Díaz, R., Pichler, F., Quesada Arencibia, A. (eds.) EUROCAST 2007. LNCS, vol. 4739, pp. 400–407. Springer, Heidelberg (2007)
2. Barnes, T., Stamper, J.: Automatic Hint Generation for Logic Proof Tutoring Using Historical Data. Educational Technology & Society 13(1), 3–12 (2010)
3. Baronett, S.: Logic. Pearson (2008)
4. Chiru, C.-G., Trausan-Matu, S.: Identification and Classification of the Most Important Moments from Students' Collaborative Discourses. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 330–339. Springer, Heidelberg (2012)
5. Gluz, J.C., Vicari, R.M., Passerino, L.M.: An Agent-Based Infrastructure for the Support of Learning Objects Life-Cycle. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 696–698. Springer, Heidelberg (2012)
6. Luckin, R., du Boulay, B.: Ecolab: The Development and Evaluation of a Vygotskian Design Framework. Int. J. of Artificial Intelligence in Education 10(2), 198–220 (1999)
7. Lukins, S., Levicki, A., Burg, J.: A tutorial program for propositional logic with human/computer interactive learning. SIGCSE Bull 34(1), 381–385 (2002)
8. Markov, Z., Larose, D.T.: Data Mining the Web: Uncovering Patterns in Web Content, Structure, and Usage. John Wiley & Sons, Hoboken (2007)
9. Stamper, J., Barnes, T., Croy, M.: Enhancing the Automatic Generation of Hints with Expert Seeding. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part II. LNCS, vol. 6095, pp. 31–40. Springer, Heidelberg (2010)
10. Vygotsky, L.S.: Thought and Language. The M.I.T. Press, Cambridge (1986)
11. Yacef, K.: The Logic-ITA in the classroom: A medium scale experiment. Int. J. of Artificial Intelligence in Education 15(1) (2005)

The Effect of Automatic Reassessment and Relearning on Assessing Student Long-Term Knowledge in Mathematics

Yutao Wang and Neil T. Heffernan

Department of Computer Science
Worcester Polytechnic Institute
100 Institute Road, Worcester, MA
{yutaowang,nth}@wpi.edu

Abstract. Intelligent Tutoring Systems (ITS) give assessments to estimate a student's current knowledge. A great deal of work in the past years, (e.g. KDD Cup2010) has focused on predict students immediate next performance, while what is important is will the student retain that knowledge for later use. Some previous studies such as Wang, et al, Xiong, et al. have started to investigate this question by trying to predict student retention after a time interval of several days. We created a novel system that would automatically reassess and allow students to relearn the material to enhance a student's long-term knowledge. It is showed before that this intervention raised student learning, and now we are wondering if it also makes assessment of student long-term knowledge better (i.e, more predictive power). The result shows that the reassessment and relearning information is very useful in assessing student long-term knowledge.

keywords: Intelligent Tutoring System, dynamic assessment, reassessment and relearning, long-term knowledge, student modeling.

1 Introduction

The ASSISTments project is premised on the notion our schools are asked to do too much testing. Every minute testing is a minute stolen from instruction. The solution is to use data from students learning for assessment purposes. Intelligent Tutoring Systems (ITS) give assessments to estimate student current knowledge and predicts student performance on the immediate next action has been investigated by many researchers. But what if our goal is not to ask "do they know this right now?" but "will they retain this knowledge later?" This is a more important question because the purpose of education is to teach students so that they can retain it rather than immediately understand it but quickly forget. Some previous study [1,2] have investigated this question by trying to predict student performance after a several days interval. In this paper, instead, we are trying to predict student performance after a much longer – six months interval.

Compared to traditional assessment, the dynamic assessment [3] that we are conducted in this study utilizes the amount of assistance that students require to judge the depth of student knowledge. We would not be the first to show that letting students

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 490–495, 2014.

learn could help assessing. Different researchers showed that by offering increasingly more explicit prewritten hints in response to incorrect responses, better assessment can be achieved [4,5,6]. ASSISTments itself has been used in the past along similar lines [7] and has been shown that we can better predict students state test scores if we use the number of hints, their responses and other student data.

We created a novel system that would automatically reassess and allow students to relearn the material to enhance students' long-term knowledge. We call it the Automatic Reassessment and Relearning System (ARRS). Details on how ARRS works can be found here [8]. The ARRS system gives us an opportunity to investigate two interesting questions. First, do the models for assessing student knowledge retention several days later perform differently from those for assessing student knowledge retention after a longer time interval (six months)? Second, can we do better in assessing student knowledge retention after six months by utilizing the extra information gathered from the ARRS system? The main difference between this study and previous ones is that not only features of student learning behavior, but also features of student relearning behavior were investigated.

Different logistic regression models were built and analyzed to address these two questions. The result showed that given the same feature set, higher accuracy can be achieved in assessing shorter interval knowledge retention than the longer interval retention, which indicates that assessing longer interval retention could be a harder task. With the extra information of student reassessment and relearning, however, we were able to assess student longer interval retention even better than the shorter interval retention. This result suggests that reassessment and relearning information is very useful in assessing student long-term knowledge. Details about the experiments, including a brief introduction to the ARRS system, can be found on our webpage [8].

2 Methods

2.1 The Tutoring System and Dataset

The data used here came from two ARRS experiment classes in the ASSISTments platform in school year 2010-2011. This data is available here [8]. The ARRS is a sub-system build in the ASSISTment platform, which automatically reassess student a week later, a month later, and then finally two months after a student originally masters a skill (master here means achieve a preset level -- typically three consecutive correct answers). If students fail the reassessment, they will be given an opportunity to relearn the topic until master it again.

There were 128 students, 33 skills and 53449 data instances in this experiment. Students were separated into groups 1 and 2, and skills were separated into groups A and B. At the beginning of the experiment, all students completed a first assignment of each skill. Then group 1 students did group A skills assignments in the ARRS while group 2 students did group B skills assignments in the ARRS. After six months, all students were given a one item per skill posttest.

To simplify the analysis, in this study we focused on the first reassessment and relearning phase, that means only data from the first assignment (*first phase*) and the

one week later reassessment and relearning assignment (*second phase*) were included in this study. We also excluded 29 students since they missed either the first assignment or the posttest for some skills. We excluded student skill pairs in the ARRS condition where seven days later reassessment or relearning was not finished. These data pre-processing result in 1538 student skill pairs for the control condition and 1587 student skill pairs for the ARRS condition.

2.2 Models and Analysis

Logistic regression models were built to assessing student long term knowledge. Features includes the prior knowledge *firstp_pretest*, information of student's original learning process: *firstp_avg_correct, firstp_avg_phint, firstp_avg_attempt, firstp_nquestions*, the prior knowledge at seven days later *secondp_pretest*, and information of student re-learning process: *secondp_avg_correct, secondp_avg_phint, secondp_avg_attempt, secondp_nquestions*. Forward input stepwise procedure was conducted to eliminate useless features.

We used The Root Mean Squared Error (RMSE) of predicting a posttest score as a measure of assessing accuracy. *secondp_pretest* was the target for assessing shorter term retention, and *posttest* was the target when assessing longer term retention.

A 5- fold cross validation was done for all of the models. That is, we randomly separated all student skill pairs into five folds, and ran all the models five times. Each time the models were trained on four folds and tested on the remaining one fold.

RQ1: Do the models for assessing student knowledge retention several days later perform differently from those for assessing student knowledge retention after a longer time interval?

To answer this question, we built two comparable models as shown in Table 1. The Shorter-term_Phase1_Model used features from the first assignment to predict student knowledge retention one week later, while the Longer-term_Phase1_Model used features from the first assignment to predict student knowledge retention six months later. To avoid the influence of the relearning in predicting the longer term knowledge retention, we used control group data to evaluate the Longer-term_Phase1_Model. And we used ARRS group data to evaluate the Shorter-term_Phase1_Model because there is no data on control group's shorter term knowledge retention.

Table 1. Short-term_Phase1_Model (SP1) vs. Long-term_Phase1_Model (LP1)

Model	Dependent	Data	Feature Selected	RMSE
SP1	*phase2_pretest*	ARRS	*firstp_avg_correct* *firstp_avg_attempt*	0.4049
LP1	*posttest*	Control	*firstp_avg_correct* *firstp_avg_phint* *firstp_avg_attempt*	0.4296

Since both conditions had the same group of students, we were able to compute a student level paired t-test to determine whether the RMSE difference between these two models was statistically reliable. The result is statistically reliable, $t(98) = 2.58$, $p = 0.01$. The result suggests that the six months knowledge retention is harder to assess than the seven days knowledge retention is not surprising, and some may say it's trivial. However, the short term model helped us in setting up a baseline for the models of assessing longer term retention to compare with.

RQ2: Can we do better in assessing student knowledge retention six months later by utilizing the extra information gathered from the ARRS system?

Similar to RQ1, we built several logistic regression models as shown in Table 2. All these models predicted the posttest score using the ARRS group data.

Phase1and2_Model used all the features from both the first assignment, and the ARRS assignment seven days later in a single stepwise logistic regression model.

To improve upon the Phase1and2_Model, we considered the fact that some of the ARRS student skill pairs do not have relearning features because they answered their reassessment question correctly. This caused large amount of missing data when we use a single model to describe all the ARRS data. We then built a model called Phase1and2_Combined_Model, which was the combination of two sub-models: Phase1and2_norelearning_Model, and Phase1and2_relearning_Model. The Phase1and2_norelearning_Model ran on the student skill pairs in which the students did not need to relearn the material for the skill, while the Phase1and2_relearning_Model ran on the student skill pairs in which the students needed and finished the relearning assignment.

Table 2. Phase1and2_Model vs. Phase1and2_Combined_Model

Model	Data	Feature Selected	RMSE
Phase1and2_Model	ARRS	*firstp_avg_correct firstp_avg_phint secondp_avg_correct secondp_nquestions*	0.3886
Phase1and2_Combined_Model	ARRS	--	0.3861
Phase1and2_norelearning_Model	ARRS no relearn	*firstp_avg_correct firstp_avg_phint*	--
Phase1and2_relearning_Model	ARRS relearning	*firstp_avg_correct secondp_avg_correct secondp_nquestions*	--

In Table 2, the feature column of Phase1and2_Combined_Model is empty, because it is the combination of two different models (Phase1and2_norelearning_Model and Phase1and2_relearning_Model). Also, the RMSE column for Phase1and2_norelearning_Model and Phase1and2_relearning_Model is empty, because the dataset of these two

models are different with other models in the table, thus the RMSEs of these two models are not comparable with other models'.

Until now, we could draw two conclusions. First, by comparing Longer-term_Phase1_Model in Table 1 and Phase1and2_Model in Table 2, we observed the extra features gathered from ARRS did improve the accuracy in assessing student long term knowledge. A student level paired t-test suggested this improvement was statistically reliable, $t(98) = 4.61$, $p < .001$. Compared to the short term model Shorter-term_phase1_Model, however, this new long term model has a better, but not reliably better, RMSE, $t(98) = 1.82$, $p = 0.07$. Second, by separating models according to whether or not a student needed relearning for one skill, we were able to further improve the model for assessing long term knowledge. Although the improvement between the Phase1and2_Model and the Phase1and2_Combined_Model was not reliable, $t(98) = 1.05$, $p = 0.30$, amazingly, the Phase1and2_Combined_Model was able to reliably improve upon the short term model Shorter-term_Phase1_Model in Table 1, $t(98) = 2.02$, $p < 0.05$. This proved again the importance of the relearning features, especially the average correctness in the relearning phase and the number of questions students need to relearn a material.

3 Discussion and Future Work

In this paper, we compared model performance between assessing student shorter interval knowledge retention and longer interval knowledge retention. Results suggested that longer interval knowledge retention is harder to assess. We then investigated the effect of the extra features gathered from ARRS and concluded that relearning features are useful in assessing long-term knowledge retention.

One limitation of this work is the amount of data. The experiment was a pioneer study of ARRS, and has only several thousands of data instances. Verifying the result of this study in a larger dataset or a different tutoring system could be helpful.

Another limitation is that we only used the information from the one week later reassessment and relearning phase. Future study could further investigate the predicting power of data from the later phases of two weeks, one month, and two months later.

4 Contributions

This paper analyzed data gathered from a novel system, which automatically reassesses student knowledge and allows them to relearn the material, to evaluate its power in assessing student long-term knowledge and makes several contributions.

First, assessing student current knowledge has been investigated by researchers in ITS for many years. Recently some researchers have discovered the difference between assessing current knowledge and knowledge retention several days later. We further explored this topic, and compared the model performance between assessing student shorter interval retention (seven days later) and longer interval retention

(six months later). Results showed that the later is a harder problem to address. We then concentrated on improving the assessing accuracy of the longer interval retention.

Second, for the task of predict student knowledge six months later, compared to other dynamic testing methods, we not only looked at the assessment power of the features in student learning process, but also the assessment power of these features in student relearning process. To do so, instead of using data from a single session, we also used data from a second session at one week later. We built and analyzed different stepwise logistic regression models to see if number of problems (learning speed) and other features (hints and attempt) of the second session help the prediction. Result shows that having data from both the learning session and the relearning session lead to better prediction. More interestingly, it shows that tracking how much relearning (measured by the number of problems student need to finish before re-mastery a skill) students need was a useful predict. This indicates that student relearning time is a useful indicator of the depth of student knowledge.

Furthermore, we found that making separate models according to weather or not a student needs relearning for a skill gives better prediction, and surprisingly, even better than predicting students' shorter interval retention.

Acknowledgements. We acknowledge funding from NSF (#1316736, 1252297, 1109483, 1031398, 0742503), ONR's 'STEM Grand Challenges' and IES (#R305A120125 & R305C100024).

References

1. Wang, Y., Beck, J.: Incorporating Factors Influencing Knowledge Retention into a Student Model. In: Proceedings of the Educational Data Mining Conference 2012, pp. 201–203 (2012)
2. Xiong, X., Li, S., Beck, J.E.: Will You Get It Right Next Week: Predict Delayed Performance in Enhanced ITS Mastery Cycle. In: FLAIRS Conference 2013 (2013)
3. Sternburg, R.J., Grigorenko, E.L.: Dynamic testing: The nature and measurement of learning potential. Cambridge University Press, Cambridge (2002)
4. Brown, A.L., Bryant, N.R., Campione, J.C.: Preschool children's learning and transfer of matrices problems: Potential for improvement. Paper presented at the Society for Research in Child Development meetings, Detroit (1983)
5. Campione, J.C., Brown, A.L.: Dynamic assessment: One approach and some initial data. Technical Report No. 361. Cambridge, MA: Illinois University, Urbana. Center for the Study of Reading. ED269735 (1985)
6. Attali, Y.: Immediate Feedback and Opportunity to Revise Answers: Application of a Graded Response IRT Model. Applied Psychological Measurement (2011)
7. Feng, M., Beck, J., Heffernan, N., Koedinger, K.: Can an Intelligent Tutoring System Predict Math Proficiency as Well as a Standardized Test? In: Baker, Beck (eds.) Proceedings of the 1st International Conference on Education Data Mining, pp. 107–116 (2008)
8. ARRS study: `https://sites.google.com/site/assistmentsdata/arrs`

Predicting Student Performance in Solving Parameterized Exercises

Shaghayegh Sahebi[1], Yun Huang[1], and Peter Brusilovsky[2]

[1] Intelligent Systems Program, University of Pittsburgh, Pittsburgh, USA
[2] School of Information Sciences, University of Pittsburgh
{shs106,yuh43,peterb}@pitt.edu

Abstract. In this paper, we compare pioneer methods of educational data mining field with recommender systems techniques for predicting student performance. Additionally, we study the importance of including students' attempt time sequences of parameterized exercises. The approaches we use are Bayesian Knowledge Tracing (BKT), Performance Factor Analysis (PFA), Bayesian Probabilistic Tensor Factorization (BPTF), and Bayesian Probabilistic Matrix Factorization (BPMF). The last two approaches are from the recommender system's field. We approach the problem using question-level Knowledge Components (KCs) and test the methods using cross-validation. In this work, we focus on predicting students' performance in parameterized exercises. Our experiments shows that advanced recommender system techniques are as accurate as the pioneer methods in predicting student performance. Also, our studies show the importance of considering time sequence of students' attempts to achieve the desirable accuracy.

1 Introduction

Parameterized questions and exercises have recently emerged as an important tool for online assessment and learning. A parameterized question is essentially a template for the question, created by an author. At presentation time, the template is instantiated with randomly generated parameters. As a result, a single question's template is able to produce a large number of different questions. One of the benefits of this technology is in the self-assessment context: the same question can be used again and again with different parameters. This allows every student to achieve understanding and mastery.

On the practical side, this property and other benefits, such as re-usability and being cheating-proof, made parameterized exercises very attractive for the large-scale online learning context. In turn, it made platforms that supported parameterized questions such as LON-CAPA [8] or edX very popular for college-offered online learning and MOOCs.

On the research side, a range of studies have confirmed the value of parameterized questions as e-learning tools [5,9,1,4]. At the same time, Hsiao et. al's [4] experience with parameterized questions in the self-assessment context demonstrated that the important ability to try the same question again and again is

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 496–503, 2014.

not always beneficial, especially for students who are not good in managing their learning. The analysis of a large number of student logs revealed some considerable number of unproductive repetitions. For example, we can observe many cases where students repeatedly try and correctly solve the same exercise with different parameters (which is at the time apparently easy for them) instead of focusing on new, more challenging questions. We can also observe repetitive failed attempts to solve the same exercise for which the students are apparently not ready, instead of focusing on simpler exercises and missing knowledge.

We believe that this unproductive practice could be avoided if a personalized e-learning system featuring parameterized exercises can predict the success of students' future problem-solving attempts in the same way as a recommender system can predict, for example, whether a user would or would not like a new movie. The ability to predict students' performance in the context of solving parameterized exercises could enable the system to intercept non-productive behavior and recommend a more efficient learning path. We also believe that the presence of a large volume of learning data that is now collected in online learning systems makes the task of performance prediction possible. In addition, we beleive that the repetition of exercises makes the attempt sequencing of students' activities an important factor to predict their performance. As a result, we expect the time-aware (or sequence-aware) approaches to perform better in this context. However, so far, there have been no attempts to explore approaches for predicting success in solving parameterized exercises. This paper attempts to bridge this gap by exploring a range of techniques for performance prediction. We compare advanced log-driven time-aware prediction approaches such as Bayesian Knowledge Tracing [2], Performance Factor Analysis [11], and tensor factorization (as an advanced collaborative filtering approaches [6]) with matrix factorization (as a baseline approach that does not model attempt sequences).

2 Background: Predicting Student Performance

The traditional approach to predict user experience with unknown items using the past experience of the user, along with a large community of other users, was developed in the field of collaborative recommender systems [10]. While collaborative filtering approaches were designed to predict user taste, not user performance, technically it is resolved to predicting a score for unknown items based on the past experiences of users. We can consider users of a collaborative filtering system as students, items as skills/questions/steps in solving the problem, and user rating as the predicted value representing student's success/failure. In recent years, more modern approaches, such as matrix factorization [7] and tensor factorization [6] have been used in recommender systems. There are several works applying factorization techniques to student modeling, such as Thai et. al's tensor factorization [12]. But none of these works are focused on predicting user performance in parameterized questions at the question level.

Another approach for Predicting Student Performance (PSP) in problem solving is based on the idea of cognitive modeling. With cognitive modeling, each

problem or problem-solving step (item) is associated with specific units of knowledge (Knowledge Components or KCs) to be mastered. Observing students' past successes and failures, a cognitive modeling system attempts to model student mastery for each unit of knowledge. The traditional approach for cognitive modeling is Bayesian Knowledge Tracing (BKT) [2], which employs a two-state dynamic Bayesian network estimating the latent cognitive state (student knowledge) from students' performance.

More recently Performance Factor Analysis (PFA) [11] has emerged as a powerful approach for cognitive modeling and performance prediction. PFA takes into account the effects of the initial difficulty of the KCs and prior successes and failures of a student on the KCs associated with the current item.

The problem of PSP in the context of solving parameterized problems is somewhat harder than predicting solving regular "solve-once" problems. Traditional modeling approaches are not fully adequate for parameterized problem case since they can't distinguish repeated attempts to solve the same problem from solving a new problem related to the same skills. While there are some works focused on performance prediction in classes with parameterized exercises, they focus on a much coarser level of prediction, such as PSP in the whole class [9].

3 The Approaches

As we stated in the introduction section, we expect the time-aware approaches perform better than time-ignorant approaches in PSP for parameterized exercises. Also, we expect advanced recommender systems approaches to perform as good as the pioneer methods in PSP. To study these expectations, we experiment on four student modeling approaches: BKT [2], PFA [11], Tensor Factorization [6], and Matrix Factorization [13]. As the previous work in PSP is focused on knowledge tracing and regression models, we choose a method of each: BKT and PFA. As for approaches of recommender systems, we choose a tensor factorization method that can include students' attempt sequence and a matrix factorization method. Each of these methods has their positive and negative aspects; e.g. BKT can model the time sequence of student attempts while PFA cannot model that explicitly; PFA can handle multiple knowledge components while BKT can only model one KC; and tensor factorization and matrix factorization methods predict a personalized performance for each student. We choose a Max baseline in addition to the above methods. This baseline predicts success (the majority class) for every attempt. Using this baseline, we explore how our models preform given our imbalanced data. In the following, we provide a brief description of each of the methods.

Bayesian Knowledge Tracing: The Bayesian Knowledge Tracing [2] model assumes a two-state learning model where each Knowledge Component (skill, or rule) is either in the learned or unlearned state. It uses a simple dynamic Bayesian network where the observable variable represents student performance (correct or incorrect) and the hidden variable represents student knowledge state. There are four parameters in BKT : the initial knowledge parameter ($p(L_0)$)

represents the probability that the student knows a KC before practicing on any items associated with the KC; the learning rate parameter ($p(T)$) represents the probability that a student learns a KC by practicing; the guess parameter ($p(G)$) represents the probability when a student doesn't know a KC but answers the item correctly; the slip parameter ($p(S)$) represents the probability when a student knows the KC but answers the item incorrectly.

Performance Factor Analysis: Performance Factor Analysis [11] predicts student's performance based on the easiness of the current Knowledge Component(s), student's prior correct responses and incorrect responses on the KC(s) associated with the current item using a standard logistic regression model. The correctness of response of a student on an item is modeled as the dependent variable here. PFA does not model time sequences directly, but it considers them as the number of past successes and failures.

Matrix/Tensor Factorization: Matrix factorization is a popular approach in the recommender systems field. In the educational data mining domain, to predict student performance, we can model a user's attempt on all of the items as a one-dimensional binary array of length q (number of items). If a user succeeds in solving that item, the value for that element will be one and zero otherwise. Considering all of the items and all of the students, we can model all students' success or failure on all questions using an $s \times q$-matrix. Since different students might have different number of attempts on various items, we consider only the success or failure of the last attempt of the student. Some of the values of this matrix are unknown to us because some students might have never tried an item. The task of predicting user performance aims to find the values of these unknown elements of the matrix. In this paper, we use a Bayesian probabilistic matrix factorization (BPMF) method [13] to predict the success or failure of students in various questions.

However, a student might have more than one attempt with different results on an item. Thus, we should consider a method to incorporate time into the factorization model. One way of doing so is to use tensors. A tensor is a multi-dimensional or N-way array. A matrix is a 2-way tensor. In our problem, the sequence of one user's attempt on one item can be seen as a t-dimensional array consisting of zeros and ones. Zeros are representative of student's failure in that particular attempt of the item and ones are indicative of success. Consequently, if we want to model all the attempts each student has made on each item, we will end up with a three-dimensional tensor of the size $s \times q \times t$, which has binary values of failure or success. The task of predicting user performance here aims to find the success or failure of a student in each attempt of an item. Tensor factorization methods try to decompose a tensor into lower-dimensional space and predict the missing values of the tensor by approximating them using this lower-dimensional representation. In this paper, we use the Bayesian probabilistic tensor factorization (BPTF) introduced by Xiong et. al [13] to predict the success or failure of students.

4 The Dataset

Our dataset was collected from the online self-assessment system QuizJET [4], which provides parameterized questions for learning Java programming. Each parameterized question is generated from a template filling parameters inside the question with random (and reasonable) values to avoid providing the exact same question to the student. Students can try different versions of the same question multiple times until they acquire the knowledge or give up. The dataset was collected from Fall 2010 to Spring 2013 (six semesters). The subject domain is organized into reasonably coherent topics, each topic has several questions. Each question is assigned to one topic. We experimented on $27,302$ records of 166 students on 103 questions. The average number of attempts on each question is equal to three. Students have at least one attempt to at most 50 attempts in one question. Our dataset is imbalanced: the total number of successful attempts in the data equals to $18,848$ (69.04%) and the total number of failed attempts is 8454. We used user-stratified 5-fold cross-validation to split the data, so that the training set has 80% of the users (with all their records) randomly selected from original dataset, while the remaining 20% of the users were retained for testing. We performed a 5-fold cross-validation to perform the comparison in our studies. We ensured that all of the questions seen in the test set have at least one student attempt in the training set. In this way, all models are predicting unseen students on observed questions in each run. Simple statistics of are dataset are shown in Table. 1.

Table 1. Dataset Statistics

	Average	Min	Max
#attempts per sequence	3	1	50
#attempts per question	265	25	582
#attempts per student	165	2	772
#different students per question	87	7	142
#different questions per student	54	1	101

5 The Experiment

Our approach is to consider each question to be a distinct subtopic and use questions as knowledge components for modeling. Among the methods discussed above, BPTF, BKT, and PFA each consider the student's attempt sequence in a way: BKT models it explicitly as an HMM, BPTF has a smooth changing condition for students' attempts and PFA summarizes this information in the number of previous successful and unsuccessful attempts. On the contrary, BPMF does not consider this information. To examine the performance of these approaches

and compare them using different information resources, we design the following experiment.

The Procedure. We treat a question (item) as a knowledge component (KC) in this set of experiments. By using question (item) level KCs, we would be able to capture a question's characteristic for predicting different attempts on the same question. To model the tensor, we use the three dimensions of student, question, and attempt. Each element of the tensor shows the success (1) or failure (0) of student in that question for the specific attempt. To model the matrix, we use the two dimensions of student and question. Each element of the matrix shows the success (1) or failure (0) of student in the last available attempt of that question.

We use existing tools implementing the above methods to perform our experiments. We use EM algorithm for BKT and set the initial parameters as follows: $p(L_0) = 0.5$, $p(G) = 0.2$, $p(S) = 0.1$, $p(T) = 0.3$. For running PFA, we use the implementation of logistic regression in WEKA [3]. For BPTF and BPMF, we utilize the Matlab code prepared by Xiong et. al.[1]. We experimented with different latent space dimensions for BPTF and BPMF (5, 10, 20 and 30) and chose the best one, which has the latent space dimension of 10.

Table 2. Results of the Methods with Question as Unit to Predict Student Performance

Methods	Accuracy	RMSE	TP	TN	FP	FN	Maj. precision	Min. precision	Maj. recall	Min. recall
BKT	74.38(0.8)	0.4152	3527.6	534.8	1156.0	242.0	75.33	68.69	93.43(0.9)	32.00
PFA	74.69(1.0)	0.4185	3381.4	701.4	989.4	388.2	77.34	64.16	89.56(1.1)	41.63
BPTF	74.26(0.9)	0.4189	3423.4	636.2	1054.6	346.2	76.42	64.59	90.60(1.4)	37.88
BPMF	71.73(0.5)	0.4365	3386.4	531	1159.8	383.2	74.39	58.46	89.95(1.6)	31.21
Max	69.04	0.5564	3769.9	0	1690.5	0	0.6904	0	100	0

The Results. The results of our experiments are shown in Table 2. Numbers in parenthesis show the confidence interval with $P < 0.05$. We can see that the accuracy of all models, except BPMF, are very close to each other. The BPMF model lacks the time sequencing of student attempts and performs poorly compared to the other three methods. All methods beat the Max baseline's accuracy. Among the models, BKT has slightly more true positives and false positives. It means that BKT tends to predict more positive values (successes) for the students. It over estimates the student's performance. BPTF has the next most true positives and false positives. PFA has more true negatives and

[1] `http://www.cs.cmu.edu/~lxiong/bptf/bptf.html`

false negatives than other approaches. It means that PFA tends to predict more failures for the students. BKT has the highest minority precision and significantly highest majority recall. PFA has the highest majority precision and highest minority recall. It means that if PFA predicts a success for a student and if BKT predicts a failure for students, their prediction is more likely to be true compared to the other methods.

6 Conclusion and Future Work

In this study, we explored several student modeling approaches in predicting student performance in solving parameterized exercises, particularly in the programming area. All the models we studied (BKT, PFA, BPTF, and BPMF) outperformed the max baseline, showing the feasibility of applying these student models to parameterized exercising systems which are different from the traditional step-by-step, fine-grained designed tutoring systems.

In our experiment, we saw that the sequencing information is an important factor in PSP for parameterized exercises and time-aware models perform better than the time-ignorant matrix factorization method. These time-aware methods do not differ significantly in results of PSP. This result encourages us to seek for more advanced approaches in this area, as future work.

In addition, the success of using BPTF, which is one of the advanced matrix factorization techniques in the recommendation area, encourages more research on applying more recommendation techniques in PSP. Giving that factorization techniques do not need to know the exact Knowledge Components that influence students' performance, they reduce the manual effort in exercising authoring for student modeling, which is promising for providing student modeling in a larger scale.

Our effort in this work in treating a question (item) as a KC for BKT and PFA. However, we haven't explored whether using more coarse-grained or fine-grained level KCs would give better prediction performance. Particularly, since PFA is designed for modeling multiple KCs, we need further experiments to compare these models when each item is associated with multiple KCs.

Also, since our study uses user-stratified cross-validation, which requires models to predict for new students, BPTF and BPMF encounter an unfavorable situation, since it is hard to give highly accurate prediction for the student with little or no information for that specific student. We will further explore these models' performance giving different amount of information of students or questions that the model is predicting for.

Obtaining reasonable accuracy of predicting performance for parameterized questions is necessary to investigate how to give recommendations to help students: whether to keep practicing on the current question or to move to another suitable question or to learn from reading an example.

References

1. Brusilovsky, P., Sosnovsky, S.: Individualized exercises for self-assessment of programming knowledge: An evaluation of quizpack. ACM Journal on Educational Resources in Computing 5(3), Article No. 6 (2005)
2. Corbett, A.T., Anderson, J.R.: Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction 4(4), 253–278 (1994)
3. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. ACM SIGKDD Explorations Newsletter 11(1), 10–18 (2009)
4. Hsiao, I.-H., Sosnovsky, S., Brusilovsky, P.: Adaptive navigation support for parameterized questions in object-oriented programming. In: Cress, U., Dimitrova, V., Specht, M. (eds.) EC-TEL 2009. LNCS, vol. 5794, pp. 88–98. Springer, Heidelberg (2009)
5. Kashy, E., Thoennessen, M., Tsai, Y., Davis, N.E., Wolfe, S.L.: Using networked tools to enhanse student success rates in large classes. In: FIE, vol. I, pp. 233–237. Stipes Publishing L.L.C., (1997)
6. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM 51(3), 455–500 (2009)
7. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42(8), 30–37 (2009)
8. Kortemeyer, G., Kashy, E., Benenson, W., Bauer, W.: Experiences using the open-source learning content management and assessment system lon-capa in introductory physics courses. American Journal of Physics 76(438) (2008)
9. Minaei-Bidgoli, B., Kashy, D.A., Kortemeyer, G., Punch, W.F.: Predicting student performance: An application of data mining methods with an educational web-based system. In: FIE 2003 (2003)
10. Parra, D., Sahebi, S.: Recommender systems: Sources of knowledge and evaluation metrics. In: Velásquez, J.D., Palade, V., Jain, L.C. (eds.) Advanced Techniques in Web Intelligence-2. SCI, vol. 452, pp. 149–175. Springer, Heidelberg (2013)
11. Pavlik, P.I., Cen, H., Koedinger, K.R.: Performance factors analysis-a new alternative to knowledge tracing. In: AIEd, pp. 531–538 (2009)
12. Thai-Nghe, N., Horvath, T., Schmidt-Thieme, L.: Context-aware factorization for personalized student's task recommendation. In: Int. Workshop on Personalization Approaches in Learning Environments, vol. 732, pp. 13–18 (2011)
13. Xiong, L., Chen, X., Huang, T.-K., Schneider, J.G., Carbonell, J.G.: Temporal collaborative filtering with bayesian probabilistic tensor factorization. In: SDM, vol. 10, pp. 211–222 (2010)

A Study of Exploring Different Schedules of Spacing and Retrieval Interval on Mathematics Skills in ITS Environment

Xiaolu Xiong and Joseph E. Beck

Department of Computer Science
Worcester Polytechnic Institute
100 Institute Road, Worcester, MA
{Xxiong,josephbeck}@wpi.edu

Abstract. The present study was designed to help answer several questions regarding the impact of spacing and expanding retrieval practice on mathematics skills. For this study, we set up four different interval schedules (1 day; 4 days; 7 days; 14 days) in an ITS environment, and examined the impact on retention performance by comparing results across groups. There were significant performance differences on different groups of students, and all fours groups of students showed small declines in the retention performance with longer intervals. Furthermore, we examined students with high-, medium-, and low-knowledge of skills, and found a strong effect on retention performance with the basis of initial performance on skills. In addition, students with weaker knowledge showed a much more rapid forgetting than students with higher knowledge. These results suggest retention intervals should probably not be fixed, but should vary based on the student's knowledge of the skill.

Keywords: knowledge retention, retrieval practice, spacing effect, intelligent tutoring system.

1 Introduction

Expanding retrieval practice is based on the robust memory phenomenon known as the spacing effect, in which memory for repeated items is better when repetitions spaced apart rather than massed together [5, 6]. In expanded retrieval, these repetitions are spaced increasing intervals, making it necessary to retain the skill for longer and longer amounts of time before one attempt to retrieve it. This effect is specifically important to a cumulative subject as mathematics: we are more concerned with students' capability to remember the knowledge that they acquired over a long period of time.

Inspired by the notion of robust learning [2] and the design of the enhanced ITS mastery cycle (Figure 1) proposed by Wang and Beck [7], we developed and deployed a system called the Automatic Reassessment and Relearning System (ARRS) to make decisions about when to review skills which students have mastered. ARRS is an implementation of expanding retrieval in the ITS environment, Unlike most ITS system [4] which the tutoring stopped if the student mastered a given

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 504–509, 2014.

skill, ARRS assumes that if a student masters a skill with three correct responses in a row, such mastery is not necessarily an indication of long-term retention. Therefore, ARRS will present the student with a reassessment test on the same skill at expanding intervals spread at least 3 months of schedule, that is firstly 7 days after mastery, then 14 days, 28 days and 56 days after the previous test. If a student fails the reassessment test, ASSISTments will give him an opportunity to relearn the skill.

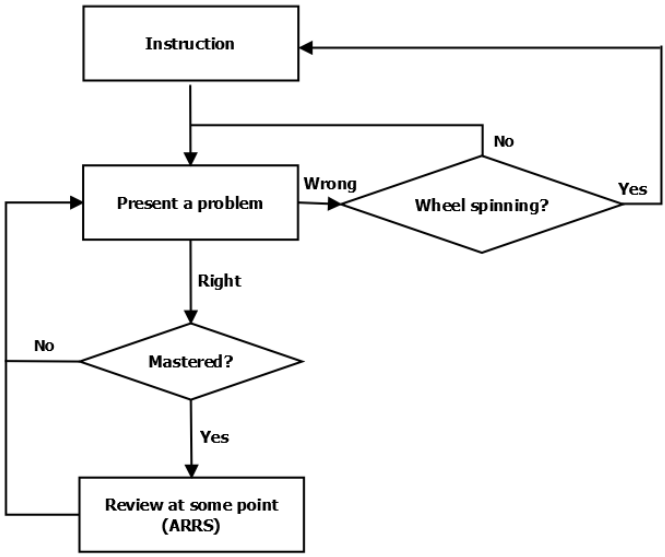


Fig. 1. The enhanced ITS mastery learning cycle

We refer to the number of problems required to achieve mastery as the *mastery speed*, it represents a combination of how well the student knew this skill originally, and how quickly he can learn the skill. We observed that, in general, the slower the mastery speed, the lower the probability that the student can answer the problems in the retention test correctly. Students who mastered a skill in 3 or 4 problems had an 82% chance of responding correctly on the first retention test, while students who took over 7 attempts to master a skill only had a 62%.

Previous studies showed that mastery speed is an extremely important feature for predicting student's retention performance and has a long term effect on students' retention performance [8]. According to these results, we can say that students with different mastery speed have different retention patterns, so we decided to start the exporting the optimal retrieval schedules for different levels of students.

2 An Experiment on Different Schedules of Retention Interval

We first conducted an experiment to investigate how different retention intervals affect student retention performance. There were several objectives for this experiment. A central goal was to investigate knowledge-related differences in terms of spacing and retention interval. As we mentioned before, students who receive retention tests have demonstrated mastery in the initial problem set, which we refer to as the mastery learning problem set. We already observed these students have significantly differences in the fixed-schedule retention tests. Thus, it is worth to find out how mastery speed affects the retention performance given different intervals. This experiment tested students with different retention intervals to explore this question.

The participants were 672 middle and high school students from 34 classes. Teachers of these classes enabled ARRS in ASSISTments voluntarily, and they assigned mathematics mastery learning problem sets according to whatever instructional content they would normally cover in class. Teachers also required their students to use ASSISTments to finish their homework on a daily basis. Students were randomly allocated to one of four conditions which applied with different retention intervals: 174 students were assigned to the 1-day condition, 170 students were assigned to 4-day retention test condition, 162 student and 166 students were assigned to 7-day and 14-day condition. Students worked on their assignments in various environments include school computer labs, home computers and mobile devices. Prior to this experiment, students and teachers already had experiences of using ASSISTments and working with ARRS.

Students were randomly assigned to one of four retention interval conditions: 1-day, 4-day, 7-day, or 14-day. The differences among these conditions were the interval between achieving mastery and receiving the reassessment test. For example, Students in the 1-day condition received the corresponding retention tests the day after they finished the mastery learning problem sets; while students in 14-day conditions received reassessment test 14 days after they finished the mastery learning problem sets. It is important to notice that all reassessment tests were released only on weekdays; this particular behavior of ARRS was designed to cooperate with teachers, and it delayed the assigning of the retention tests which were scheduled to be released on Saturdays and Sundays.

This experiment began on September 15, 2013 and ended on December 15, 2013. During these three months, students constantly received mastery learning problem sets as homework assignments from their teachers. Once they answered three consecutive questions correctly in a mastery learning problem set, a retention test was scheduled based on which condition a student was in and ready to be assigned (e.g., 1, 4, 7, or 14 days after mastery). For mastery learning problems sets, to finish on time, students were required to complete it within one day of when the teacher assigned it. Similarly, for ARRS tests, which were generated by ASSISTments according to the appropriate schedule interval, students had one day to complete these tests. However, it was not uncommon for students to not always complete assignments on time.

3 Results and Discussion

In this study, we asked whether a different retention interval would affect students' retention performance. We were particularly interested in whether or not longer spacing would impede students' retention. In order to determine if different retention interval affected students' performance, we examined students' retention test performance in different conditions.

As we expected, students in longer retention interval had lower retention performance than students in shorter retention interval, but none of the differences are particularly large, even the 1-day performance (80.4%) and 14-day performance (76.0%) only differed by 4.4%. We also noticed that students in the 4 days and 7 days conditions had very close retention performance, namely 77.6% and 77.5%, and this can be explained by the some portion of 4 days retention tests had been delayed one or two days to skip weekends.

When considering whether there were changes in retention performance of students with different mastery speed, we grouped the data by three identified mastery speed bins, then we also examined students' retention test performance. Table 1 shows the retention performance by mastery speed and retention interval.

Table 1. Retention performance by mastery speed and retention interval

	All retention tests (maximizes external validity)		Retention tests completed on time (maximizes internal validity)	
Retention test delay	# tests	% correctness	# tests	% correctness
mastery speed 3 - 4				
1 day	1186	84.4%	462	85.1%
4 days	1169	82.2%	389	84.6%
7 days	1171	81.7%	409	84.1%
14 days	1233	81.2%	419	83.8%
mastery speed 5 - 7				
1 day	467	77.9%	184	75.5%
4 days	432	76.2%	149	73.2%
7 days	362	77.1%	147	72.9%
14 days	420	73.1%	150	72.7%
mastery speed > 7				
1 day	280	67.5%	110	70.0%
4 days	320	62.8%	111	65.8%
7 days	267	59.6%	105	68.6%
14 days	243	54.8%	85	60.0%

The left part of Table 1 shows how students performed on retention tests, and includes data from all students. Including data from all students' results in high external validity as it ensures that our results generalize to other, similar, populations of learners. However, we have seen some tests were completed more than one week later after they were due. Including such data in the study makes it difficult to determine which experimental condition the student was in. How should we analyze students who were in the 7-day condition but completed their retention test 14 days later?

To account for students not being conscientious in completing retention tests on time, we have selected tests which were finished on time (finished no more than one day after released and made available to students). As a result, performance on these tests reflects retention performance on the intervals specified by the study. That is, a student in the 7-day condition was answering his retention test after a delay of between 7 and 8 days, but 14 days would not be possible. Although this approach maximizes internal validity, it also introduces a selection bias. Students who finish their assignments on time are not a random sample of the population, but rather are those who watch their assignment schedules more closely, and those who cared more about finishing assignments on time. These non-random selection effects make these students not perfectly representative of the population as a whole. This tension between internal and external validity is common in field research, and we present both sets of data.

In all students, we have seen consistent decrease in retention performance with longer retention intervals, whether they were high mastery level, medium mastery level or low mastery level students. The results from Table 1 also demonstrated a main effect of mastery speed on retention performance: students with slower mastery speed had significantly lower performance than students with a faster mastery speed ($p \approx 2.2 \times 10^{-27}$); this statement is true even when we comparing 1-day performance of students with slow mastery speed versus 14-day performance of students with fast mastery speed (67.5% for mastery speed > 7 versus 81.2% for mastery speed on 3 or 4). A large and interesting effect is that students with slower mastery speed had larger decrease in retention performance as retention intervals got longer. This interaction effect was statistically reliable ($p \approx 3.4 \times 10^{-22}$). For example, high mastery level student had a decrease of 3.2% between 1 day tests and 14 days tests but retention performance of low mastery level students dropped 12.7%. The horizontal comparisons on Table 1 also suggest that students who finished test on scheduled intervals were more likely to retain skills, confirming our suspicion above about these students not being a representative sample.

4 Contributions, Future Work and Conclusions

As this paper contributes to a large body of literature empirically demonstrating the effects of spaced learning, it makes three unique contributions. First, this paper studied actual effects of spaced learning over long time period for mathematics materials and practices whereas most ITS studies were focused on shorter term and only few

looked effects over time. Second, this experiment investigated the concept of finding the optimal retention interval by using mastery speed for students with different mastery speed. Moreover, this study suggested the necessity of retention tests as a measurement method of robust learning.

Our goal is to find the optimal spacing schedules for students and the best way to boost their performance in long-term mathematics learning; there are so many open problems worth of future research: Is there a better to predict who will retain a skill? Do these mistakes indicate lack of effort or interest on the student's part, or a genuine lack of knowledge? What should we do after students fail a retention test, should we just reply on the connection between well-learned procedural skills and long-term retention [1]? We are also interested in interventions that can decrease the rate of wheel spinning [3]. Most importantly, there are some very challenging problems that we believe can be answered in our following studies. First, do assigning high frequent retention tests and relearning assignments to low knowledge student help to improve their mastery level? And what other tutoring methods we can use if a student fails to retain a skill?

This paper presents the first study of exploring the optimal spacing schedule in learning mathematics skills. With the experiment data we collected, we revealed the relationships between master speed and retention performance in different retention intervals, and most importantly, these relationships will help dictate which learning schedules and memory techniques are most suitable for learning and retrieving.

Acknowledgments. We want to acknowledge the funding on NSF grant DRL-1109483 as well as funding of ASSISTments. See here (http://www.webcitation.org/67MTL3EIs) for the funding sources for ASSISTments.

References

1. Anderson, J.R.: Rules of the mind. Routledge (1993)
2. Baker, R.S.J.D., Gowda, S.M., Corbett, A.T., Ocumpaugh, J.: Towards Automatically Detecting Whether Student Learning is Shallow. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 444–453. Springer, Heidelberg (2012)
3. Beck, J.E., Gong, Y.: Wheel-Spinning: Students Who Fail to Master a Skill. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 431–440. Springer, Heidelberg (2013)
4. Beck, J.E., Jia, P., Sison, J., Mostow, J.: Predicting student help-request behavior in an intelligent tutor for reading. In: Brusilovsky, P., Corbett, A., de Rosis, F., et al. (eds.) UM 2003. LNCS (LNAI), vol. 2702, pp. 303–312. Springer, Heidelberg (2003)
5. Crowder, R.: The effects of repetition. In: Principles of learning and Memory, pp. 264–321. Lawrence Erlbaum Associates, Hillsdale (1976)
6. Melton, A.W.: Repetition and retrieval from memory. Science 158, 532 (1967)
7. Wang, Y., Beck, J.: Using Student Modeling to Estimate Student Knowledge Retention. In: Proceedings of the 5th International Conference on Educational Data Mining, pp. 201–203 (2012)
8. Xiong, X., Li, S., Beck, J.: Will you get it right next week: Predict delayed performance in enhanced ITS mastery cycle. In: The 26th International FLAIRS Conference (2013)

Towards Providing Notifications to Enhance Teacher's Awareness in the Classroom

Roberto Martinez-Maldonado, Andrew Clayphan, Kalina Yacef, and Judy Kay

School of Information Technologies, University of Sydney, NSW 2006, Australia
{roberto,ajc,judy,kalina}@it.usyd.edu.au

Abstract. Students often need prompt feedback to make the best from the learning activities. Within classrooms, being aware of students' achievements and weaknesses can help teachers decide how to time feedback. However, they usually cannot easily assess student's progress. We present an approach to generate automated notifications that can enhance teacher's awareness *in runtime*. This paper formulates the theoretical framing and describes the technological infrastructure of a system that can help teachers orchestrate learning activities and monitor small groups in a multi-tabletop classroom. We define the design guidelines underpinning our system, which include: i) generating notifications from teacher-designed or AI-based sources; ii) enhancing teacher's awareness in the orchestration loop; iii) presenting both positive and negative notifications; iv) allowing teachers to tune the system; and v) providing a private teacher's user interface. Our approach aims to guide research on ways to generate notifications that can help teachers drive their attention and provide relevant feedback for small group learning activities in the classroom.

Keywords: Orchestration, Notifications, F2F Collaboration, Classroom, CSCL.

1 Introduction

Teachers have a crucial role as managers of the different elements of the learning environment [3]. They are responsible for conducting the class design, ensuring productive use of time, resources, learning technologies and providing attention to each student. Students often need scaffolding and prompt feedback on performance to make the best from the learning activities designed by the teacher [14]. Thus, besides *orchestrating* the multiple activities that occur in different dimensions within the classroom, teachers also should provide feedback to the students. The provision of feedback is an essential part of effective learning achievement [14, 15]. Being aware of students' progress, achievements and weaknesses can help teachers enhance the provision of timely and effective feedback [13]. In the classroom, this support should ideally be provided while the learning activity is still underway, so necessary adjustments can be made. However, even though providing timely feedback is very important, teachers can easily become absorbed with their multiple *orchestration* responsibilities, making it difficult for them to attend to the students who need it most.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 510–515, 2014.

Fig. 1. Left: Multi-tabletop pervasive classroom for small group collaboration. Right: Notifications in the teacher's dashboard.

A wide range of learning technologies have been used over the past years to improve instruction and learning in the classroom [11]. However, the development of tools to effectively support teacher's awareness and help them provide enhanced feedback to students has been relatively neglected [2, 8]. Research suggests that some sort of integrated assessment is needed in order to give effective feedback [1]. Unfortunately, teachers often cannot assess the quality of students' artefacts, partial outcomes, student's performance or their collaborative interactions *on-the-fly*. This opens up an opportunity to exploit the use of emerging technology that can unobtrusively capture aspects of students' activity and then automatically alert teachers about events that are hard to assess within the time constraints of a class.

We propose an approach to automatically generate notifications for teachers in a timely manner during a class. Our system is implemented in a classroom enhanced with pervasive technologies: the MTClassroom [8]. This learning environment is ideal to plan and orchestrate small group activities by exploiting the affordances of five horizontal and three vertical shared devices (Figure 1, left). In our setting, the notifications can be generated by assessing, in real-time, qualitative aspects of the knowledge artefacts being built by the students (in the form of concept maps) and comparing them with a model of expert knowledge and a set of misconceptions defined by the teacher. Alternatively, notifications can be generated based on quantitative aspects of student's collaboration that may be associated with undesired patterns of interaction, such as social loafing [10] or strategies of low achieving groups [9].

2 Related Work

Previous research has delivered tools that enhance teacher's awareness and reflection through different dashboards or visualisations. Verbert et al. [16] observed that these kinds of tools have been deployed in three learning contexts: online learning settings, face-to-face lectures and face-to-face small group work within classrooms. Teacher's awareness tools and the automated generation of notifications have only been explored in online learning settings [16]. For the very different case of face-to-face work in classrooms, an important example tool is the Tinker Board [3], that shows

information on a large display to support reflection on a small-group activity. Similarly, the Tinker Lamp [3] is a widget that students can use to indicate to the teacher which stage of the activity they are up to. Martinez-Maldonado et al. [8] explore the impact of showing visualisations of student's data to help teachers decide where to place her attention. Other studies have agreed that this type of information can be useful for teachers using both private or public displays [4] and can be integrated with different devices like tablets [5], smart-boards [3] and tabletops [8]. Our approach is the first effort we are aware of that describes guidelines to build a system that provides teachers with automatic notifications in the classroom.

3 Context of the Learning Environment

As a foundation to define our approach, we first introduce the context of the learning environment. Then we describe an example authentic scenario. We built our system targeting university level learning activities for tutorial classes that can be held in the MTClassroom [8]. This is the first classroom with multiple interactive tabletops that can (i) unobtrusively capture data about each learner's activity, linking it to the learner's identity; and (ii) provide orchestration tools and real-time student's data analysis. It is composed of 5 interconnected multi-touch tabletops, each well suited for face-to-face work in groups of up to 5 students and enhanced with the CollAid [7] sensing system. Each tabletop records the activity of students within each group to a central repository that can be accessed by other services in real-time. One of these services generates visual indicators to enhance teacher's awareness and shows them in the MTDashboard. The MTDashboard is displayed on a handheld device that allows teachers to orchestrate the MTClassroom (Figure 1, right). One of the applications that can be used in this learning environment is CMate [7]. This is a concept mapping tool that records activity logs, traces of the task progress and information about student's maps. A concept map is a directed graph in which nodes represent the main *concepts* of the subject matter and the edges are labelled with a linking word to form meaningful statements called *propositions* [12]. More information about the environment can be found in the technical papers of CMate [7] and MTClassroom [8].

An example study where our approach can be applied was conducted during an undergraduate course on *Human-Computer Interaction* (HCI). A total of 95 students were enrolled in this subject. Students were divided at the beginning of the semester into 6 tutorial classes, with around 15 students each. Each tutorial was facilitated by a class teacher. In the example course, the students were organised into 24 groups of 3, 4 or 5, who worked together during the tutorial sessions. The same 1-hour weekly tutorial ran in each tutorial, with three different class teachers (the main teacher had one class and the other 2 class teachers had 2 and 3 classes). The learning goals for students using CMate commonly consist of collaborating with their group to create a concept map that answers a *focus* question. In this way, the teacher needs to monitor up to five small groups building five different concept maps in parallel. It is not easy for the teacher orchestrating the class while, at the same time, assessing each concept map to know if students are building a high quality map or have misconceptions.

4 Approach and Design Guidelines

Figure 2 illustrates the context where our approach is deployed and the process that the teacher can follow to design, enact and diagnose the classroom activity. We describe this process in terms of the design guidelines underpinning our system.

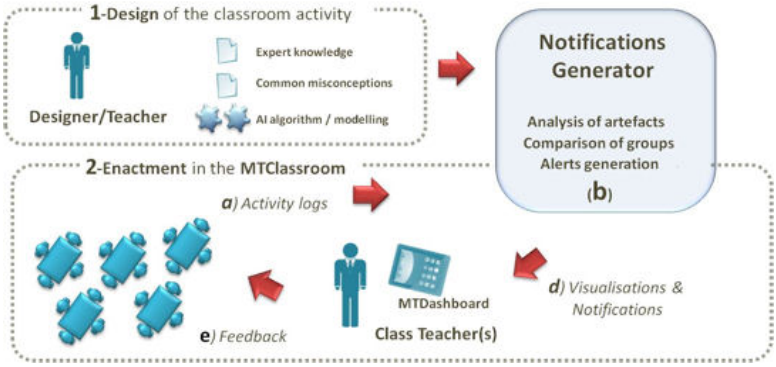


Fig. 2. MTFeedback context: conceptual diagram of the approach

i) Generating notifications from on teacher-designed or AI-based sources. This process starts with the teacher designing the learning activity before the classroom sessions (1). In this stage, the main teacher designs a macro-script for the sessions. The teacher can define a source of expert knowledge and common misconceptions that can be used to match student's artefacts automatically. In our study, these can be defined by the teacher using CMapTools, a third-party widely used concept map editor [12]. The expert knowledge is represented as a concept map that contains the propositions that the teacher considers the students should have in their maps. Common misconceptions that the teacher wants to track are defined separately, as a set of propositions. The teacher can also select AI-based sources to generate notifications that may consist of matching students' activity logs with patterns of interaction associated with either high or low collaboration groups [9]. For example, our previous work, using sequence mining on tabletop touch data, found that it is possible to identify high collaborative groups which often work in parallel, interacting with other students' objects and focus on the crucial elements of the problem to solve [9].

ii) *Enhancing teacher's awareness in the orchestration loop*. The designed activity is then enacted in the classroom (2). In the classroom, teachers commonly follow an orchestration loop [3] where small group activities can be described as follows: the teacher monitors the groups, assesses their performance to decide which group(s) may need support, attends to the chosen group and starts the cycle again. Our approach aims to support the teacher's decision making about which groups most need their immediate attention by enhancing their awareness of each group's progress in this orchestration loop. We describe this as the following process: a) The pervasive interactive tabletops capture, synchronise and gather activity logs of each group in a central repository; b) our system compares each group's logged activity against the expert knowledge and the list of misconceptions; c) notifications may be generated

accordingly and sent to the MTDashboard; d) the MTDashboard interface shows visualisations and notifications to the teacher; and e) the teacher looks at the dashboard and decides whether a certain group(s) needs feedback or not.

iii) Presenting both positive and negative notifications. It has been found that receiving too many notifications can produce a negative effect on users, as it makes it hard to readily determine what has changed over time [6]. To avoid this, we give two types of negative notifications and one positive notification; the choice of these was defined by the main teacher's pedagogical requirements. First, a *Misconception Notification* (MN) is triggered for the group that has the most misconceptions in the classroom. Our system assesses groups every half a minute, deciding which group, if any, needs a new notification generated. This way, the teacher can eventually determine whether all groups have recurrent misconceptions or if the whole class needs a clarification of the activity. Similarly, the system may provide alerts when patterns of either high or low collaboration are detected for certain groups.

iv) Allowing teachers to tune the system. It is important to allow the teacher to tune or configure the rules used to generate notifications as well as the timing or pace in which they are displayed on the teacher's dashboard. For example, a second negative notification is the *Slow-Group Notification* (SN). For this, our system compares the progress of all the groups in the classroom and flags a group as being left behind if it has less than half of the propositions created by the top achieving group. This rule was tested on a dataset collected in sessions run in previous semesters [8] but it can be tuned by the teacher. By contrast, a *Positive Notification* (PN) can be generated when a group had at least P% of their propositions matching the expert knowledge (the parameter P was tuned by the teacher to be 50% in our study).

v) Providing a private teacher's user interface. Figure 1 (right) shows the teacher's dashboard as displayed on a handheld tablet. The interface has a set of buttons for the teacher to control the classroom technology. It also features up to five visualisations, each associated with an active tabletop. Inspired by [8], we used visualisations that indicate the size of each group's solution and the proportion that matches the expert knowledge (a figure with an outer and an inner circle respectively). The notifications appear as a square (red for negative or green for positive) around the group information. For example, Figure 1 (right) shows two notifications: a negative notification for the group with the most misconceptions (red square around the *blue table*), and a positive one for a group that included half of the expert knowledge and had no misconceptions (green square around the *red table*). Teachers can be instructed to get a message on the screen with more information about the notification by tapping inside of the coloured square.

5 Conclusions

Providing teachers access to automatically captured data can enhance their awareness [4], however, effective ways to show this information in the classroom are still needed. This paper describes the theoretical and technological framing, and the design guidelines, of a system that can provide teachers with notifications of small group work in runtime. We aim to exploit student's data captured at a multi-tabletop classroom to alert the teacher about aspects of student's collaboration, and their solutions, that cannot be easily analysed by the teacher on-the-fly. This can help

teachers provide immediate or delayed feedback. Even though our approach can be generalised to other kinds of small group learning activities, this paper presented a learning context in terms of a collaborative concept mapping activity as an instance of the application of our approach. The sources to generate notifications can be simple (expert knowledge and misconceptions defined by the teacher). Our future work will investigate the impact of using our system with real teachers and students, in-the-wild. Future work should also consider the use of machine learning techniques that can detect potential problems in groups to alert teachers proactively.

References

1. Brown, S.: Assessment for learning. Learning and Teaching in Higher Education 1(1), 81–89 (2004)
2. Bull, S., Wasson, B., Johnson, M.D., Petters, D., Hansen, C.: Helping Teachers Effectively Support Group Learning. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, Springer, Heidelberg (2012)
3. Dillenbourg, P., Zufferey, G., Alavi, H., Jermann, P., Do-Lenh, S., Bonnard, Q.: Classroom orchestration: The third circle of usability. In: Proc. CSCL 2011, pp. 510–517 (2011)
4. Kharrufa, A., Martinez-Maldonado, R., Kay, J., Olivier, P.: Extending tabletop application design to the classroom. In: Interactive Tabletops and Surfaces 2013, pp. 115–124 (2013)
5. Kreitmayer, S., Rogers, Y., Laney, R., Peake, S.: UniPad: orchestrating collaborative activities through shared tablets and an integrated wall display. In: Proc. Ubicomp 2013, pp. 801–810 (2013)
6. Lurie, N.H., Swaminathan, J.M.: Is timely information always better? The effect of feedback frequency on decision making. Organizational Behavior and Human Decision Processes 108(2), 315–329 (2009)
7. Martinez-Maldonado, R., Collins, A., Kay, J., Yacef, K.: Who did what? who said that? Collaid: An environment for capturing traces of collaborative learning at the tabletop. In: Interactive Tabletops and Surfaces, pp. 172–181 (2011)
8. Martinez-Maldonado, R., Dimitriadis, Y., Kay, J., Yacef, K., Edbauer, M.-T.: MTClassroom and MTDashboard: supporting analysis of teacher attention in an orchestrated multi-tabletop classroom. In: Proc. CSCL 2013, pp. 119–128 (2013)
9. Martinez-Maldonado, R., Yacef, K., Kay, J.: Data Mining in the Classroom: Discovering Groups' Strategies at a Multi-tabletop Environment. In: Proc. EDM 2013, pp. 121–128 (2013)
10. Martinez-Maldonado, R., Yacef, K., Kay, J., Schwendimann, B.: Unpacking traces of collaboration from multimodal data of collaborative concept mapping at a tabletop. In: Proc. ICLS 2012, pp. 241–245 (2012)
11. Muir-Herzig, R.G.: Technology and its impact in the classroom. Computers & Education 42(2), 111–131 (2004)
12. Novak, J., Cañas, A.: The Theory Underlying Concept Maps and How to Construct and Use Them. In: Florida Institute for Human and Machine Cognition (2008)
13. Race, P.: Using feedback to help students learn. HEA, York (2001)
14. Shute, V.J.: Focus on formative feedback. Review of Educational Research 78(1), 153–189 (2008)
15. Stronge, J.H.: Qualities of effective teachers. Association for Supervision and Curriculum Development, Alexandria, VA (2007)
16. Verbert, K., Govaerts, S., Duval, E., Santos, J., Assche, F., Parra, G., Klerkx, J.: Learning dashboards: An overview and future research opportunities. PUC, 1–16 (2013)

Survey Sidekick: Structuring Scientifically Sound Surveys

I-Han Hsiao[1], Shuguang Han[2], Manav Malhotra[1], Hui Soo Chae[1], and Gary Natriello[1]

[1] EdLab, Teachers College Columbia University, New York City, NY, USA
{ih2240,mm2625,hsc2001,gjn6}@columbia.edu
[2] School of Information Sciences, University of Pittsburgh, Pittsburgh, PA, USA
shh69@pitt.edu

Abstract. Online surveys are becoming more popular as a means of information gathering in both academia and industry because of their relatively low cost and delivery. However, there are increasing debates on data quality in online surveys. We present a novel survey prototyping tool that integrates embedded learning resources to facilitate the survey prototyping process and encourage creating scientifically sound surveys. Results from a controlled pilot study confirmed that survey structure follows three guided principles: simple-first, structure-coherent and gradual-difficulty-increase, revealing positive effects on survey structures under learning resources influences.

Keywords: Survey Design, Hidden Markov Model, Ill-defined domain.

1 Introduction

The web has lowered the barrier to collect information through surveys [1]. Survey Monkey[1], one of the most popular online survey tools, has successfully created more than 15 million online surveys. However, until today, most of the online survey tools mainly focus on the support of survey delivery and simple analytics, neglecting the quality of the survey. For experienced survey researchers, they can rely on their expertise and experience to ensure survey validity and reliability. Non-experienced survey creators may be at a disadvantage from a lack of feedback or guidance, unknowingly creating biased and incomplete surveys. In this paper, we study and report an innovative solution to encourage creating scientifically sound surveys.

In traditional Artificial Intelligence, intelligent tutoring systems have succeeded in automatically providing feedback for problem solving and direct instructions, in the form of examples or definitions of the concepts [2] or auto-grading [3]. Recent Intelligent Tutoring Systems face new challenges due to the increasing importance of interdisciplinary study in ill-defined domains, where there is no guarantee of getting quick and sound feedback and the quality of answers is difficult to evaluate. For instance, reasoning legal arguments [4], providing semantic and constructive feedback for survey design [5], and programming assignments [3, 6] among others.

[1] Survey Monkey: https://www.surveymonkey.com

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 516–522, 2014.

Constrain-based tutors are studied to effectively provide feedback for ill-defined problems. However, it is time-consuming to manually generate constraints for a broad domain [9,10]. A less costly way of receiving constructive feedback is to obtain answers from online Q&A systems [11], crowdsourcing, [12] or collecting community feedback through a systematic peer review process [13]. However, these approaches still present several challenges such as low answering rate [14], answer quality ambiguity [15], and among others. To address these new challenges and move from automatic assessment to a more data-driven approach for feedback generation, there are techniques such as considering probabilistic distance to solution for assessing the progress to identify misconceptions or the problem solving path [6], forms of latent semantic analysis (LSA) for automatic evaluation and topic mapping [3]. QUAID [16] is one of the few web tools that assist survey methodologists in examining survey questions such as wording, syntax, and semantics of questions. Our focus is on survey structure and adaptive learning resources during the survey prototyping process. Applying previous research conclusions into our study, we hypothesize that embedded learning resources and providing automatic hints [8] during survey prototyping process with dynamic survey structure modeling will enhance survey design quality. Before providing effective feedbacks, understanding user behaviors in creating survey also helped better suggesting learning resources. To research all the issues addressed above, we present an innovative system – Survey Sidekick and study the effectiveness of our approach.

2 Survey Sidekick

Survey Sidekick (https://surveysidekick.com) is an online survey tool developed by EdLab, Teachers College Columbia University. The beta version was launched in October 2012 and is currently open via invitation. We currently have 444 users, with 102 of them designing one or more surveys. Survey Sidekick supports design, delivery, data analytics and reporting. The system includes embedded learning resources (orange icons) and interface support (blue icons) (Figure 1). Both are displayed at the relevant moment during prototyping process at the side of the questions or the entire survey. Embedded learning resources are tutorials extracted from a survey design textbook [17]. Further design rationale is reported in [5]. In this work, we extend the dynamic learning resources support by evaluating the survey structure & question composition.

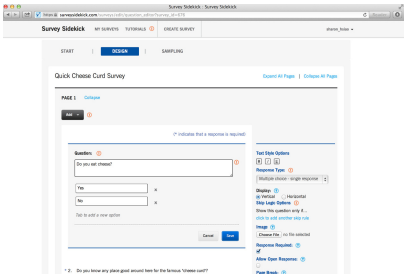


Fig. 1. Survey editing interface; https://surveysidekick.com

3 Modeling Sequential Survey Structure Using HMM

The Hidden Markov Model (HMM) is a popular method for modeling sequential data. In this study, we employ the HMM to model users' hidden tactics in designing a survey, and use the choice of each answer type (e.g. free text, Likert, and multiple choice) as the outcome of the hidden tactics. The hidden *tactics* together can be thought of as the *strategy* used to design the survey. A similar study is conducted by Yue et al. [18] in understanding users' information seeking behavior. In Survey Sidekick, there are 7 different types of question/answer types supported. They are: Numeric (N), Free text (F), Short answer (S), Multiple choices with single correct answer (MS), Multiple choices with multiple correct answers (MM), Likert with 5-value scale (LI), and Likert list with more than 5-values (LL).

We have a sequence of survey answer types from T_1 to T_M, and each is from the predefined set: **TS** = {N, F, S, MS, MM and LI}. HMM assumes that we also have a sequence of hidden states, from H_1 to H_M, and each answer type is generated by a corresponding hidden state, but different answer types can be generated by the same hidden state with different probabilities. A HMM model has several parameters: the number of hidden states HS, the start probability of each state $\boldsymbol{\pi}$, the transition probabilities among any two hidden states $\boldsymbol{A_{ij}}$ and the emission probability from each state to each action $\boldsymbol{b_{ij}}$.

4 Evaluation and Results

We randomly selected the training dataset from Survey Sidekick, which contains 38 surveys with 1,048 questions. For the test data set, we recruited 22 subjects and randomly assigned them into control and treatment groups to design a survey on the same scenario [5], where control group had no learning resources access and treatment group did. All the usage logs, including the survey content (questions, questions types, survey layouts, survey administration) and learning resources usage (modules, sub-modules, access points: static list view or dynamic box view) were collected.

4.1 Survey Structure Analysis

The first step of using HMM is to determine the number of hidden states, which also refers to the model selection problem. A complex model with large number of states will help to increase the sequence likelihood, as there are more parameters that can be used to describe the model more precisely. The tradeoff is a high risk of over-fitting. We chose hidden state (HS=7) because it had the best performance under Akaike Information Criterion (AIC). The emission probability of each hidden state and transition probability are shown in Table 1, in which the probabilities under 0.05 were removed for clarity of presentation. The hidden states can be thought as the underlying "*tactics*" or "*strategies*" the surveyors use to design their survey. For example, in HS2, the designers focused on collecting information based on the Likert questions, while in HS5, the designers tend to collect data either using Likert questions or

multiple choices questions. However, some hidden states (e.g. the HS1) also have high probabilities of generating both Likert question and free text questions, which suggests they make alternative choices for collecting the same type of information.

Table 1. The Hidden States of Survey Structure ($\boldsymbol{b_{ij}}$)(left); Transitions among the hidden states ($\boldsymbol{A_{ij}}$) (right)

	N	LI	MM	MS	F	S
HS1		0.41			**0.60**	
HS2		**0.84**			0.07	0.09
HS3				0.12	**0.85**	
HS4			**0.53**	0.37		
HS5		**0.43**	0.15	0.42		
HS6			**0.63**			0.35
HS7	0.09			**0.86**		

	HS 1	HS 2	HS 3	HS 4	HS 5	HS 6	HS 7
HS 1	**0.84**		0.10			0.07	
HS 2		**0.91**				0.07	
HS 3			**0.61**	**0.34**			0.05
HS 4	0.10			**0.77**			0.14
HS 5					**0.93**		0.07
HS 6			**0.28**		0.05	**0.41**	**0.24**
HS 7						**0.35**	**0.61**

4.1.1 Simple First Principle

HS7 has a high prior probability (start probability), which means that surveys usually begin with HS7, asking a numeric question or simple multiple-choice questions (i.e. a demographic question with numeric or multiple-choice question type such as what is date of birth). HS2 also has a reasonable start probability, in which the Likert questions or short answer questions may be asked. Moreover, the prior probability also indicates the complex question type (free text) is less likely to appear as the survey starters (HS1 & HS3). The result aligned with literature in designing the opening survey where Iarossi [7] suggested using simple questions to begin the survey.

4.1.2 Structure Coherence Principle

The probability in each diagonal cell is the highest in each row, which suggests an interesting fact in the survey designing process: the same types of questions tend to be used closely together. It indicates a consistency among sub-sections. One of the biggest benefits of designing a survey this way is that the structure coherence may help designers reduce cognitive load that caused by switching between different types of questions. Such finding is again supported by the design principle proposed by Iarossi [7]: finishing one topic before raising a new topic, which focused more on the content consistency, but the HMM structure also strongly suggests consistency at the structure level. In addition, maintaining survey structure consistency appears to be a more manageable task when the survey involves *skip logic*, or detailed questions.

4.1.3 Gradual Difficulty Increase Principle

We also observed several inter-type transitions: HS6→HS3, HS6→HS7, HS7→HS6 and HS3→HS4. Take HS7→HS6 for instance, after raising the opening questions (HS7), the designers may continue asking simple short answer questions or more difficult multiple-choice questions. To give a concrete example, after demographic

questions (usually numeric type) or *skip logic* questions (usually multiple choice with single correct answer) are asked, a more difficult multiple-choice question or free-text based question is likely to be extended to solicit more in-depth information from the survey takers. If however the short answer type questions were asked, the next step will either stay in the same state (self-transition), go back and ask another round of simple questions (HS6→HS7), or ask even more difficult questions, e.g. the open-ended questions (HS6→HS3). The transition from HS3→HS4 (free text question to multiple-choice question) also suggests that the designers tend to choose to ask even more in-depth questions for the open-ended questions. In addition, we found that HS1, HS2 and HS5 are less likely to transit to other survey hidden states. Their correspondent question types, such as Likert and free text questions appear to be at the very end of the entire survey.

4.2 Effects of Learning Resources

To evaluate how learning resources affect on survey prototyping structures, we looked at the topics accessed by users, survey question types, question text, survey layout edits and moves. We found that on average every user in learning-resources-enabled group studied 5.18 topics, and 57 topics in total. They had significantly more *moves*, or structural edits, ($p<0.05$) in survey design. However, did they learning resources they studied actually affect the moves? We found that among all the topics studied by the users, a large portion (49.12%) of the topics were about survey structures, including *Survey Layout, Question Structure, Skip Logic.* Based on structural principles found in section 5.1, we calculated the question sequencing likelihoods for both groups. The learning-resources-enabled group was found to have 10.57% of question sequences in line with the structural principles. On the other hand, the controlled group only achieved 1.69% in line with structural principles. This demonstrates the learning resources' positive effect on users' decision-making process with respect to survey structure.

5 Discussion and Limitation

The study results showed the hidden variables of HMM can uncover users' latent "factors" in the survey design process. The inter-type transitions provide valuable information on improving survey design. More importantly, three survey design principles were verified: **simple-first**, **structure-coherent** and **gradual difficulty increase**. Such principles allow us to predict the survey structure and provide valuable feedback when designers distort the sequential difficulty of survey questions or put too many transitions between multiple survey question types. We also recognize several limitations of this study: 1) current baseline model only used the complete surveys in Survey Sidekick, and therefore it did not fully take the expertise of the survey content creator into account. Thus, the results may not be indicative of best practices. 2) We interpreted a scientifically sound survey as equivalent to a structurally sound survey. We did not measure the survey reliability and validity in current

study design. However, we believe that the Survey Sidekick has attempted to address such issues by designing "official-ness" [5] and other features in the system.

References

[1] Evans, J.R., Mathur, A.: The value of online surveys. Internet Research 15(2), 195–219 (2005)

[2] Aleven, V., Koedinger, K., Cross, K.: Tutoring answer explanation fosters learning with understanding. In: Lajoie, S., Vivet, M. (eds.) Artificial Intelligence in Education, pp. 199–206 (1999)

[3] He, Y., Hui, S.H., Quan, T.T.: Automatic summary assessment for intelligent tutoring systems. Computers & Education 53(3), 890–899 (2009)

[4] Pinkwart, N., Ashley, K.D., Lynch, C., Aleven, V.: Evaluating an Intelligent Tutoring System for Making Legal Arguments with Hypotheticals. To appear in International Journal of AI in Education; Special Issue on Ill-Defined Domains 19(4) (2009), Aleven, V., Lynch, C. Pinkwart, N., Ashley, K. (eds)

[5] Hsiao, I., Malhotra, M., Joo, J., Chae, H.S., Natriello, G.: Survey Sidekick: Learning & designing scientifically sound surveys. In: Paper for "Human-Computer Interaction and the Learning Sciences" Workshop, CSCL 2013, Madison, WI (2013)

[6] Sudol, L.A., Rivers, K., Harris, T.K.: Calculating Probabilistic Distance to Solution in a Complex Problem Solving Domain. In: EDM, pp. 144–147 (2012)

[7] Iarossi, G.: The Power of Survey Design: A User's Guide for Managing Surveys, Interpreting Results, and Influencing Respondents. The World Bank, Washington, D.C (2006)

[8] Barnes, T., Stamper, J.: Automatic hint generation for logic proof tutoring using historical data. Journal of Educational Technology & Society, Special issue on Intelligent Tutoring Systems 13(1), 3–12 (2010)

[9] Le, N.-T., Menzel, W.: Using Constraint-Based Modelling to Describe the Solution Space of Ill-defined Problems in Logic Programming. In: Advances in Web Based Learning (ICSL 2007), pp. 367–379 (2007)

[10] Mitrovic, A.: An Intelligent SQL Tutor on the Web. International Journal of Articial Intelligence in Education 13(2-4), 173–197 (2003)

[11] Kittur, A., Chi, E.H., Suh, B.: Crowdsourcing user studies with Mechanical Turk. In: Proc. of CHI 2008 (2008)

[12] Snow, R., O'Connor, B., Jurafsky, D., Ng, A.Y.: Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Honolulu, Hawaii, October 25-27 (2008)

[13] Hsiao, I., Brusilvsky, P.: The Role of Community Feedback in the Student Example Authoring Process: An Evaluation of AnnotEx. British Journal of Educational Technology 42(3), 482–499 (2011)

[14] Liu, Z., Jansen, B.J.: Factors influencing the response rate in social question and answering behavior. In: Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW 2013), pp. 1263–1274. ACM, New York (2008)

[15] Harper, F.M., Raban, D., Rafaeli, S., Konstan, J.A.: Predictors of answer quality in online Q&A sites. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2008), pp. 865–874. ACM, New York (2008)

[16] Graesser, A.C., Cai, Z., Louwerse, M.M., Daniel, F.: Question Understanding Aid (QUAID) A Web Facility that Tests Question Comprehensibility. Public Opinion Quarterly 70(1), 3–22 (2006)
[17] Dillman, D.A.: Mail and internet surveys: The tailored design method, vol. 2. Wiley, New York (2000)
[18] Yue, Z., Han, S., He, D.: Modeling search processes using hidden states in collaborative exploratory web search. In: Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing, pp. 820–830 (2014)

Authoring Tools for Collaborative Intelligent Tutoring System Environments

Jennifer K. Olsen[1], Daniel M. Belenky[1], Vincent Aleven[1], Nikol Rummel[1,2], Jonathan Sewall[1], and Michael Ringenberg[1]

[1] Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA
jkolsen@cs.cmu.edu, dbelenky@andrew.cmu.edu,
{aleven,sewall,mringenb}@cs.cmu.edu
[2] Institute of Educational Research, Ruhr-Universität Bochum, Germany
nikol.rummel@rub.de

Abstract. Authoring tools have been shown to decrease the amount of time and resources needed for the development of Intelligent Tutoring Systems (ITSs). Although collaborative learning has been shown to be beneficial to learning, most of the current authoring tools do not support the development of collaborative ITSs. In this paper, we discuss an extension to the Cognitive Tutor Authoring Tools to allow for development of collaborative ITSs through multiple synchronized tutor engines. Using this tool, an author can combine collaboration with the type of problem solving support typically offered by an ITS. Different phases of collaboration scripts can be tied to particular problem states in a flexible, problem-specific way. We illustrate the tool's capabilities by presenting examples of collaborative tutors used in recent studies that showed learning gains. The work is a step forward in blending computer-supported collaborative learning and ITS technologies in an effort to combine their strengths.

Keywords: Problem solving, collaborative learning, intelligent tutoring system, authoring tools.

1 Introduction

While most Intelligent Tutoring Systems (ITSs) are geared towards individuals, there has been some evidence that collaborative ITSs are also beneficial [5-6], [14]. ITSs take advantage of features, such as step-based guidance and hints, to support successful learning [12] while Computer Supported Collaborative Learning (CSCL) environments provide support for learning through collaboration scripts, which provide structure for tasks and interactions within a group, and help support the development of mutual understanding and explanation [3]. Despite these benefits, the combination of the two may not be more widely used because of a lack of effective and flexible authoring tools for creating collaborative learning opportunities within ITSs [8]. While there has been ongoing work to develop collaboration tools to make collaboration scripts more accessible and easier to use across learning domains [1], [7], [11], [13], these tools often do not take advantage of beneficial ITS features. We have

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 523–528, 2014.

created a tool that flexibly supports the use of collaboration scripts while also providing support for ITS features by extending an existing ITS authoring tool, the Cognitive Tutor Authoring Tools (CTAT) [2].

To demonstrate the utility of the tool, we will present examples from experiments we have run where we have created learning opportunities based on *collaboration scripts*. According to Dillenbourg [3], a collaboration script consists of a set of phases where each phase has five attributes: the task, the group composition, the distribution of the task (this includes who gets what information *and* who does what, such as through roles), the mode of the interaction, and the order of the phases. Any of these attributes can change between phases, and to allow for flexibility in the scripts developed, an authoring tool needs to support each of these attributes independently so the script can dynamically change with the problem state.

The enhancement to CTAT described in this paper supports the development of ITSs that contain these attributes. Authors can create collaborative ITSs by embedding various problem-specific features that trigger dynamically, based on the problem state, to move students through different phases of the collaboration script, all without programming. In this paper, we provide examples of collaborative script phases (i.e. cognitive group awareness [4] and sharing unique information) developed using CTAT. These examples were used in two "pull-out studies," run in three elementary schools, with a total of about 70 participating students in collaborative conditions and illustrate the flexibility of authoring collaborative tutors.

2 Authoring Tool Extensions to Support Collaboration

In this section, we describe how one type of ITS, a collaborative example-tracing tutor [2], can be authored with CTAT. Similar to how tutors for individual learners are developed, an author creates two key components: A user interface designed for the problem being tutored (in Flash) and a behavior graph (in the CTAT software), which stores all of the acceptable solution paths and commonly-occurring incorrect steps. Behaviorally, example-tracing tutors are similar to other types of ITSs, providing all the key functionality defined by VanLehn [12], and below we describe how the CTAT extension has allowed communication between tutors for collaboration.

2.1 Authoring Collaborative Tutors

To expand CTAT so it supports *collaborative example-tracing tutors*, we added the capability to run *multiple synchronized tutor engines,* one for each student in a collaborating group (see Figure 2). It is important to note that any number of tutor engines can be run in synchronized fashion. Specifically, for any given problem in a collaborative tutor, there is a separate behavior graph file per collaborating student and a separate interface file. The collaborative version of CTAT allows authors to synchronize the tutors so that it can maintain a problem state that is in sync between tutor engines (and between collaborating students). When one of the collaborating students takes an action, this input is sent to both the student's tutor engine and their partner's tutor engine.

By contrast, tutor output is only sent to the corresponding student interface. One result of this input sharing is that student actions taken on one interface will be "mirrored" on the other interface in the corresponding interface component. Yet this set up also allows for differentiation in the tutor feedback provided to collaborating partners, for example by means of unique feedback, individualized hints, information based on roles, and different sets of available actions at any given point in a problem. This set-up allows for great flexibility in authoring tutors with embedded collaboration scripts. In particular, the power of the approach comes from being able to craft tutors in which the collaborators have different views on the same problem and tasks are distributed across collaborators, so as to structure and support their different roles according to particular collaborative phases in a collaboration script. There are many collaboration features, such as the cognitive group awareness and unique information described below, as well as other scripts such as the jigsaw and the tutee/tutor paradigm, where the benefit of the activity comes from the students having different roles and responsibilities in the problem-solving task.

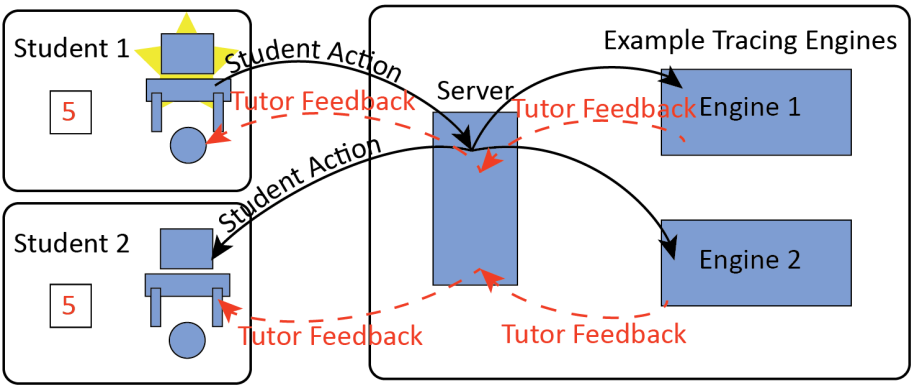


Fig. 1. Diagram displaying the communication between two synchronous tutor engines. A student interface action is shown with the solid line and feedback is shown with the dotted line. Student 1 has entered a 5 into the interface, which has been distributed to both example-tracing engines, and each student has received individual feedback based on the result.

To author a collaborative tutor, each of the steps to create an individual tutor is followed for each member of the collaboration, typically, in interleaved fashion. First, an author creates an interface through drag-and-drop with Flash. Each interface can be identical or designed to match the student's roles. Once an interface is created, an author creates a behavior graph by demonstrating problem-solving behavior on the interface. After the behavior graph is created through demonstration, the graph can be annotated with hints and error feedback messages. The hints and feedback provided can be the same for each student or can be customized for each student. To author collaborative tutors, CTAT allows multiple behavior graphs to be open simultaneously and to each connect to their own student interface. This allows authors to test the collaboration and synchronization of the tutor engines.

3 Collaboration Examples Using CTAT for Collaboration

Below we describe two examples of collaborative phases, which have been shown to be successful [4], to demonstrate the flexibility of CTAT in supporting different collaboration features. Specifically, building on our prior work on the Fractions Tutor [10], we created a collaborative tutoring system to help elementary students learn fractions. In three school studies, we have observed positive learning gains related to tutor usage [9]. As students use the tutor, they each sit at their own computer and communicate via Skype. The two examples illustrate the types of collaboration features that can be implemented within a ITS using the collaborative version of CTAT.

The first example demonstrates a task that supports cognitive group awareness, in which the students are learning conceptual knowledge about equivalent fractions. Cognitive group awareness refers to having information about group members' knowledge, information, or opinions, and sharing of this information has been shown to help guide collaboration [4]. In this example, cognitive group awareness is combine with step-by-step support for problem solving as follows: First, the collaborating partners each answer the same question separately; then, the tutor displays both partners' answers to promote discussion; and, finally, the partners provide a final answer endorsed by both (see Figure 1, panel A1). The students are not given feedback on their individual answer but are shown what their partner selected and are asked to select the correct answer as a pair. This allows each student to see their partner's understanding of the question before discussing and choosing a group answer.

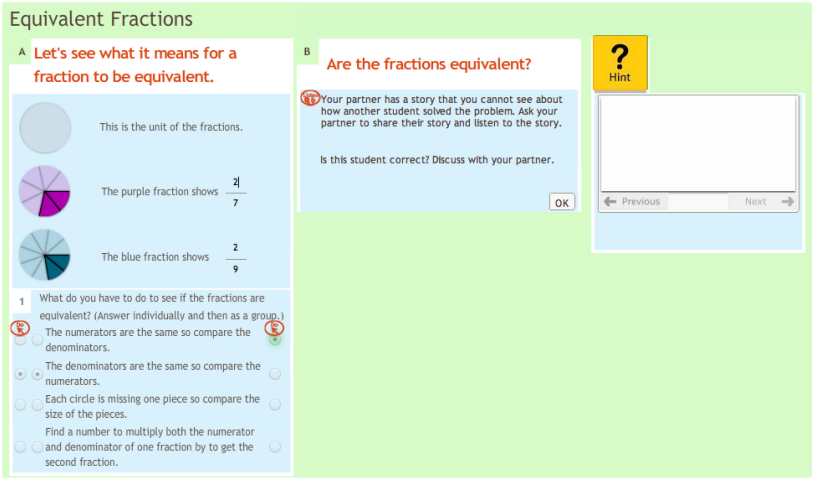


Fig. 2. Example conceptual tutor problem. Panel A1 displays an example of support for cognitive group awareness. Panel B displays an example of individual information.

We also used the enhanced version of CTAT to implement a second collaboration script phase, in which students are provided with unique information to share with their partner. As in the previous example, the collaborative tutor provides a different

view on the same problem for each collaborating partner. Specifically, we implemented a script that distributes information between the partners and supports the sharing of this information. Students are either shown an example response about the fractions and asked to share with their partner, as indicated by the "share" icon, or are asked to listen to their partner's information, as indicated by the "listen" icon (see Figure 1, panel B). After the first student shares their example response, the students then switch roles, with the second student receiving a different example to share. This activity provides each student with a different viewpoint that they can then use to start a discussion. Both example phases illustrate a range of collaborative tasks that can be supported using CTAT for collaboration by integrating the group formations (individual or dyadic tasks), the task distributions (roles and unique information), and the timing of the phase for the different tasks (ordering of the tasks) into a ITS environment that can provide feedback and hints to the student.

4 Discussion and Conclusion

CSCL has been shown to be an effective paradigm for knowledge acquisition [6], yet most authoring tools for ITSs do not support collaborative learning. We extended CTAT so it supports the authoring of collaborative tutors, allowing for scripts to be *flexibly* developed to align with the problem state and goals, while maintaining the typical ITS advantages. With this new version of CTAT, authors can develop collaborative ITSs with embedded collaboration scripts, so that features that support effective collaboration can be intertwined with those that support problem solving and the support for collaboration and problem solving can unfold dynamically with the problem state and can be shared among collaborating students. Unlike many CSCL tools, the tutor follows along with the students and can provide personalized hints and feedback on domain knowledge.

The extension to support collaborative authoring required a relatively small number of changes to CTAT, although these changes enable a wide range of collaborative tutoring interactions to be authored. First, we made it possible to use multiple tutor engines in synchronized fashion. Each tutor engine "serves" a single student in a group, but has access to the actions of the other students. This loose coupling makes it possible for the tutor engines to maintain a shared problem state yet respond differently to each student. CTAT provides the flexibility to develop a wide range of scripts. Collaborative tutors built using the CTAT extension have been used successfully in two different studies [9].

ITS and CSCL work often proceed somewhat separately. The work reported here represents a step forward in blending certain ITS tools and CSCL tools, in an effort to combine their strengths. Authoring collaborative ITSs with CTAT works well for collaboration scripts closely tied to the problem state but does not support collaboration scripts that are more independent of the problem, such as conversational agents. Cognitive group awareness and unique information were given as examples in this paper, but the design space is much larger and limits are still being determined. We look forward to continued use of our combined tool in the ITS and CSCL communities to explore the range of collaborative tutoring interactions it can support.

Acknowledgments. We thank the Cognitive Tutor Authoring Tools team for their help. This work was supported by Graduate Training Grant # R305B090023 and by Award # R305A120734 both from the US Department of Education (IES).

References

1. Adamson, D., Rosé, C.P.: Coordinating Multi-Dimensional Support in Collaborative Conversational Agents. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 346–351. Springer, Heidelberg (2012)
2. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: A New Paradigm for Intelligent Tutoring Systems: Example-tracing Tutors. International Journal of Artificial Intelligence in Education 19, 105–154 (2009)
3. Dillenbourg, P.: Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In: Three Worlds of CSCL. Can we Support CSCL? pp. 61–91 (2002)
4. Janssen, J., Bodemer, D.: Coordinated Computer-Supported Collaborative Learning: Awareness and Awareness Tools. Educational Psychologist 48(1), 40–55 (2013)
5. Lesgold, A., Katz, S., Greenberg, L., Hughes, E., Eggan, G.: Extensions of intelligent tutoring paradigms to support collaborative learning. In: Dijkstra, S., Krammer, H.P.M., van Merrienboer, J.J.G. (eds.) Instructional models in computer-based learning environments, pp. 291–311. Springer, Berlin (1992)
6. Lou, Y., Abrami, P.C., d'Apollonia, S.: Small group and individual learning with technology: A meta-analysis. Review of Educational Research 71(3), 449–521 (2001)
7. Miao, Y., Hoeksema, K., Hoppe, H.U., Harrer, A.: CSCL Scripts: Modelling Features and Potential Use. In: Proceedings of the 2005 Conference on Computer Support for Collaborative learning: learning 2005: the Next 10 years!, pp. 423–432. ISLS (2005)
8. Murray, T., Blessing, S., Ainsworth, S.: Authoring tools for advanced technology learning environments: Toward cost-effective adaptive. In: Interactive and Intelligent Educational Software. Kluwer Academic Publishers, Amsterdam (2003)
9. Olsen, J.K., Belenky, D.M., Aleven, A., Rummel, N.: Using an intelligent tutoring system to support collaborative as well as individual learning. In: Intelligent Tutoring Systems. Springer, Heidelberg (in press)
10. Rau, M.A., Aleven, V., Rummel, N., Rohrbach, S.: Sense Making Alone Doesn't Do It: Fluency Matters Too! ITS Support for Robust Learning with Multiple Representations. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 174–184. Springer, Heidelberg (2012)
11. Stegmann, K., Streng, S., Halbinger, M., Koch, J., Fischer, F., Hußmann, H.: eXtremely Simple Scripting (XSS): A framework to speed up the development of computer-supported collaboration scripts. In: Proceedings of the 9th International Conference on Computer Supported Collaborative Learning, vol. 2, pp. 195–197. ISLS (2009)
12. VanLehn, K.: The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. Educational Psychologist 46, 197–221 (2011)
13. Wecker, C., Stegmann, K., Bernstein, F., Huber, M.J., Kalus, G., Rathmayer, S., Kollar, I., Fischer, F.: Sustainable script and scaffold development for collaboration on varying web content: The S-COL technological approach. In: Proceedings of the 9th Int'l Conference on Computer Supported Collaborative learning, vol. 1, pp. 512–516. ISLS (2009)
14. Walker, E., Rummel, N., Koedinger, K.: CTRL: A research framework for providing adaptive collaborative learning support. User Modeling and User-Adapted Interaction: The Journal of Personalization Research (UMUAI) 19(5), 387–431 (2009)

A System Architecture for Affective Meta Intelligent Tutoring Systems

Javier Gonzalez-Sanchez[1], Maria Elena Chavez-Echeagaray[1], Kurt VanLehn[1], Winslow Burleson[1], Sylvie Girard[2], Yoalli Hidalgo-Pontet[1], and Lishan Zhang[1]

[1] Arizona State University, Tempe, AZ, USA
{javiergs,mchaveze,kurt.vanlehn,winslow.burleson,
yoalli.hidalgopontet,lishan.zhang}@asu.edu
[2] University of Birmingham, Birmingham, UK
s.a.girard@bham.ac.uk

Abstract. Intelligent Tutoring Systems (ITSs) constitute an alternative to expert human tutors, providing direct customized instruction and feedback to students. ITSs could positively impact education if adopted on a large scale, but doing that requires tools to enable their mass production. This circumstance is the key motivation for this work. We present a component-based approach for a system architecture for ITSs equipped with meta-tutoring and affective capabilities. We elicited the requirements that those systems might address and created a system architecture that models their structure and behavior to drive development efforts. Our experience applying the architecture in the incremental implementation of a four-year project is discussed.

Keywords: architecture, component-based, tutoring, meta-tutoring, affect.

1 Introduction

Intelligent Tutoring Systems (ITSs) seem capable of becoming untiring and economical alternatives to expert human tutors. This possibility has proven difficult to achieve, but significant progress has been made. The use of ITSs has become more common, and there is significant work about their pedagogical and instructional design but not about their technological implementation. ITSs are software products and, as for any other software product, their implementation on a massive scale relies on the principle of assembly instead of crafting them as one-of-a-kind systems. Component-based software engineering [1] is an appropriate approach for handling mass production. Component-based software engineering addresses the development of systems as an assembly of parts (components), with the development of these parts as reusable entities and with the maintenance and upgrading of systems through customizing and replacing such parts.

Following a component-based approach, we have defined a system architecture to drive the development of ITSs equipped with affective and meta-tutoring capabilities, called affective meta intelligent tutoring systems (AMTs). Defining a system architecture

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 529–534, 2014.

is the first step in creating a component-based software framework to implement AMT-like applications. This system architecture takes advantage of previous experiences with ITS implementations; most of that previous experience was extracted from the analysis made on existing ITS behavior described in [2], as well as from previous experience in the development of real-time affective companions, mainly by the work described in [3].

This paper is organized as follows: Section 2 provides the terminology and background for system architectures, ITS behavior, and affect recognition; Section 3 describes the AMT system architecture; Section 4 describes the implementation of an application following the AMT system architecture and discusses its software metrics; and Section 5 provides a conclusion.

2 Terminology and Background

The following terminology and background summary contextualizes the work described in this paper:

System Architecture. A system is a group of interacting, interrelated or independent modules forming a complex whole. Modules are self-contained entities that carry out a specific function; they are implemented as a set of parts called components. A system architecture is a conceptual model that describes the modules and components of a system and how they interconnect with each other; it becomes a software design model by mapping each component to a set of classes following software engineering methodologies. The system architecture is essential for realizing the system's quality attributes [4].

ITS Behavioral Description. ITSs are typically used to assign tasks to students; tasks are composed of steps that the student must accomplish. The structure of this kind of ITSs, called step-based, is described in [2] and can be summarized as follows: (1) the group of tasks known by the ITS conforms its *Knowledge Base*; (2) a *Task Selector* chooses from the *Knowledge Base* the Task that the student must solve by considering the student's previous performance reported by an *Assessor*; (3) a *User Interface* (a tool or an environment) provides the space in which the interaction between the tutor and the student occurs; (4) a *Step Analyzer* methodically examines the student's steps and determines whether they are correct or incorrect and then reports that information to a *Pedagogical Module* and to an *Assessor*; (5) a *Pedagogical Module* provides support (hints and feedback); the provided support depends on current steps and the student's previous performance; and (6) an *Assessor* measures the performance of the student (requested hints, time used to go from one step to another, etc.).

Affect Recognition Strategies. Research shows that learning is enhanced when affective support is present [5]. To provide that support, ITSs need to recognize students' affect. Diverse strategies exist for affect recognition; the one we are considering for this work uses sensing devices to read students' physiological responses; this strategy uses, among others, brain-computer interfaces, eye-tracking systems, face-based emotion recognition systems, and diverse sensors to measure skin conductance (arousal), posture, and finger pressure [6].

3 System Architecture

The system architecture was engineered [7] on the principles of encapsulation, low coupling, centralized shared data, and layering. Functionality is encapsulated in simple components; components that are complex and/or serve diverse purposes are split into several collaborative components. Components are low-coupled to facilitate replacement, i.e., to increase modifiability. A centralized data-sharing mechanism is used to pass data among modules to reduce latency. Components are organized in a three-layer structure in which the bottom layer encodes utility services for data management and communication responsibilities; the middle layer encodes the business logic; and the topmost layer encloses the user interface, which handles the interaction with the user. Since the user interface is particular to a specific system, it is not described here. Fig. 1 shows modules (boxes), components (gray boxes), and their relationships (arrows) as follows:

Tutor Module. It encapsulates the ITS behavior. Its components and relationships are summarized in Section 2.

Meta Tutor Module. It encapsulates the logic for providing meta-tutoring recommendations and promoting meta-skills in the student. The *Meta Tutor* module has two components: (1) an *Inspector* that reads *Tutor* events (populated in the *Shared Repository*) and filters those that suggest an intervention is needed; and (2) an *Engine* that provides intelligence to the *Meta Tutor*; the *Engine* is notified by the *Inspector* of compelling events and it infers the type of intervention that must be done. Interventions consist of showing a message or disabling channels of user interaction. The *Engine* implements the policies about how and when interventions must be done. It communicates the interventions to the *User Interface* for its execution.

Affective Companion Module. It encapsulates the logic for generating affective interventions. The *Affective Companion* has two components: (1) an *Event Selector* reads the data for *Tutor* events and affective states (populated in the *Shared Repository*) and filters combinations that suggest an intervention is needed; and (2) an *Affective Engine* that implements the affective intelligence; the *Affective Engine* is notified by the *Event Selector* of compelling combinations of *Tutor* events and affective state data and infers the type of intervention that must be done. Interventions consist of motivational messages. The *Affective Engine* implements the policies about how and when interventions must be done. It communicates the interventions to the *User Interface* for its execution.

Shared Repository. It is a centralized means for passing data among the other modules, which are running concurrently. The *Shared Repository* module follows a blackboard architectural model, in which a common data repository, "the blackboard," is updated by some modules and read by others. The *Tutor* posts events to the blackboard and the *Emotion Recognition Subsystem* posts affective state reports. The *Meta Tutor* and *Affective Companion* observe the blackboard, looking for data that triggers an action on their side.

Emotion Recognition Subsystem. It is a facade that provides a simplified interface to a source of affective state data, such as a third-party system, framework, or library.

The system architecture prioritizes the quality attributes of modifiability, extensibility, and integrability. Modifiability refers to the ease with which a component can be modified for use in applications or environments other than those for which it was specifically designed; affective and cognitive intelligence require this quality since they are implemented in different ways. Extensibility refers to being prepared for extension into unforeseen contexts since not all application requirements can be determined in advance; our system architecture required this quality to make feasible the addition of new tutoring, meta-tutoring, or affective support capabilities. Integrability is the process of combining software subsystems to assemble an overall system; AMT system architecture requires the integration of a third-party system or code (1) for affect recognition to support the functionality of the *Affective Companion* module and (2) for decision-making (machine-learning algorithm implementation) to support the functionality of the *Affective Companion* and *Meta Tutor* modules.

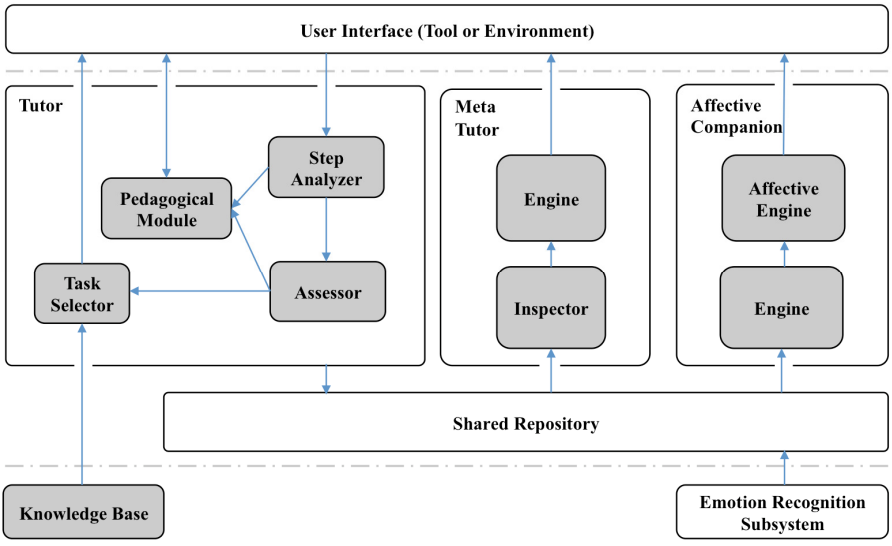


Fig. 1. AMT System Architecture

4 Usage and Discussion

The AMT system architecture has been used as a reference during a four-year project focused on developing an AMT application [8]. The AMT application was implemented in Java with Swing components. The final version is composed of 16 packages, 120 classes, 1507 methods, 1810 attributes, and 37,374 lines of code. A production-rule system and a third-party implementation of emotion recognition algorithms were used to support the application development. A detailed description of moving from AMT system architecture to software design is outside the scope of

this paper due to space limitations; nevertheless, a description of mapping the ITS module to a software design can be found in [9]. The four-year implementation process was managed using a revision control system and comprises 1,643 revisions and 8 released versions. Differences between released versions include, among others, changes in requirements, enhancements of decision-making strategies, and bug fixing. A total of 15 developers were involved in the different stages of the project, and a team of at least four developers was working concurrently in every stage.

The results of applying the system architecture were measured indirectly by evaluating the structural software quality of the systems developed under its influence using software metrics. Due to space limitations we report the evaluation of four AMT application releases, one from each development year, as follows: (1) Release 742 implemented the first deployed *Tutor*; it was focused on the *User Interface* (a tool) and had limited tutoring capabilities; coding the skeleton of the system was the primary goal during this year. (2) Release 1277 refashioned the *User Interface* and implemented an enhanced *Tutor*. (3) Release 1545 included a *Meta Tutor*, continued refashioning the *User Interface,* and enhanced the *Tutor* module. (4) Release 1643 added the *Affective Companion* capabilities, enhanced the *Meta Tutor*, and refactored the *User Interface* and *Tutor*. The metrication of structural qualities, shown in Table 1, includes measures for size, complexity, and coupling as follows: number of packages (P), number of classes (F), number of functions (Fn), number of lines of code (LoC), number of comments (LoCm), average cyclomatic complexity (AvC), maximum afferent coupling (MaxAC), and maximum efferent coupling (MaxEC) [10].

Table 1. Comparison of software metrics for modules in diverse AMT application releases

Release	Tutor								
	Date	P	F	Fn	LoC	LoCm	AvgC	MaxAC	MaxEC
742	07/2010	5	24	347	10656	2861	3.11	4	5
1277	07/2011	9	42	650	20839	4127	3.61	8	9
1545	07/2012	11	55	885	24542	4654	3.03	9	9
1643	07/2013	14	62	936	25189	4816	2.96	12	10

Release	Meta Tutor								
	Date	P	F	Fn	LoC	LoCm	AvgC	MaxAC	MaxEC
1545	07/2012	1	22	202	3346	437	2.68	4	4
1643	07/2013	1	22	248	4210	458	3.05	5	7

Release	Affective Companion								
	Date	P	F	Fn	LoC	LoCm	AvgC	MaxAC	MaxEC
1643	07/2013	3	36	323	7975	1403	2.59	9	6

Even though we had a high turnover in the development team, the size, complexity, and coupling remained at acceptable values. Size measurements (P, F, Fn, LoC, and LoCm) show a correspondence of the requirements implemented in each release and the size of the application, as well as a balance in its granularity. The average complexity (AvgC) at the module level remains within acceptable ranges (below five); at a fine-grain level (classes), not shown in the table, those values are not always acceptable. The decrease in average complexity in the latest versions of *Tutor* shows the refactoring outcome (functionality was fixed and developers focused on code improvements). Lower values in coupling measures (MaxAC and MaxEC) are

better since they are a sign of independence; the high values of coupling in *Tutor* can be justified because they belong to the *User Interface* (highly connected); *Meta Tutor* values are acceptable, but *Affective Companion* values suggest that a refactoring would be required in the implementation of this module.

5 Conclusions

In this paper, we have presented the AMT system architecture, the first step for creating a component-based software framework to implement AMT-like applications. We have defined its requirements and qualities and have shown how the AMT system architecture addresses them to support large-scale reuse. Software metrics for different releases of one AMT application show how the system architecture provided a flexible partition of the system that facilitates modifiability, extensibility, and integrability. With this proposed system architecture, we aim to share our experience, looking forward to making the development of AMT-like systems an easier, faster, and standardized process.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grant No. 0910221.

References

1. Crnkovic, I.: Component-Based Software Engineering - New Challenges in Software Development. Software Focus 2(4), 127–133 (2001)
2. VanLehn, K.: The Behavior of Tutoring Systems. International Journal of Artificial Intelligence in Education 16(3), 227–265 (2006)
3. Burleson, W.: Affective Learning Companions: Strategies for Empathetic Agents with Real-Time Multimodal Affective Sensing to Foster Meta-Cognitive Approaches to Learning, Motivation, and Perseverance. MIT PhD thesis (2006)
4. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, 2nd edn. Addison-Wesley, Boston (2003)
5. Lehman, B., D'Mello, S.K., Person, N.: All Alone with Your Emotions. In: 9th International Conference on Intelligent Tutoring Systems. Springer (2008)
6. Gonzalez-Sanchez, J., Chavez-Echeagaray, M.E., Atkinson, R., Burleson, W.: Multimodal Detection of Affective States: A Roadmap Through Diverse Technologies. In: ACM SIGCHI Conference on Human Factors in Computing Systems. ACM (2014)
7. Firesmith, D.G., Capell, P., Falkentha, D., Hammons, C.B., Latimer IV, D.T., Merendino, T.: The Method Framework for Engineering System Architectures. CRC Press (2008)
8. VanLehn, K., Burleson, W., Girard, S., Chavez- Echeagaray, M.E., Gonzalez-Sanchez, J., Hidalgo-Pontet, Y., Zhang, L.: The Affective Meta-Tutoring project: Lessons Learned. In: 15th International Conference on Intelligent Tutoring Systems. Springer (2014)
9. Gonzalez-Sanchez, J., Chavez-Echeagaray, M.E., VanLehn, K., Burleson, W.: From Behavioral Description to A Pattern-Based Model for Intelligent Tutoring Systems. In: 18th International Conference on Pattern Languages of Programs. ACM (2011)
10. Pressman, R.S.: Software Engineering, 7th edn. McGraw-Hill, Boston (2009)

Towards Automatically Building Tutor Models Using Multiple Behavior Demonstrations*

Rohit Kumar, Matthew E. Roy, R. Bruce Roberts, and John I. Makhoul

Raytheon BBN Technologies, Cambridge, MA, USA
{rkumar,mroy,broberts,makhoul}@bbn.com

Abstract. Automation of tutor modeling can contribute to scalable development and maintenance of Intelligent Tutoring Systems (ITS). In this paper, we are proposing a modification to the process used to build Example Tracing tutors which are a widely used tutor model. Our approach automatically uses behavior demonstrations by multiple non-experts (such as learners) to create a partially annotated generalized tutor model.

Keywords: Tutor Models, Example Tracing Tutors, Authoring, Automation.

1 Introduction

Example-Tracing Tutors (ETT) are a popular and effective tutor model that have been used to build ITS for a wide range of learning domains [1] since their introduction over a decade ago [2]. The popularity of this model is rooted in the reduction of effort & expertise requirements associated with building these tutors. This objective is furthered by the availability of well-developed general purpose authoring tools such as the Cognitive Tutors Authoring Tools (CTAT) [3] and the ASSISTment Builder [4]. The effectiveness of these models is based in their ability to capture learner behaviors at a fine-grained level and provide step-by-step guidance in structured learning activities.

Building ETTs involve three stages: (1) User interface development, (2) Behavior demonstration, (3) Generalization and Annotation of the behavior graph. While authoring tools listed earlier support non-programmers through each of these stages, the work in all of these stages is completely manual. Note that while this process does not require ITS developers to have advanced computing expertise, their expertise in the learning domain is exercised. Web based tools, such as the ASSISTment Builder, have enabled a community of educators with the relevant domain & pedagogical expertise to participate in this process of building ETTs.

As ITS are being deployed to a large active user pool, it is now possible to pilot the user interface with a small sample of learners to collect multiple behavior demonstrations. In this manner, the effort of behavior demonstration (Stage 2) can be distributed to a scalable *workforce*. An algorithm that can automatically create a generalized behavior graph from the multiple demonstrations collected in this way can significantly reduce the (Stage 3) effort of the ITS developer.

* This research was funded by the US Office of Naval Research (ONR) contract N00014-12-C-0535.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 535–544, 2014.

The key contribution of this paper is the proposal to adopt the modified process for building ETTs suggested above. We will characterize the problem of automatically generalizing (Stage 3) from multiple demonstrations and describe a preliminary algorithm to solve this problem. The rest of the paper is organized as follows: Section 2 situates our work in the context of our ongoing efforts for developing a general purpose learning platform for STEM domains. Section 3 will formally characterize the problem as well as describe and evaluate our preliminary approach. Section 4 discusses the implication of automating the process of building ETTs and wide range of research directions that derive from this modification to the current practice.

2 Background

2.1 The BBN Learning Platform

At BBN, we are developing a domain-independent platform for building and delivering problem-solving based learning activities over the web. The platform comprises of an extensible learning environment as well as a workbench that includes a number of tools for content development, maintenance, reporting and administration.

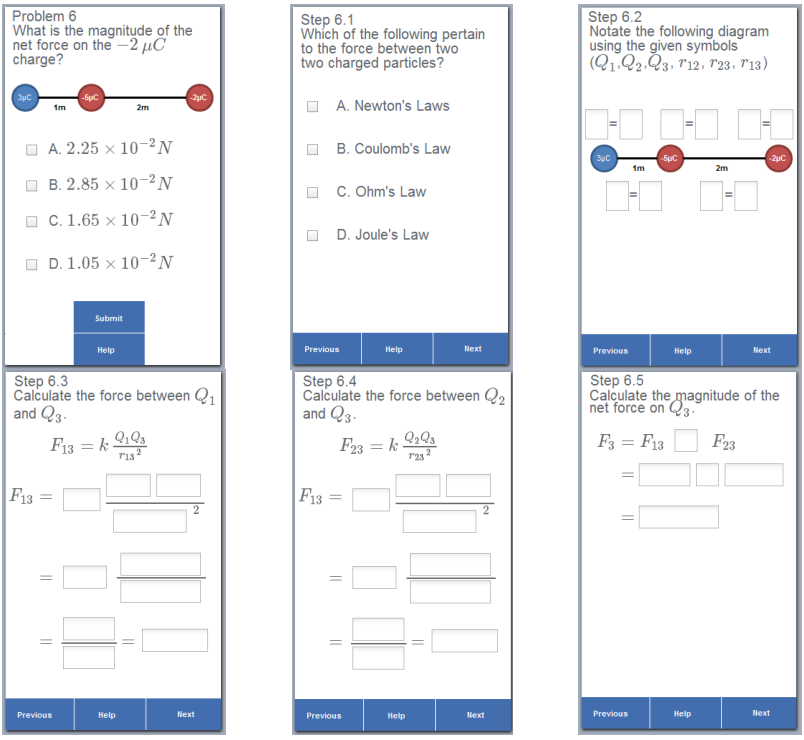


Fig. 1. A Physics problem and its Solution UI in the BBN Learning Environment

The choice of problem solving as the underlying learning activity is motivated by its applicability to a wide range of STEM domains. Specifically, we are focusing on applying the learning platform to build a high school Physics learning system covering topics in Electricity and Magnetism.

Figure 1 shows a rendering of a Physics problem in our learning environment. The UI employs a tile metaphor to allow use of the learning environment on multiple web-enabled devices including touch screen devices with relatively low screen resolution (such as smart phones). The first tile shows a problem statement. Learners can solve a given problem without tutor assistance or can click on the help button in which case the decomposition of the problem into a series of solution steps is presented. In addition to the access to this decomposition upon requesting help, we have implemented a tutoring engine that uses ETTs to provide feedback and scaffolded hints to the learners.

Our workbench includes two separate tools for supporting the three stage ETT development process. First, a WSIWYG problem authoring tool allows non-programmers to build user interfaces. The solution steps shown above showcase some of the UI elements that are available to the authors. Second, a model building tool enables them to demonstrate problem solutions and edit/annotate the behavior graph which is accessible side-by-side within the same tool.

2.2 Related Work

As a result of the successful wide use of the Knowledge Tracing and Example Tracing tutors, researchers working with these types of tutor models have been able to collect data that capture solution traces from multiple learners. In recent years, a number of researchers have published interesting work investigating the use of this trace data.

Sudol et al. [5] aggregated solution paths taken by different learners to develop a probabilistic solution assessment metric. Johnson et al. [6] are creating visualization tools for interaction networks that combine learner traces from open-ended problem solving environments. They have developed an algorithm for reducing the complexity of combined networks to make them more navigable. In a similar spirit, work by Ritter et al. [7] used clustering techniques to reduce the large feature space of student models to assist in qualitative model interpretation.

Note that some of this existing work has used traces generated by learners after an ITS has been developed. The work presented in this paper focuses on use of solution demonstrations by learners to assist in building the ITS. McLaren et al. [8] also proposed the use of activity logs from novice users to bootstrap tutor model development.

3 Learning Tutor Models

3.1 Behavior Demonstrations and Behavior Graphs

Before we present our algorithm for automatically generalizing behavior demonstration, this section will describe the representation we use for capturing behavior demonstrations and visualize behavior graphs.

Behavior demonstrations are captured as a sequence of user interface (UI) events. Each event is represented as a 2-tuple <element_id, data> that includes an identifier of the UI element and data associated with the event. Figure 2a shows the UI of a step in a middle-school level fractions problem. It comprises of 5 active elements which are annotated with their identifiers. A hypothetical demonstration path through those elements is also annotated. Figure 2b shows the events associated with that demonstration path. Note that the data field need not be limited to a single attribute. For example, in certain applications, event qualifiers such as type, duration, etc. may be included in the data. Beside the sequential nature of events in a behavior demonstration, events may have the same element identifier. Such events indicate a modification (e.g. a correction). We will refer to events that are succeeded by other events with the same identifier as *retracted* events. Note than unlike traces (discussed in Section 2.2) which are solution paths through an existing behavior graph, behavior demonstrations are not constrained to valid paths in the behavior graph. Because of this, behavior demonstrations by different users can be very different from each other in terms of the length of the sequence and the events they contain.

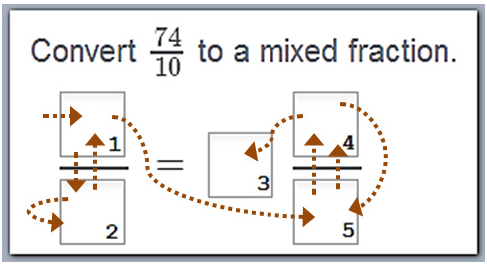


Fig. 2a. UI of a sample problem step

```
  ...
< 1, 10 >
< 2, 74 >
< 2, 10 >
< 1, 74 >
< 5, 10 >
< 4, 4  >
< 5, 5  >
< 4, 2  >
< 3, 7  >
  ...
```

Fig. 2b. Behavior demonstration data

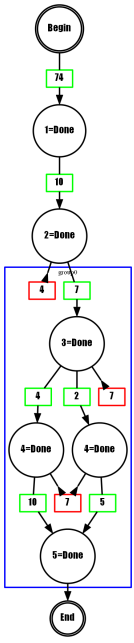


Fig. 2c. Subsection of a Behavior Graph

Behavior graphs are directed graphs. A manually constructed behavior graph corresponding to the above UI is shown Figure 2c. The nodes in this graph (shown as circles) correspond to valid partial solution states. Edges in the graph represent behavior events some of which are correct and lead to the next state while other are incorrect and lead back to the same state. Correct edges are labeled using green boxes and incorrect edges are labeled using red boxes. Edges are also annotated with the data

field of events and may be annotated with additional information such as element identifiers if necessary for readability. Besides a sequential organization of solution state, the behavior graph above showcases an alternate path between the states labeled 3=Done and 5=Done. Alternate paths are an important feature of behavior graphs especially for the use of ETTs in ill-defined learning domains. They support learner's exploration of alternate solutions to a problem.

In addition to nodes and edges, behavior graphs include unordered groups which indicate that states within a group may be traversed in any order. The states bound by the blue box are an example of an unordered group is shown in Figure 2c. This unordered group indicates that the three UI elements corresponding to the mixed fraction may be filled in any order.

Behavior graph authoring tools support a number of additional annotations on both the nodes and the edges. For example, incorrect edges may be annotated with corrective feedback provided to learners when their trace traverses that edge. Nodes are usually annotated with hints for tutoring applications and with skills for student modeling and assessment applications.

3.2 Desirable Characteristics of Behavior Graphs

Readable
One of the key characteristics of Behavior Graphs that makes them a popular model is that they are readable by ITS authors without requiring a deep understanding of computational or cognitive sciences. Automatically created behavior graphs should be editable with existing authoring tools to facilitate manual annotation and modifications. Ideal generation algorithms should create concise graphs without losing other desirable characteristics. This may involve collapsing redundant paths and even pruning spurious or infrequent edges.

Complete
In order to minimize author effort, generated behaviors graphs should be as complete for creating an ETT as possible. As a minimal criterion, at least one valid path to the final solution should be included. Note that the creation of a complete graph (even manually) relies on the availability of one or more *complete* behavior demonstrations.

Accurate
Behavior graphs should be error free. This includes being able to accurately capture the correct and incorrect events by learners depending on their current state.

Robust
One of the reasons for the success of good ETTs is the ability to use them with a wide range of learners under different deployment conditions. Automatically generated behavior graphs should retain this characteristic, e.g., by identifying alternate paths and unordered groups. A robust behavior graph need not necessarily be the most unconstrained graph, which maybe prone to gaming behaviors by learners. It is not unforeseeable that the use of a data-driven approach could contribute to creating behavior graphs that are more robust than those authored to a human expert.

3.3 Algorithm for Automatically Creating Behavior Graphs

Now we will describe a preliminary four-stage algorithm that combines multiple behavior demonstrations to automatically create a behavior graph. Several simplifying assumptions are made about the demonstrations which are explicitly noted to encourage the development of more robust algorithms.

Stage 1. Reduce Retracted Events
We assume that all retracted events in a demonstration correspond to mistakes which were corrected by the user when the prior event is retracted. We process each available demonstration independently to combine the data from all retracted events into the last occurring event with the same element in each demonstration. The combined data values from the retracted events are considered as incorrect inputs for that element. This stage of the algorithm is similar to pre-reduction step used by Johnson et al. [6].

Stage 2. Calculate Sequence of States
We assume that there is one and only one path through the UI elements of the solution interface. This stage calculates the most frequently taken path through those elements to create a sequence of states for the automatically generated behavior graph. In the current implementation, we also assume that all demonstration end in a correct solution. For each unique UI element, collect events from all available demonstrations that were generated by the element under consideration. After stage 1, there should be at most one such event in each demonstration. As these events are collect, the positional index an event is found in each demonstration is preserved. Elements are sorted in an increasing order of the mode of their positional indices to obtain the sequence of states. Mean is used as a tie-breaker if elements have the same positional mode.

Stage 3. Generate Edges
Given the sequence of states, we can generate a behavior graph by constructing edges between the states. For each unique correct data value an element takes in the demonstrations, we generate a correct edge between to the state corresponding to the element from the previous state. Similarly, for each incorrect data value (identified in Stage 1), an incorrect edge is generated at the previous state. The frequency of a data value is used to highlight each edge. This information can be used to prune a behavior graph for readability. Due to the small amount of data used in the experiment presented in this paper, no pruning is applied to the graphs generated.

Stage 4. Identify Unordered Groups
Two adjacent states are added to an unordered group if the corresponding UI elements frequently share each other's positional indices in the multiple demonstrations. Currently, we use a heuristic function ($\sqrt{\#demonstrations}$) to determine the threshold frequency. Unordered groups between adjacent pairs of states are merged.

3.4 Pilot Data Collection

We conducted an experiment to collect behavior demonstrations for five Physics problems on the topic of Electrostatics. We recruited nine subjects to participate in the experiment.

All subjects were adult BBN employees who had completed a high-school Physics course that covered topics in Electricity and Magnetism during their education. None of the subjects are educated in advanced Physics or are practicing Physicists in their professions. No refresher of the subject matter was provided prior to the experiment to elicit common mistakes from the subjects. They were allowed to use a scientific calculator and were provided data (Coulomb's constant, Charge of an electron) required to solve the problem.

Each subject spent one hour on the experiment. Time spent on the experiment counted towards their regular work hours. During the one hour, a sequence of five problems was presented, one at a time. Each problem included a problem statement and a number of steps. Figure 1 shows one of the problems used in our data collection. The data collection was completed over two days.

Table 1. Pilot Data Collection Statistics

	#	#Demonstration Events					#UI
	Demonstration	Min.	Max.	Total	Avg.	St.Dev.	Elements
problem1	9	5	8	49	5.44	0.96	5
problem2	9	18	28	195	21.67	3.43	18
problem6	9	35	52	377	41.89	6.08	37
problem10	6	37	41	230	38.33	1.86	37
problem15	4	54	58	223	55.75	1.48	55

All nine subjects were able to complete the first three problems (problem1, problem2, problem6) within an hour. Six subjects completed the fourth problem (problem10) and only four completed the fifth problem (problem15). Table 1 shows some statistics about the behavior demonstrations used in our experiment.

3.5 Analysis and Results

The algorithm described in Section 3.3 was applied to the set of behavior demonstrations available for each problem to automatically create a behavior graph for each problem. Figure 3 shows the automatically created graphs for problem1, problem2 and problem15. Because of the large numbers of UI elements in problem2 and problem15, only part of their behavior graphs are shown. The automatically generated behavior graphs use the same representation as manually authored behavior graphs such as the one shown in Figure 2c to allow further annotation of these graphs within our model building tools (mentioned in Section 2.1).

Ideally, tutor models should be evaluated in terms of learning efficacy by deploying them in a relevant sample learner population. However, since our learning platform is currently under development, we will use a number of other metrics, shown in Table 2, to evaluate the automatically generated graphs with respect to some of the desirable characteristic listed in Section 3.2. Descriptive statistics about the generated graphs (Number of nodes, edges, groups) are included.

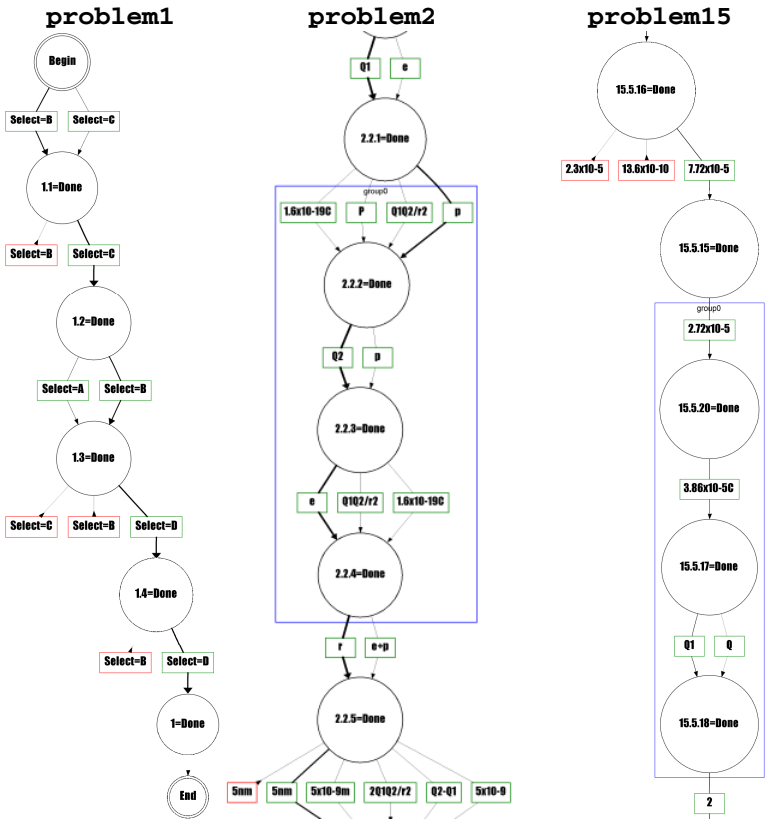


Fig. 3. Examples of Automatically Generated Behavior Graphs

Table 2. Statistics about Automatically Generated Behavior Graphs

	# Nodes	Compr. Ratio	Correct Edges #	Correct Edges Accuracy	Incorrect Edges #	Incorrect Edges Accuracy	Error Rate	Branch. Factor	# Groups
problem1	7	7.0	7	71.4	4	100.0	8.2	2.2	0
problem2	20	9.8	59	74.6	30	100.0	8.7	4.9	3
problem6	39	9.7	102	74.5	40	97.5	7.2	3.8	2
problem10	39	5.9	89	68.5	8	100.0	12.2	2.6	1
problem15	57	3.9	93	95.7	4	100.0	1.8	1.8	2
Average		7.3		76.7		99.5	7.6	3.1	

Readable. Compression Ratio measures the rate at which demonstration events are reduced into behavior states (i.e. nodes). A trivial algorithm that generates a full interaction network from the available demonstrations will have a compression ratio of 1.0. Our algorithm achieves an average compression ratio of 6.63. Problems with more demonstrations are able to achieve higher compression because our algorithm combines identical events during Stage 3 & 4.

Complete. The minimal criterion for completeness is guaranteed by the assumptions made at Stage 2 of our algorithm. Once we operationalize our authoring tools, we would like to measure additional authoring effort required annotate and modify automatically generated graphs as a measure of completeness.

Accurate. Edge accuracy measures the percentage of Correct & Incorrect edges that were accurately classified by the algorithm. Error rate is a frequency weighted combination of edge accuracy that measures the fraction of learner events that will be inaccurately classified by the automatically generated behavior graph. We believe this should be the primary metric for evaluating automatic behavior graph generation. As we see from Table 2, both our accuracy metrics have scope for significant improvement. Note that the trivial algorithm that generates an interaction network would achieve an error rate of 0% on the demonstrations used to build the network. For experimental validity, it is better to use held out demonstrations to measure accuracy metrics. Due a small amount of data used in our experiment, this is an evaluation shortcoming.

Robust. Branching factor is the average number of data values available at each UI element. A large branching factor indicates the capability to process a large variety of learner inputs at each state. Average number of retracts, a related metric, measures the average number of retracted events identified during Stage 2 of our algorithm. Held-out demonstrations can also be used to measure the robustness towards unseen user inputs. Finally, a larger number of unordered groups is indicative of flexibility a graph affords to learners to explore the solution paths of a problem.

4 Discussion

In this paper, we have proposed a modification to the current process used to develop ETTs that employs multiple behavior demonstrations to automatically generate a partially annotated behavior graph. The impact of this modification is not only the potential for scalable ITS development by reduction in authoring effort, but also the increased validity of tutor models generated from behavior demonstrations that are collected from the intended end-user. We have also presented a preliminary algorithm for automated behavior graph generation as well as a number of analytical metrics that can be used to evaluate the performance of the algorithm in terms of a set of desirable characteristics. Results presented here establish a baseline to compare improved algorithms.

There are some shortcomings of our current approach. First, our algorithm merges alternate solution paths into a single sequence of states due to an assumption made at Stage 2. Ability to extract alternate paths will be useful for application of our process to ill-defined learning domains. Second, our algorithm does not discover navigational constraints in problem solving interfaces that are used to prevent the learners from loafing or gaming the system. By including navigational events during the process of collecting behavior demonstration, it is possible to automatically include these constraints in the behavior graphs. Similarly, our algorithm can be extended to automatically discover optional elements in the solution interface. Finally, it would be useful to modify our algorithm to improve existing manually authored behavior graphs to facilitate automated ITS maintenance.

Note that the problem of combining multiple sequences of data (e.g. protein sequences) into graph like structures has been explored by researchers working on other types of intelligent systems. Also, work in ontology alignment, social network analysis and graph induction has developed techniques that could motivate innovative approaches to automatically generating behavior graphs. In addition to improving behavior graph generation algorithms, a key directions leading from this work is the need to integrate these algorithm with ITS authoring tools.

In conclusion, we emphasize the need to employ data-driven approaches for ITS development. The process modification discussed in this paper is a step towards distributed ITS development that can employ communities of learners and education for scalability. An interesting sub-problem of selection of appropriate users for eliciting behavior demonstrations needs further attention. Finally, we want to note that in real world ITS deployments, it is possible to collect couple of order of magnitudes more behavior demonstrations for generating and maintaining behavior graphs. Access to such data is likely to lead to drastically different approaches to ITS development.

References

1. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: A new paradigm for intelligent tutoring systems: Example-tracing tutors. International Journal of Artificial Intelligence in Education 19(2), 105–154 (2009)
2. Koedinger, K.R., Aleven, V., Heffernan, N.T., McLaren, B.M., Hockenberry, M.: Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 162–174. Springer, Heidelberg (2004)
3. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: The Cognitive Tutor Authoring Tools (CTAT): Preliminary evaluation of efficiency gains. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 61–70. Springer, Heidelberg (2006)
4. Razzaq, L., Patvarczki, J., Almeida, S.F., Vartak, M., Feng, M., Heffernan, N.T., Koedinger, K.R.: The ASSISTment Builder: Supporting the Life Cycle of Tutoring System Content Creation. IEEE Transactions on Learning Technologies 2(2), 157–166 (2009)
5. Sudol, L.A., Rivers, K., Harris, T.K.: Calculating Probabilistic Distance to Solution in a Complex Problem Solving Domain. In: Yacef, K., Zaïane, O., Hershkovitz, H., Yudelson, M., Stamper, J. (eds.) Proceedings of the Fifth International Conference on Educational Data Mining, pp. 144–147 (2012)
6. Johnson, M., Eagle, M., Stamper, J., Barnes, T.: An Algorithm for Reducing the Complexity of Interaction Networks. In: D'Mello, S.K., Calvo, R.A., Olney, A. (eds.) Proceedings of the Sixth International Conference on Educational Data Mining, pp. 248–251 (2013)
7. Ritter, R., Harris, T.K., Nixon, T., Dickison, D., Murray, R.C., Towle, B.: Reducing the Knowledge Tracing Space. In: Barnes, T., Desmarais, M., Romero, C., Ventura, S. (eds.) Proceedings of the Second International Conference on Educational Data Mining, pp. 151–160 (2009)
8. McLaren, B.M., Koedinger, K.R., Schneider, M., Harrer, A., Bollen, L.: Bootstrapping Novice Data: Semi-Automated Tutor Authoring Using Student Log Files. In: Proceedings of the Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes, 7th International Conference on Intelligent Tutoring Systems (2004)

Testing Language Independence in the Semiautomatic Construction of Educational Ontologies

Angel Conde, Mikel Larrañaga, Ana Arruarte, and Jon A. Elorriaga

University of the Basque Country (UPV/EHU) 20080, Donostia, Basque Country
{angel.conde,mikel.larranaga,a.arruarte,jon.elorriaga}@ehu.es

Abstract. In this paper, the language independence of DOM-Sortze for creating Educational Ontologies from electronic textbooks is tested. DOM-Sortze has been designed to be language and domain independent. Initially, it was tested with documents written in the Basque language. In this work, DOM-Sortze has been enhanced to deal with the English language. In addition, the benefit of incorporating Wikipedia as a knowledge source in the elicitation process of the Educational Ontology is also considered. The obtained results confirm the language independence of this approach.

Keywords: Domain Module, Educational ontology, Ontology learning, Language independence.

1 Introduction

To be effective, any Technology Supported Learning System (TSLS) requires an appropriate representation of the knowledge to be mastered by the student, i.e., the Domain Module. The Domain Module is considered the core of any TSLS [1]. In the literature, the Domain Module has been represented in several means, including the ontological approach [1]. In the approach presented throughout this paper, the Domain Module is described by means of an Educational Ontology, the Learning Domain Ontology (LDO), and Learning Objects (LOs) [2]. Educational Ontologies encapsulate the domain knowledge of a TSLS along with the related pedagogical knowledge [3]. The LDO contains the main domain topics and the pedagogical relationships among them. Pedagogical relationships can be structural ─*isA* and *partOf*─ or sequential ─*prerequisite* and *next*─ [2].

Ontology learning, i.e., gathering domain ontologies from different resources in an automatic or semiautomatic way has been addressed in many projects [4, 5]. Most of these projects aim at building or extending a domain ontology or populating lexical ontologies such as Wordnet [6] or EuroWordnet [7]. Ontology learning usually combines machine learning and Natural Language Processing (NLP) techniques to build domain ontologies or to enhance and populate some base ontologies. Ontology learning relies on the assumption that there is semantic knowledge underlying syntactic structures. For instance, Text2Onto [8] uses Hearst's patterns [9] to gather taxonomic relationships, and nested terms-based methods to identify the set of candidate domain topics.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 545–550, 2014.

OntoLT [10], a Protégé (http://protege.stanford.edu/) plug-in for the extraction of ontologies from texts, identifies taxonomic relationships on nested terms, relying on the appearance of a term and some modifiers (*Genus et Differentiam*).

DOM-Sortze [2] is a suite of applications and web-services that address every task for the semiautomatic development of the Domain Module from electronic textbooks: the acquisition of the LDO, generation of LOs for the topics to be mastered, and the supervision of the construction process. This web-service oriented approach makes DOM-Sortze flexible and platform-independent. DOM-Sortze was designed to be able to deal with different languages and domains. This suit of applications does not strongly depend on a concrete language, even though it has been initially applied on textbooks written in the Basque language.

DOM-Sortze combines NLP techniques with heuristic reasoning and ontologies to construct the Domain Module. It uses a set of heuristics and patterns based on syntactic information that allow the identification of meaningful pieces of knowledge from which the LDO is built. Furthermore, it has been observed that similar or equivalent patterns exist for other languages such as English [11]. Thus, DOM-Sortze can easily be enhanced to deal with a new language. It needs the heuristics and the patterns for identifying the topics and relationships, and to enhance the NLP Analysis Service with a NLP analyser for the new language.

In this paper, the automatic identification of structural relationships from document outlines written in English is addressed to confirm the language independence of this approach. Two versions of the process are tested: first the heuristic-based process (Section 2) and then the Wikipedia-enhanced process (Section 3). To end up, some final remarks and future work are provided.

2 Heuristic-Based Elicitation from Document Outlines

Document outlines are useful sources of information for acquiring the Domain Module in a semi-automatic way; they are usually well-structured and contain the main topics of the domain. In addition, they are considerably summarised, so a lot of useful information can be extracted with a low cost process. The authors of textbooks have previously analysed the domain and decided how to organise the content according to pedagogical principles in order to promote the learning and understanding of their content. The organisation of the textbook is reflected in the outline. Thus, most of the implicit pedagogical relations can be inferred from the outline. The outline analysis process consists of two phases: basic analysis and heuristic analysis.

In the **basic analysis**, the main topics of the domain and the relationships between these topics are mined from the outline. In this approach, each index item is considered as a domain topic. Besides, the structure of the document outline is used as a means to gather pedagogical relationships. A subitem of a general topic is used to explain part of it or a particular case of it. Therefore, structural relationships are defined between every outline item and all its subitems.

In the **heuristic analysis** the results of the basic analysis are refined based on a set of heuristics that categorise the relationships identified in the previous step and mine

new ones, mainly prerequisite relationships. The heuristics entail the condition to be matched, and the post-condition, i.e., the relationships that are recognised. The heuristic analysis relies on the empirically gathered confidence of the heuristic, i.e. the percentage of times the heuristic fires correctly.

The identification of Structural Relationships is carried out to categorize the relationship between an item and its subitems. In previous experiments, it was noticed the isA relationships could be inferred in different cases (see Table 1). On the one hand, homogeneous subitems allow the identification of such relationship. Both subtitems share a common head (**clustering**) which is enhanced with some modifier following a *Genus et differentiam* pattern. A set of heuristics (*group heuristics*) allow the identification of *isA* relationships from homogeneous structures. On the other hand, other fragments containing *isA* relationships are more heterogeneous. In the example, three kinds or security methods are presented. The first one is an acronym whereas the second one is a proper name. *Individual heuristics* are aimed at the identification of structural relationships in these situations.

Table 1. Examples of outline fragments from which *isA* relationships can be inferred

Homogeneous subitems	**Heterogeneous subitems**
5. Numerical classification	6. Transport and network-level security methods
5.1 Exclusive **clustering**	**6.1 SSL**
5.2 Hierarchical **clustering**	**6.2 IPSec**
	6.3 Virtual private networks

The structural relationships are identified in the heuristic-driven process described next. For each outline item, a group heuristic that matches is looked for. *Group heuristics* identify *isA* relationships from homogeneous subitems or if the outline item entails certain keywords. If such a heuristic fires, then *isA* relationship is defined between the outline item and each of its subitems. Otherwise, the *individual heuristic* that triggers is search for on every subtitem. Different heuristics can be fired together in the same group of subitems so, the most confident one is returned; the default heuristic (partOf) is returned when no other heuristic condition is met [12]. Then, the list of applied heuristics is processed to get the confidence on an underlying *isA* relationship using (Equation 1),

$$conf_{isA} = \frac{\sum_{h \in H_i} f(h) * c(h) - \sum_{h \in H_p} f(h) * c(h)}{n} \quad (1)$$

where h represents a heuristic, *f(h)* is the number of times the heuristic *h* is triggered, *c(h)* is the confidence on heuristic *h*, H_i the set of heuristics that identify *isA* relationships and H_p the set of heuristics that reinforce the hypothesis that the relationship is a *partOf* relationship, and n represents the number of subitems. If the $conf_{isA}$ value goes beyond a threshold, then the structural relationships are refined as *isA*, otherwise, the relationships are labeled as *partOf*.

To support the acquisition of structural relationships from document outlines written in English, equivalent heuristics to those described in [12] have been defined. Those heuristics rely on syntactic patterns and do not use any domain-specific knowledge. Some of those heuristics rely on NLP services, for instance, those to identify entity names. Therefore, the NLP services have to provide the same functionality for English, to which end they were enhanced to use the Illinois Named Entity Tagger [13] for NLP tasks. This tool has been mainly used for entity recognition.

2.1 Experiment

To validate the proposal, 57 outlines of different courses have been processed. The evaluation of the proposal described throughout this paper was conducted following a *gold-standard* approach. The authors of the paper and lecturers of the courses defined the LDOs that were used as optimal output. These LDOs were restricted to the topics referred on the outlines and the structural relationships between those topics (1197 *partOf*, 483 *isA*). Then, every outline was processed and the automatically gathered ontologies were compared to the gold-standard. The process was evaluated in terms of *recall*, i.e., the percentage of identified relationships, and *precision*, i.e., the percentage of correctly classified relationships.

HP (Heuristic Process) columns on Table 2 show the results of this experiment. The overall *precision* and *recall* measures are positive (83.85%). Furthermore, the scores achieved for the *partOf* relationships were even higher; 84.12% *precision* and 98.66% *recall*. However, the recall for *isA* relationships dramatically dropped to 21.20%, although the *precision* was still good (78.95%). A deep analysis of the results was conducted to determine why the results were worse than expected. The lack of knowledge on certain domains significantly affected the performance. For instance, it was observed that many of the topics involved in the missing *isA* relationships contained proper names; however, the entity name recognizer used in the experiment was unable to identify them. A training process would be necessary to fulfill such purpose. Given that the process aims to be domain-independent, this was not an option.

To improve the results, a new step was included in the elicitation process using Wikipedia as an additional resource. This improvement is described in next section.

Table 2. Results of the Heuristic Process (HP) and the Wikipedia-Enhanced Process (WEP)

	partOf		***isA***		***Total***	
	HP	WEP	HP	WEP	HP	WEP
Precision (%)	84.12	89.19	78.95	77.30	83.85	87.70
Recall (%)	98.66	96.49	21.20	50.53	83.85	87.70

3 Enhancing the Elicitation Process with Wikipedia

Wikipedia is a collaborative online encyclopedia containing over 30 million articles in 287 languages (as of January 2014). It has a vast, constantly evolving tapestry of richly interlinked articles, i.e., concepts and semantic relationships [14]. Wikipedia is an

appropriate resource for NLP given that it is: domain independent (it has a large coverage), up-to-date, and multilingual [15]. Ponzetto and Strube [15] derived a large scale taxonomy containing *isA* relationships from Wikipedia. In the proposal presented throughout this paper, this taxonomy has been used to discover missing *isA* relationships. In most cases, these kinds of relationship appear in lower-levels (involving leave nodes) of the LDO. To improve the results an additional process is carried out:

1) Identify groups of sibling nodes (topics) of the LDO extracted from the outline; 2) select the groups of leave nodes in which the *partOf* relationship has been identified to apply the subsequent steps; 3) normalize the nodes (removing plural marks, apostrophes and avoiding case differences); 4) link every node to those Wikipedia articles which are labeled with the normalized text of the node; 5) run a disambiguation process based on Wikiminer [14] to map each node to a unique article; 6) process every group using Ponzetto and Strube's taxonomy [15] to look for common ancestor; 7) infer *isA* relationships in those groups that share a common ancestor, as long as it does not appear at top-levels in the taxonomy.

3.1 Experiment

The results of the Wikipedia-Enhanced Process (WEP) have also been tested using the gold-standard (see WEP columns on Table 2). The overall performance has improved (87.70% *precision* and *recall*). Regarding *partOf* relationships, the *recall* has slightly decreased (96.49% vs. 98.66%) but the *precision* has slightly increased from 84.12% to 89.12%. In regards to *isA* relationships, the *recall* has dramatically increased from 21.20% to 50.53% whereas the *precision* was hardly affected (77.30% vs. 78.95%).

4 Conclusions

In this paper, the language independence of DOM-Sortze for creating Educational Ontologies has been tested. DOM-Sortze is a suite of applications and web-services aiming at the semiautomatic development of Domain Modules from electronic textbooks. In order to build the Learning Domain Ontology, DOM-Sortze relies on a heuristic-driven document outline analysis. This suite was designed to be language and domain independent. Initially, it was tested with documents of different areas written in the Basque language. In this work, DOM-Sortze has been enhanced to deal with the English language. The heuristics used for the outline analysis have been adapted from those identified for the Basque language.

An experiment using 57 outlines of courses that cover different areas has been conducted. Furthermore, an additional step in which Wikipedia is used to refine the relationships has been included. This new step dramatically improved the *recall* for the *isA* relationships (29.33% enhancement), while the precision was barely affected. In addition, the overall performance also increased, as the *precision* for the *partOf* relationship slightly improved, minimally decreasing its *recall*.

The obtained results confirm the language independence of this approach. In addition, the use of the Wikipedia places the presented proposal on the fast track towards the multilingual Educational Ontology learning.

Acknowledgements. This work is supported by the Basque Government (GIC12/79) and the University of Basque Country (UPV/EHU) (UFI11/45).

References

1. Nkambou, R.: Modeling the Domain: An Introduction to the Expert Module. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) Advances in Intelligent Tutoring Systems. SCI, vol. 308, pp. 15–32. Springer, Heidelberg (2010)
2. Larrañaga, M., Conde, A., Calvo, I., Elorriaga, J.A., Arruarte, A.: Automatic Generation of the Domain Module from Electronic Textbooks: Method & Validation. IEEE Transactions on Knowledge and Data Engineering 26, 69–82 (2014)
3. Fok, A.W.P., Ip, H.H.S.: Educational Ontologies Construction for Personalized Learning on the Web. In: Jain, L.C., Tadman, R.A., Tadman, D.K. (eds.) Evolution of Teaching and Learning Paradigms in Intelligent Environment. SCI, vol. 62, pp. 47–82. Springer, Heidelberg (2007)
4. Pazienza, M.T., Stellato, A. (eds.): Semi-automatic ontology development: Processes and resources. Information Science Reference, Hershey (2012)
5. Buitelaar, P., Cimiano, P., Magnini, B. (eds.): Ontology Learning from Text: Methods, Applications and Evaluation. IOS Press (2005)
6. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1998)
7. Vossen, P.: Extending, Trimming and Fusing WordNet for Technical Documents. In: NAACL-2001 workshop on WordNet and Other Lexical Resources: Applications (2001)
8. Cimiano, P., Hotho, A., Staab, S.: Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. Journal of Artificial Intelligence Research 24, 305–339 (2005)
9. Hearst, M.A.: Automatic Acquisition of Hyponyms from Large Text Corpora. In: Procs. of the 14th International Conference on Computational Linguistics, pp. 539–545 (1992)
10. Buitelaar, P., Olejnik, D., Sintek, M.: A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 31–44. Springer, Heidelberg (2004)
11. Verbert, K.: An Architecture and Framework for Flexible Reuse of Learning Object Components. Phd Thesis. Faculteit Ingenieurswetenschappen, Katholieke Universiteit Leuven (2008)
12. Larrañaga, M., Rueda, U., Elorriaga, J.A., Arruarte, A.: Acquisition of the Domain Structure from Document Indexes Using Heuristic Reasoning. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 175–186. Springer, Heidelberg (2004)
13. Ratinov, L., Roth, D.: Design Challenges and Misconceptions in Named Entity Recognition. In: Proc. of the Thirteenth Conference on Computational Natural Language Learning. CoNLL 2009, pp. 147–155. Association for Computational Linguistics (2009)
14. Milne, D., Witten, I.H.: An open-source toolkit for mining Wikipedia. Artificial Intelligence 194, 222–239 (2013)
15. Ponzetto, S.P., Strube, M.: Deriving a large scale taxonomy from Wikipedia. In: Proceedings of the 22nd National Conference on Artificial intelligence, AAAI 2007, vol. 2, pp. 1440–1445. AAAI Press (2007)

Authoring Tutors with SimStudent: An Evaluation of Efficiency and Model Quality

Christopher J. MacLellan, Kenneth R. Koedinger, and Noboru Matsuda

Human-Computer Interaction Institute
Carnegie Mellon University, Pittsburgh, PA 15213, USA
{cmaclell,noboru.matsuda}@cs.cmu.edu, koedinger@cmu.edu

Abstract. Authoring Intelligent Tutoring Systems is expensive and time consuming. To reduce costs, the Cognitive Tutor Authoring Tools and the Example-Tracing Tutor paradigm were developed to make the tutor authoring process more efficient. Under this paradigm, tutors are constructed by demonstrating behavior directly in a tutor interface, reducing the need for programming expertise. This paper evaluates the efficiency of authoring a tutor with SimStudent, an extension to the Example-Tracing paradigm that is designed to produce greater generality in less time by induction from past demonstrations and feedback. We found that authoring an algebra tutor in SimStudent is faster than Example-Tracing while maintaining equivalent final model quality. Furthermore, we found that the SimStudent model generalizes beyond the problems that were used to author it.

1 Introduction

Intelligent Tutoring Systems (ITSs) are a widely used educational technology [1] that has been shown to improve learning over many traditional forms of instruction [2–7]. One challenge associated with ITSs is that they are difficult to build and require developers to make decisions about trade offs between power, usability, fidelity, and cost [8]. To overcome the challenge of authoring high-quality tutors, many authoring tools have been developed [8]. We focus on the Cognitive Tutor Authoring Tools (CTAT), which has been shown to decreases the time required to build a tutor by as much as 50% [9]. CTAT achieves these gains by providing a drag-and-drop interface builder and by providing support for authoring two types of tutors: Cognitive Tutors and Example-Tracing Tutors.

Cognitive Tutors provide step-by-step feedback to students while they solve problems by comparing their actions to a model of expert behavior for the given domain. This model uses production rules, if-then rules that map each state in a tutoring interface to a legal action that might be taken on that state [10]. These production rules are quite general, in that a single rule might apply to many states throughout problem solving. However, in general these models are costly to produce. It can take 200-300 hours of development to produce a Cognitive Tutor for one hour of instruction and tutor development usually requires multiple

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 551–560, 2014.

kinds of expertise (i.e., domain expertise, Cognitive Psychology, and Computer Science) [8, 11].

Example-Tracing Tutors were developed to reduce the costs of producing a Cognitive Tutor [9, 11]. These tutors reduce the technical costs of tutor development by allowing domain experts and Cognitive Psychologists to build a cognitive model by demonstration rather than by programming a production rule model. To build an expert model in this paradigm, the tutor author demonstrates every legal action at every step for every problem. The resulting cognitive model, called a behavior graph, is a simplified production rule model, where each production rule maps a single state to a single action. While some methods for generalization do exist, these models are still much less general than Cognitive Tutors. However, in practice this limitation is balanced out by the ease of authoring– in many cases individuals can learn to author tutors in one afternoon [9].

While CTAT drastically reduces the cost of authoring ITSs, the tutors that it can produce are at two ends of an authoring spectrum: Cognitive Tutors are difficult to produce, but are maximally general, while Example-Tracing Tutors are easy to produce, but are maximally specific. Recent extensions to Example-Tracing Tutors have addressed how to make Example-Tracing Tutors more general. Existing techniques include specifying sequences of actions that might be executed in any order, employing regular expressions or formulas for matching demonstrations, and duplicating behavior graph structures for many problems of similar type, an approach called mass-production [11]. While these techniques have improved Example-Tracing Tutor generality, more research into how general expert models might be produced without technical expertise is still an active area of research.

One promising development is the SimStudent architecture, a CTAT module that tries to bridge the gap between Example-Tracing Tutors and Cognitive Tutors by learning production rule models from demonstrations and problem-solving feedback [12]. Previous work has shown that authoring a model by tutoring (both demonstrations and feedback) is more efficient than demonstrations alone. However, the SimStudent approach to authoring has never been compared to the more widely used Example-Tracing approach.

We compare authoring time for a tutor built with SimStudent and Example-Tracing by using a Keystroke-Level Model (KLM) [13], a simple human information processing model that estimates how many seconds it would take a trained user to perform authoring actions. This analysis shows that SimStudent can reduce authoring time by as much as 50%, for domains that SimStudent has adequate background knowledge. Additionally, we evaluate the quality of the model produced by each approach and show that while both approaches produce models with equivalent quality by the end of authoring, SimStudent shows the ability to generalize from authored to unauthored problems along the way. Before showing these results, we review CTAT and how it can be used to author an Example-Tracing Tutor and then show how the SimStudent architecture can be used to author a tutor through CTAT.

2 Authoring an Example-Tracing Tutor in CTAT

In this section, we show an example of how CTAT can be used to author an Example-Tracing tutor for one- and two-step Algebra equation solving for a given tutor interface. For more details see [9].

To construct an Example-Tracing tutor, one demonstrates behavior directly in the tutoring interface. Traces of these actions are recorded in a behavior graph. A simple Algebra tutor interface and its associated behavior graph are shown in Figure 1.

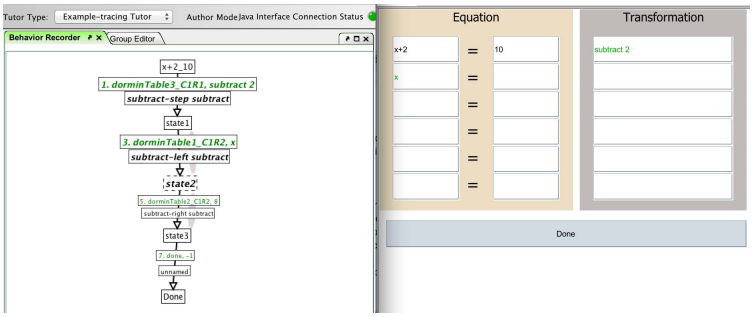


Fig. 1. The Algebra interface and the Behavior Graph produced from demonstrating behavior directly in the interface. The green text specifies the actions taken in the interface and the black text just below shows the author produced skill labels. The ellipsoids between the second and third state (partly occluded by the labels) signify that the actions can be executed in any order.

In this figure we see an interface for tutoring multi-step algebra equation solving (right) and a behavior graph (left). Each node represents a state of the tutoring interface, where the initial state represents the problem to solve. Each link coming out of a node represents an action that might be performed in the state the node represents. In Example Tracing each link is produced as a result of a single action demonstrated directly in the tutor interface, where many legal actions might be demonstrated for each state.

As an example of authoring, consider a tutor for solving the equation $x + 2 = 10$ (using the interface shown in Figure 1). To construct this tutor the author would:

1. Create an empty behavior graph.
2. Input the equation into the interface.
3. Create the initial node of the behavior graph to represent this start state.
4. Demonstrate the first action, subtract 2 from both sides. This demonstration produces a new link in the behavior graph, which the author will label with to the knowledge necessary to perform that action (this label is useful for monitoring learning).

5. Demonstrate the second action, entering x as the new left side of the equation. A second link is produced and labeled in the behavior graph.
6. Next, the third action is demonstrated, entering 8 as the new right side of the equation. A third link is produced and labeled in the behavior graph.
7. The author performs the final action, clicking the done button. This adds a final link to the behavior graph, which the author labels as requiring the done skill.
8. Because the order of the second and third actions doesn't matter, the author either selects both links and marks them as being unordered or returns to the previous state by clicking on the node in the behavior graph and demonstrating the actions in reverse order.

Figure 1 shows the resulting behavior graph (with the second and third links marked as unordered– denoted by the ellipsoids behind the skill names). For a given tutor interface, an author may produce many behavior graphs, each representing a different problem that might be solved in that interface. Other CTAT tools deploy the interface and associated behavior graphs as an ITS, a matter not discussed here.

3 Authoring Using SimStudent

While the Example-Tracing approach has proven effective for authoring, the generality of the model is quite limited. To overcome this limitation the SimStudent architecture was created. This system extends Example-Tracing by inducing more general production rule models from demonstrations and tutoring feedback (for details on this rule induction see [12]). To summarize, SimStudent learns production rules from the demonstrations and refines the conditions on these production rules based on the author's feedback.

The process of authoring a tutor with SimStudent is similar to Example Tracing, in that the SimStudent asks for demonstrations when it does not know how to proceed. However, when SimStudent already has an applicable production rule, it fires the rule and shows the resulting action in the tutor interface. It then asks the tutor author for yes/no feedback on whether this action is correct. Based on the author's feedback, SimStudent refines the conditions of its production rules and proceeds to continue trying to solve the problem. If the author's feedback is negative, SimStudent may exhaust all of its applicable production rules. In these cases, SimStudent asks the user for a demonstration of the correct action. Figure 2 shows how SimStudent communicates with the tutor author to receive a demonstration or feedback.

When authoring models in SimStudent, the author does not have to specify that interface actions are unordered, as one would need to do in Example Tracing, because the production rules learned by SimStudent are applicable in any order, as long as their conditions are satisfied. It is worth noting that the process for authoring a tutor using SimStudent produces both behavior graphs, which might subsequently be used for Example Tracing, and a more general production rule model, which might be used in a full-fledged Cognitive Tutor.

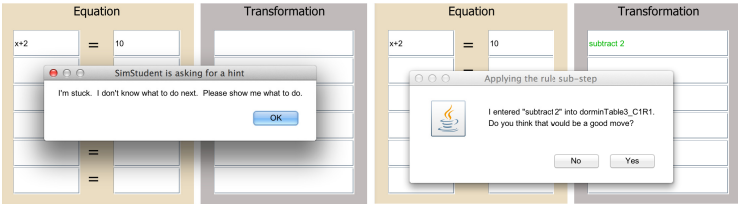


Fig. 2. The image on the left shows SimStudent asking for a demonstration when it does not know how to proceed. The image on the right shows SimStudent asking for feedback on the action it took when it does know how to proceed.

4 Method

An Algebra tutor was authored using both the Example-Tracing and SimStudent approaches. In both cases, the tutor was authored to provide step-by-step feedback on the 20 algebra equations shown in Table 1, where they are organized by the skills necessary to solve them.

We estimated the average authoring time for each approach using the KLM technique, which involved breaking down each authoring action into its primitive steps (many mental pauses, point-and-click actions, and key presses) and then using timing data for how long the average user needs to complete these primitive steps. The KLM provides an accurate prediction of error-free task execution time for an expert user [13]. Both tutors were authored using CTAT and the same Algebra tutor interface, shown in Figures 1 and 2. As shown above, the authoring actions (e.g., providing demonstrations) differ only slightly between approaches; however, the frequency of these actions differs more substantially. In particular, many demonstrations are replaced with feedback when authoring in SimStudent. To compare timings between the two approaches we kept count of the number of authoring actions needed to author each problem, ignoring those actions that were identical between approaches (e.g., creating new behavior graph or start state).

Finally, after each problem demonstration, we evaluated the model quality in terms of the 20 problems that the finished tutor should be able to teach.

Table 1. A tutor was developed to teach these 20 problems using the Example-Tracing and SimStudent approaches. The problem numberings denote the order in which problems were authored, so all problems of the same type were authored together.

Subtract	Add	Divide	Sub + Divide	Add + Divide
1. x+1=10	5. x-5=10	9. 3x=12	13. 5x+2=12	17. 2x-1=1
2. x+2=12	6. x-6=20	10. 4x=8	14. 7x+1=15	18. 3x-3=3
3. x+3=20	7. x-7=14	11. 2x=10	15. 2x+4=8	19. 5x-2=8
4. x+4=4	8. x-2=9	12. 7x=14	16. 3x+6=9	20. 7x-4=10

To evaluate each model we computed a step and recall score, similar to previous studies [12]. The step score equals the number of correct actions suggested by the model divided by the total number of actions (both correct and incorrect) suggested by the model at each step. When the model suggests no actions, the step score is 0. The step score is averaged across all steps to get an overall step score that represents the quality of the model. The recall score is equal to 1 if the model suggests a correct action on a given step and 0 otherwise. The recall score is averaged across all steps to get an overall recall score. Recall assesses how complete a model is, in terms of the percentage of steps that can be tutored.

5 Results

5.1 Authoring Time

Each approach had two authoring actions. Authoring in Example Tracing consisted of demonstrating actions and specifying actions as unordered; whereas, authoring in SimStudent consisted of a slightly longer demonstration and required the author to give feedback on SimStudent's actions. Table 2 shows the number of seconds estimated for each of these actions using the KLM. These estimates were produced by breaking each action down in terms of their primitive steps (mental pauses, pointing and clicking, and keypresses) and summing the time it would take the average user to perform these steps, using previously computed estimates [13].

Figure 3 shows the cummulative time required to author 20 problems using each approach; these estimates were computed by counting the number of tutoring actions needed to author each problem and multiplying these counts by the time estimates shown in Table 2.

5.2 Model Quality

To evaluate the quality of each model we computed the step and recall scores on all 20 problems in the training set after each problem had been authored. This is meant to assess the quality in terms of the 20 problems each model is being built to teach. Figure 4 shows the step and recall scores of each approach after each problem had been authored.

Table 2. The KLM estimates of how long it would take an author to perform each authoring action

Action	**Time (sec)**
Example-Tracing Demonstration	8.8
Example-Tracing Specify Unordered Actions	5.8
SimStudent Demonstration	10.4
SimStudent Feedback	2.4

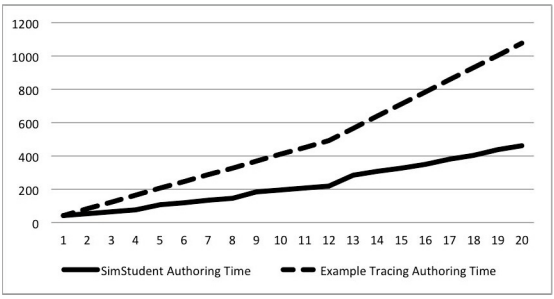


Fig. 3. Cummulative authoring time (in seconds) for each approach, as estimated by the KLM. This model only computes the time needed to perform actions that differ between approaches (demonstrations, specifying ordering, and feedback), so these estimates are slightly less than actual authoring time.

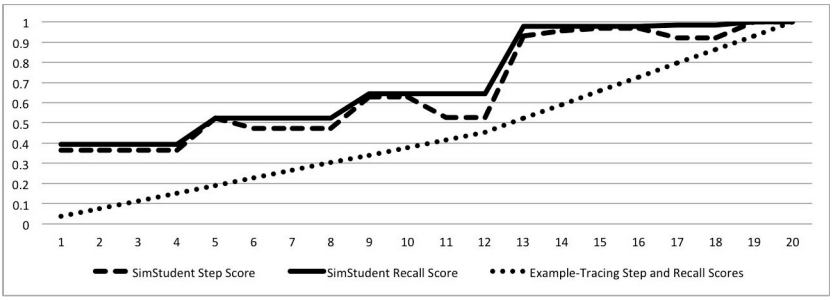


Fig. 4. The step and recall scores computed over all 20 problems after each problem has been authored (the x axis is # of problems authored so far). The Example-Tracing step and recall scores are identical at all points and are just shown with a single line. There is a slight increase in the slope of the Example-Tracing line at problem 12 because the problems transition from one to two step equations.

6 Discussion

The KLM analysis of the two approaches shows evidence that authoring using the SimStudent approach may yield improved authoring efficiency over the standard Example-Tracing approach. This efficiency gain was because SimStudent only required feedback, instead of demonstrations, when it had applicable production rules. Providing feedback (2.4 sec) takes much less time than performing a demonstration (8.8 sec for Example Tracing and 10.4 sec for SimStudent), so this results in a substantial decrease in authoring time. If SimStudent was used solely as a way to improve the efficiency of producing behavior graphs for an Example-Tracing tutor (and not as a way to author more general productions), then it appears authoring efficiency would improve.

When analyzing the model quality of the two approaches, it is important to note that by the end of the authoring process both tutors achieve 100% step and recall scores. However, the process each approach takes to get to 100% is quite different. Figure 4 shows that the Example-Tracing tutor linearly progresses towards perfect scores. Such linear progress is to be expected because it achieves perfect step and recall on all problems that have been authored and 0 step and recall on all problem that have not been authored.

For the SimStudent approach the progression is much different. After the first problem has been authored SimStudent has approximately 40% step and recall scores on the entire set of 20 problems (much larger than the 5% scores for Example-Tracing). This increase is due to the fact that SimStudent is (attempting to) learn general rules from the first problem and some of those rules transfer to the steps in other problems that have similar demands (e.g., knowing that the problem is done when you have x equals some number). After the first problem, SimStudent's step and recall scores jump every time it sees a new problem type because SimStudent learns new production rules to solve these new types of problems (such as adding or dividing) that are useful in solving subsequent problems.

The greater generalization that SimStudent demonstrates within the 20-problem it gets trained on also applies beyond those 20 problems. That is, whereas the Example-Tracing model can only tutor on these 20 problems, the SimStudent model will work on a wider set of problems. For example, the SimStudent model can tutor problems of the same type that have different numbers and minor variations of these problems, such as "(x+2)=9" or "2+x=9." This is why we see plateaus in Figure 4 where SimStudent has already learned how to solve novel problems of the same type. Additionally, SimStudent can tutor some of the steps of more complex problems (e.g., finishing "5x + 10 = 7x" after 7x has been been subtracted from both sides) thus saving time in authoring those more complex problems.

Interestingly, we also observed that as SimStudent gets tutored on new problems its Step score sometimes decreases for previously tutored problems (though never enough to regress below the progress of Example-Tracing). This regression occurs because SimStudent is biased to learn the most general production rule conditions from the examples it sees and thus often overgeneralizes in its early rule acquisition. For example, when generalizing the conditions on the divide rule after getting positive feedback (e.g., when entering divide 2 for 2x=10 – problem #11), SimStudent may learn a rule without a pre-condition specifying a need for a coefficient and thus apply too broadly (e.g., divide 2 for x-2=9). In general, this results in behavior where SimStudent tries to apply productions where they are not applicable, such as trying to add on previous subtraction problems. This overgeneralization might be desirable when trying to model student errors (an application for which SimStudent has been used in the past), but when authoring an expert model of a tutor a decrease in step score on previously authored problems is not desirable. One way to minimize this effect would be to tutor problems in an interleaved vs. blocked fashion, as suggested by previous

work [14]. By regularly returning to older problem types, SimStudent can receive negative feedback in the cases where it has overgeneralized. Alternatively, other approaches could be used to limit SimStudent's overgeneralization, such a using prior knowledge [15] to constrain the generalization.

One limitation of this analysis is that it does not account for the time it takes to develop domain predicates and primitive function operators for the SimStudent system, which are used for production rule learning. These are short pieces of code (roughly similar to writing functions in an Excel spreadsheet), but they do add development time that is not needed in the standard Example-Tracing approach. Despite this additional start-up cost, given the slopes of the lines in Figure 3 the SimStudent approach should eventually result in time savings as more problems are authored. The 16 predicates and 28 function operators used in this study [12] were developed for the algebra domain, but some may be applicable in other domains. Nevertheless, many domains will require new predicates and functions to be hand authored by someone with technical expertise, and this knowledge would need to be tested to ensure that it provides adaquate coverage of the given domain. Li and colleagues [16] have demonstrated how domain specific predicates and functions can be automatically acquired, eliminating or reducing this start-up knowledge engineering, but more work is still needed to demonstrate broader generality of this approach.

To summarize, we found that SimStudent decreases the amount of time needed to author a tutor over the standard Example-Tracing approach. This result is mainly due to the fact that less demonstrations are required with the SimStudent architecture. We also found that by the end of tutor authoring both approaches had equivalent model quality. Furthermore, we showed evidence that SimStudent produces a model that is more general than the specific demonstrations it sees, bridging the gap between an Example-Tracing Tutor and a full-fledged Cognitive Tutor. In some cases SimStudent overgeneralized, and we suggest ways that these overgeneralizations might be reduced. In conclusion, SimStudent appears to be a promising approach for reducing authoring time and producing more general models than standard Example Tracing.

Acknowledgments. This work was supported in part by a Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (#R305B090023) and by the Pittsburgh Science of Learning Center, which is funded by the NSF (#SBE-0836012). This work was also supported in part by National Science Foundation Awards (#DRL-0910176 and #DRL-1252440) and the Institute of Education Sciences, U.S. Department of Education (#R305A090519). All opinions expressed in this article are those of the authors and do not necessarily reflect the position of the sponsoring agency.

References

1. Koedinger, K.R., Corbett, A.T.: Cognitive tutors: Technology bringing learning science to the classroom. The Cambridge Handbook of the Learning Sciences, 61–78 (2006)

2. Beal, C.R., Walles, R., Arroyo, I., Woolf, B.P.: On-line Tutoring for Math Achievement Testing: A Controlled Evaluation. Journal of Interactive Online Learning 6, 1–13 (2007)
3. Graesser, A.C., Vanlehn, K., Rose, C., Jordan, P.W., Harter, D.: Intelligent Tutoring Systems with Conversational Dialogue. AI Magazine 22(4), 39 (2001)
4. Koedinger, K.R., Anderson, J.R.: Intelligent Tutoring Goes To School in the Big City. International Journal of Artificial Intelligence in Education 8, 1–14 (1997)
5. Mitrovic, A., Martin, B., Mayo, M.: Using evaluation to shape ITS design: Results and experiences with SQL-Tutor. User Modeling and User-Adapted Interaction 12(2-3), 243–279 (2002)
6. Ritter, S., Anderson, J.R., Koedinger, K.R., Corbett, A.T.: Cognitive Tutor: Applied research in mathematics education. Psychonomic Bulletin & Review 14(2), 249–255 (2007)
7. Vanlehn, K.: The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. Educational Psychologist 46(4), 197–221 (2011)
8. Murray, T.: An Overview of Intelligent Tutoring System Authoring Tools: Updated analysis of the state of the art. In: Murray, Ainsworth, Blessing (eds.) Authoring Tools for Advanced Technology Learning Environments, pp. 493–546. Kluwer Academic Publishers, Netherlands (2003)
9. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: The cognitive tutor authoring tools (CTAT): Preliminary evaluation of efficiency gains. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 61–70. Springer, Heidelberg (2006)
10. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive Tutors: Lessons Learned. The Journal of Learning Sciences 4(2), 167–207 (1995)
11. Aleven, V., McLaren, B.M., Sewall, J., Koedinger, K.R.: Example-Tracing Tutors: A New Paradigm for Intelligent Tutoring Systems. International Journal of Artificial Intelligence in Education 19(2), 105–154 (2009)
12. Matsuda, N., Cohen, W.W., Koedinger, K.R.: Creating Cognitive Tutors by Tutoring. International Journal of Artificial Intelligence in Education
13. Card, S.K., Moran, T.P., Newell, A.: The keystroke-level model for user performance time with interactive systems. Communications of the ACM 23(7), 396–410 (1980)
14. Li, N., Cohen, W.W., Koedinger, K.R.: Problem Order Implications for Learning Transfer. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 185–194. Springer, Heidelberg (2012)
15. MacLellan, C.J., Matsuda, N., Koedinger, K.R.: Toward a reflective SimStudent: Using experience to avoid generalization errors. In: AIED Workshop on Simulated Learners (July 2013)
16. Li, N., Schreiber, A.J., Cohen, W.W., Koedinger, K.R.: Efficient Complex Skill Acquisition Through Representation Learning. Advances in Cognitive Systems 2, 149–166 (2012)

Implementation of an Intelligent Tutoring System for Online Homework Support in an Efficacy Trial

Mingyu Feng[1], Jeremy Roschelle[1], Neil Heffernan[2], Janet Fairman[3], and Robert Murphy[1]

[1] SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025 USA
{mingyu.feng,jeremy.roschelle,robert.murphy}@sri.com
[2] Worcester Polytechnic Institute, 100 Institute Rd, Worcester, MA 01609 USA
nth@wpi.edu
[3] University of Maine, Orono, ME 04469, USA
janet.fairman@maine.edu

Abstract. Much research has been done on the development of an intelligent tutoring system (ITS), and small empirical studies have demonstrated the effectiveness of ITS at promoting student learning. However, large-scale implementation of ITS in school settings has not been researched thoroughly. In this paper, we describe an ongoing randomized controlled trial (RCT) to evaluate the efficacy of a web-based tutoring system—the ASSISTments—as support for homework. The program is used in 46 middle schools in the state of Maine, to provide immediate feedback to students, and to provide reports to teachers to support homework review and instruction adaptation. We describe the challenges for the RCT, approaches used to understand implementation of the system, and findings on how the system is being used.

Keywords: efficacy, implementation, Intelligent Tutoring System, homework.

1 Introduction

The field of intelligent tutoring systems (ITS) has a long history and many studies have been conducted to show the effectiveness of ITS at improving student learning (e.g., Anderson et al., 1995; Koedinger et al., 1997; VanLehn et al., 2005). Recently, VanLehn (2011) claims that ITS can be nearly as effective as human tutors. Given the promising results found, efforts have been made to introducing ITSs into schools in order to help students learn more effectively (e.g., Koedinger et al., 1997; Arroyo et al., 2009). Most of these research studies have been at a relatively smaller scale within one school, or one school district in short durations. While these studies have the advantages of being more cost-effective and able to show the results quickly, factors such as varieties in school settings, implementation fidelity, counterfactuals, user support, and user-learning curves are typically not well studied and understood. After evaluating the Cognitive Tutors Algebra I (CTAI) curriculum, one of the most well developed ITSs, in a wide variety of middle schools and high schools in seven states for 2 years, Pane et al. (2013) reported there were no effects in the first year of

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 561–566, 2014.

implementation but strong evidence in support of a positive effect in the second year. One possible reason is that the teachers improved their implementation of CTAI or recommended instructional practices after a "warm-up" year of using it (Karam et al., submitted).

Homework is a well-established practice in schools, and the research knowledge base for the effectiveness of homework is also well established (Cooper et al., 2006). Yet, without explicit interventions, homework has been commonly underutilized for improving teaching and learning. Educational technologies have gained popularity in schools (e.g., Khan Academy, DreamBox, IXL.com), but not at home. Most of the computer programs for homework are for college-level populations (e.g., WebAssign, Mastering Physics, OWL) and many have been shown to have positive effects on learning (e.g., Dufresne, et al., 2002; VanLehn et al., 2005). However, there are few rigorous independent studies of the efficacy of online homework in K-12 settings.

The Maine Learning Technology Initiative has implemented one-to-one computing and supplied every seventh-grade student and their teachers with laptop computers, and most middle schools allow students to take their laptops home. In a randomized controlled efficacy trial (RCT), we are investigating whether a web-based ITS as a homework intervention can leverage Maine's one-to-one laptop program to help improve student outcomes in mathematics as measured by a standardized test. Our focus in this paper is on the implementation of the ITS as an online homework support.

2 Background: The ASSISTments System

ASSISTments (www.assistments.org) is a web-based tutoring system that provides "formative assessments that assist." Teachers choose (or add) homework items in ASSISTments and students can complete their homework online. As students do homework in ASSISTments, they receive immediate feedback on the correctness of their answers. Some problem types also provide hints on how to improve their answers, or help decompose multistep problems into parts (see Fig. 1). Teachers may choose to assign problem sets called "skill builders" that are organized to promote mastery learning (Anderson, 2000). Teachers also receive reports on their students' homework and can use this information to organize more targeted homework reviews, to assign specific follow-up work to particular students, and to more generally adapt or differentiate their teaching.

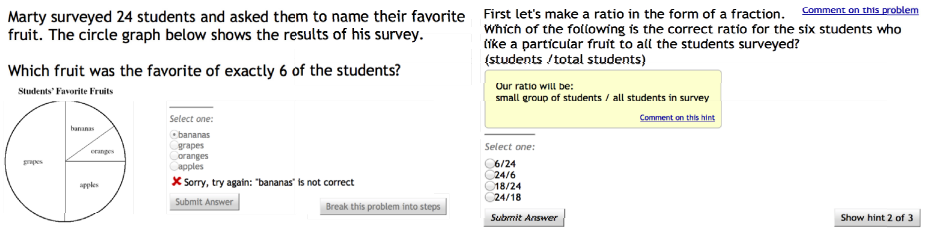


Fig. 1. Screenshots of a seventh-grade problem in ASSISTments that provides correctness feedback and break the problem into steps (left) and the first substep with a hint message (right)

Prior research also has established the promise of ASSISTments for improving student outcomes in middle school mathematics through homework support (Mendicino et al., 2009; Singh et al., 2011; Kelly et al., 2013). While the findings from these studies are encouraging, they only examined tightly controlled implementation of ASSISTments in a few schools over short durations. An investigation was not done regarding the factors that may hinder or facilitate the implementation of the intervention, which is critical for introducing the system to schools at scale.

3 Method

3.1 The Research Design

The study is an independent RCT involving 46 public schools from two cohorts, involving 114 teachers and more than 2,500 students in Maine, with schools randomly assigned to either treatment or control (i.e. "business as usual") conditions. The intervention is implemented in Grade 7 math classrooms in treatment schools over 2 consecutive years (academic years 2012–13 and 2013–14 for Cohort 1 schools and 2013–14 and 2014–15 for Cohort 2 schools). In the treatment condition, teachers receive professional development (PD) and use ASSISTments in the first year to become proficient with the system, and then teachers use ASSISTments with a new cohort of students in the second year when student outcomes are measured.

During the study, teachers in the treatment group are expected to assign approximately 25 minutes of homework in ASSISTments for a minimum of three nights per week, in order to take full advantage of the ITS. Homework assignments are expected to be a mixture of different problem types, including mastery learning problems, reassessment problems that are automatically assigned by the system, and textbook problems. Teachers will receive performance reports early the next morning via email.

The ultimate research question for the study is "*Do students who use ASSISTments for homework learn more than students who do homework without ASSISTments?*" While we are not there yet to answer this question, we hope to address an exploratory question through the data collected in the first implementation year: "*What is the implementation compliance and how much is ASSISTments used by students and teachers on learning?*"

3.2 Collecting Data at Different Stages to Facilitate Implementation

Data collection activities in the first implementation year center on understanding implementation start-up issues and identifying areas of implementation that may require additional support from the developer during the second implementation year.

Before Intervention: Understand the Context and Collect Baseline Data. A good understanding of the context of an RCT and the baseline information of the participants is needed to judge the impact of the intervention and to ensure the successful implementation of the intervention. At the beginning of the study, we conducted a 30-minute interview with principals from each school to learn about existing homework policy, data use, and other initiatives in participating schools. A pre-intervention teacher survey was administered to collect initial data about their current homework

assigning, grading, and reviewing practices; formative assessment and differentiated instruction practices; and how technologies have been utilized to support homework.

During Intervention: Monitor Implementation Fidelity. In contrast to an effectiveness trial, the goal in an efficacy trial is to determine whether an innovation has a beneficial effect in *best-case* implementations. Therefore, it is fair game to monitor and adjust implementation of the innovation. ASSISTments automatically records detailed, time-stamped data of each student and teacher usage (i.e., "the click stream"). Analyzing such data allows us to assess the extent to which students are using the system to complete homework and the extent to which teachers are assigning problems and monitoring students' nightly homework performance. The design of candidate analytics can be guided both by the categories of implementation fidelity (e.g., adherence, exposure, quality of delivery, uptake; Cordray, 2008) and by the pathways in the theory of change. By doing so, a portrait of implementation is presented to the developer team, so that they can ponder: *Is this the quality of implementation we expected as creators of the intervention? What actions can we take that might bring implementation up to our desired levels*?

Halfway through Intervention: Capture Factors That Hinder Implementation. Near the end of the first implementation year, the team conducted face-to-face interviews with a random sample of the teachers to learn about (a) factors that influenced decisions related to homework assignments, (b) teachers' perspective on the impact of ASSISTments, (c) changes in teachers' review routines and instruction strategies, (d) challenges and usability of ASSISTments, and suggestions for improvement.

During Second Year Implementation: Establish Contrast with Counterfactuals. To attribute cause and effect between interventions and outcomes, one critical task of an RCT is to compare the implementation of the intervention with counterfactuals. After a "warm-up," routines have been set up to implement the intervention, and thus the focus of data collection may shift to establish contrast between the two experimental groups. Classroom observation is a powerful tool to capture teachers' practices and their interactions with students. We developed a classroom observation protocol to characterize teachers' reviews of homework and their efforts to adapt instruction. To better understand the motivation behind instruction adaptation, observers follow up with a brief interview.

4 Findings

Below we report preliminary findings from data collected from the first year of teachers and students' usage of ASSISTments.

The **principal interviews** revealed that in general homework is required and assigned almost nightly. This confirms that homework, despite all the controversial discussion regarding its influence on learning (Kohn, 2006), remains a major practice at schools. Teacher support was brought up as one of concerns as there were many demands on teacher's time (e.g., Common Core curriculum integration, meeting AYP goals, etc.) and a new intervention just added to these. We also learned access to the Internet at home is a concern in many schools. These perspectives were brought back to the PD specialist and the system engineers of ASSISTments. A teacher support

plan was then adapted to make it more on-demand and ongoing, to better align with school PD community timelines and topics. An off-line version of ASSISTments was developed to ensure accessibility for all students during the study.

Teachers' responses to the **pre-intervention survey** revealed that teacher's general homework assignment practices align with the specified use model. Notably, even though Maine's laptop initiative has put laptop computers in the hands of every middle school student and teacher ever since 2002, we were surprised that no teachers chose "on laptop" when being asked, "*In what formats do your students usually do their homework?*" Among all of the 31 items in the survey, no significant differences have been detected between responses in the two different conditions.

Compared to self-report or observations, we found using **analytics of system logs** to monitor implementation fidelity is objective, and has lower cost and faster turnaround time. A first useful analytic was how often teachers made assignments with ASSISTments. We found that across 3 months, on average, most teachers assigned homework in ASSISTments 1–2 days in a week with only one teacher meeting the expectation of three assignments per week. Homework completion rates were around 75% and average minutes spent doing homework was 15 minutes. Both values were approximately as expected. A key "uptake" analytic was whether teachers were opening ASSISTments reports as a necessary prelude to adaptive teaching. The ASSISTments trainer was very surprised at the particular teachers who were not opening reports. These findings led to concrete plans of which teachers to follow up in the next round of school visits, what types of behaviors to target during coaching, and a change of the agenda items of the "best practices" workshop.

Although homework could provide data for adjustment of instruction, it is very time-consuming for teachers to aggregate and organize paper-based homework to scan for insights. Therefore, the **teacher interview** focused on the impact of ASSIST-ments reports on homework review. The biggest change reported by the majority of the 12 interviewees is that they can target on the problematic areas identified by the reports. The conversation shifts from checking correctness of every problem to "why" answers were wrong and the process of doing math. The homework review time reduces from 30 minutes to 15 minutes, as one teacher reported. The reports informed their planning and sometimes they had to change their plans when the report suggested students were not ready to move along. Teachers felt students were more engaged in the homework discussion because the discussion was more in time and on target. Based on the feedback from interviews, improvements were made regarding the usability of ASSISTments interface, accessibility of reports, and individual coaching.

5 Conclusion

In this paper, we present approaches used in an efficacy trial being conducted in 46 middle schools in Maine to collect data to understand the implementation of an ITS and thus better interpret the impact of the intervention on student learning. Overall, our recommendation is that researchers who are conducting RCTs to evaluate effectiveness of ITSs or other technology-based interventions in schools should focus on implementation and use different approaches to collect data at different stages to compare the implementation against a program logic model. This can lead to better control of the expected contrast between conditions, which in turn can improve the

quality of the research. The implementation data also provides a unique opportunity for researchers to learn about the value that teachers and students find from the intervention, which is often non-detectable from a 30-item standardized test given at the end of the year. Research methods presented in this paper can be informative to later studies that aim at implementing ITS interventions at scale to a large population.

Acknowledgments. The research reported here was supported by the Institute of Educational Sciences, U.S. Department of Education, through Grant R305A120125 to SRI. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
2. Anderson, J.R.: Learning and memory: An integrated approach, 2nd edn. John Wiley and Sons, Inc., New York (2000)
3. Arroyo, I., Cooper, D., Burleson, W., Woolf, B.P., Muldner, K., Christopherson, R.: Emotion sensors go to school, Conference on Artificial Intelligence in Education (2009)
4. Cordray, D.S.: Fidelity of intervention implementation. Paper presented at the IES, Research Conference (2008)
5. Cooper, H., Robinson, J., Patall, E.: Does homework improve academic achievement? A synthesis of research, 1987-2003. Review of Educational Research 76, 1–62 (2006)
6. Dufresne, R., Mestre, J., Hart, D.M., Rath, K.A.: The effect of web-based homework on test performance in large enrollment introductory physics courses. Journal of Computers in Mathematics and Science Teaching 213, 229–251 (2002)
7. Karam, R., Pane, J.F., Griffin, B.A., Slaughter, M.E.: Evaluating Cognitive Tutor Algebra I curricula at scale: Focus on implementation (submitted)
8. Kelly, K., Heffernan, N., Heffernan, C., Goldman, S., Pellegrino, J., Soffer Goldstein, D.: Estimating the effect of web-based homework. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 824–827. Springer, Heidelberg (2013)
9. Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent tutoring goes to school in the big city. Journal of Artificial Intelligence in Education 8, 30–43 (1997)
10. Kohn, A.: The homework myth: Why our kids get too much of a bad thing. Da Capo Press, Cambridge (2006)
11. Mendicino, M., Razzaq, L., Heffernan, N.T.: Comparison of traditional homework with computer supported homework: Improving learning from homework using intelligent tutoring systems. Journal of Research on Technology in Education (JRTE) 41(3), 331–359 (2009)
12. Pane, J.F., Griffin, B.A., McGaffrey, D.F., Karam, R.: Effectiveness of Cognitive Tutor Algebra I at Scale. Working paper (2013)
13. Singh, R., Saleem, M., Pradhan, P., Heffernan, C., Heffernan, N., Razzaq, L., Dailey, M.: Improving K-12 homework with computers. In: Proceedings of the Artificial Intelligence in Education Conference, Auckland, New Zealand (2011)
14. VanLehn, K.: The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist 46(4), 197–221 (2011)
15. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M.: The Andes physics tutoring system: Lessons learned. International Journal of Artificial Intelligence and Education 15(3), 1–47 (2005)

An Intelligent LMS Model Based on Intelligent Tutoring Systems

Cecilia Estela Giuffra Palomino, Ricardo Azambuja Silveira, and Marina Keiko Nakayama

Departamento de Informática e Estatística
Departamento de Engenharia e Gestão do Conhecimento
Universidade Federal de Santa Catarina - UFSC
Florianópolis – SC – Brasil
cecilia.giuffra@posgrad.ufsc.br,
ricardo.silveira@ufsc.br, marina@egc.ufsc.br

Abstract. Learning Management Systems (LMS) are increasingly being used in Computer-Assisted Education. LMS are used in distance learning and classroom teaching, as teachers and students support tools for learning. Teachers can design and provide material, activities and assessments exercises for the students. Nevertheless, this procedure is usually done in the same way for all the students, regardless of their performance and behavior differences. This research aims to propose an Intelligent Tutoring System (ITS) model to be integrated with Learning Management Systems. The proposed model of ITS is based on Multiagent Systems, in order to provide adaptability to any existent LMS. The main contribution of the presented model is to aggregate the benefits LMS at the ITS and vice versa, creating an intelligent learning environment.

Keywords: Intelligent Tutoring System, Learning Management Systems, Muliagent Systems, User Adaptation, Moodle.

1 Introduction

According to [1] Learning Management Systems (LMS) can be defined as a set of integrated interactive learning tools where the content and pedagogical resources are available online. These tools allow teachers to provide feedback to students in learning activities and are considered important resources for higher education.

LMS are satisfactorily used in e-learning, but they usually do not operate in an interactive and personalized way for the students, by posting tasks and study material according to their characteristics. Usually LMS provide the same pedagogical resources and the same content for all the students, without considering their specific, individual needs.

In order to provide adaptability to the learning environments, according to the student characteristics, and permit a high degree of interactivity between the environment and users, some research points to the use of resources provided by Artificial Intelligence (AI) [14].

The motivation of this research emerges, reflecting the problem of how to enhance the teaching and learning process on LMS using AI techniques in order to make a

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 567–574, 2014.

learning environment adapted to the characteristics of the students, individually in real time. Therefore, this paper proposes an architectural model based on ITS in order to attain personalized learning, exploring the students' skills in the best way, and making learning better and more effective.

1.1 Background

A Learning Management System (LMS) is a software application for the administration, documentation, tracking, delivery of material and assessment. LMS is a database of learning objects and are constituted by a set of technological resources and tools that use cyberspace to transmit content and enable interaction among educational process actors [10]. The use of these systems has increased significantly, due to its ease to provide interaction between students and teachers and the ease of access, from anywhere, at any time, both to the contents as well as the tools of activities and pedagogical mediation offered by the system. Several examples of LMS are available, such as Moodle, Dokeos, Sakai, Caroline, Angel, among others [1].

ITS are complex systems that involve several different types of specialty: domain knowledge, student's knowledge, pedagogical knowledge, among others [6]. According to Santos et al. [13], an ITS is characterized by incorporating AI techniques in its development project and acts as an aid in teaching-learning process. According to Conati [4], ITS is the interdisciplinary field that investigates how to develop educational systems that provide tailored instructions to the students' needs, as many teachers do. Research on ITS is concerned with the construction of environments that enable more efficient learning. The agents' technology made the ITS more tailored to individual needs and the characteristics of each student [8].

ITS have been shown to be highly effective for improving the performance and motivation of students [9]. According to Pereira et al. [12] the convergence of ITS and LMS approaches can potentiates the learning process, making the LMS an intelligent learning environment.

According to Wooldridge [15], an agent is a computer system situated in some environment that is capable of autonomous action in order to meet the objectives that are delegated to it. Intelligent agents are those that have at least the following characteristics: autonomy, reactivity, proactivity, and social ability. An agent is an autonomous entity, able to make decisions, respond on time, pursue goals, interact with other agents, and that have reasoning and character, in addition to having belief, desires and intentions, (BDI). The BDI model represents a cognitive architecture based on mental states, and has its origin in the model of human practical reasoning. An architecture based on the BDI model represents its internal processes through the mental states belief, desire and intention, and defines a control mechanism that selects a rational course of action. [5]

Agents use to inhabit an environment containing other agents, called Multiagent Systems (MAS). The main focus of MAS is to provide mechanisms of computer systems to create a society of agents which interact each other through a shared environment [2]. Recently, proposals for modeling MAS are based on two different abstractions: agents and artifacts (A&A), where the agent is an (pro-) active entity, which is responsible for controlling and accomplishing the goals through tasks, while the artifact is a reactive entity whose functions and services make that individual agents work together in a MAS [11].

1.2 Related Work

In order to know the current state of research on LMS and the use of intelligent agents as tutors in these environments, a systematic review of the literature was performed. Sources used in the research were: Capes Portal[1], IEEEXplore[2], ACM Digital Library[3], and Springer Link[4]. Several papers and articles published between 2004 and 2013 were selected. 14 searches were conducted with different sets of keywords and 46 relevant results, selected by title and abstract, in which there were 31 different research works were found. Of the 31 selected, 22 were available for reading.

Research works related to tutoring in LMS and adaptive environments were found, having as a reference the LMS Moodle. Models of intelligent agents were primarily used. This bibliographical research was conducted to find and analyze the state of the art and similar works that address the adaptability in LMS, taking into account the students' needs, learning styles, usability preferences, etc.

The present work differs from the founded related works by proposing a multiagent based architecture which uses a set of BDI agents who mimic an ITS architecture and obtain all the necessary knowledge (beliefs) from the LMS database to configure the course in a personalized way for each student, delivering subject matter and activities to the student according to their skills, in different levels of difficulty.

2 Model Definition

The architecture of the proposed model is based on a multiagent architecture and a knowledge base of these agents, which compose an ITS functional model that works based on the information obtained from the database of the LMS coupled with the system. The abstract model of the ITS agents was designed as generic as possible and a case study is performed using, as a basis, the architecture of the LMS Moodle in order to build an instance of the model. This LMS was chosen because it is a widely used LMS platform today, and the source code and documentation about it is very easily obtained.

As a part of the model, the interface between the agents and the LMS was defined as an Artifact abstraction that interacts with the LMS database. Thereby the abstract model of the ITS agents is independent of the internal architecture of the LMS coupled with the system and the model deals with any LMS by designing a new Artifact according to the LMS architecture.

An user interface of the system integrated to the LMS was also developed, in which the teacher defines the pedagogical model for the course setting up a tree for providing sequencing resources to the student and the difficulty levels of the proposed activities for the student as well as the priorities of tasks and resources posted by the teacher, in the LMS. The agents use this information to deal with the sequencing learning individually, according to the information of each student in the LMS database.

---

[1] Capes Portal - `http://www.periodicos.capes.gov.br/`
[2] IEEEXplore - `http://ieeexplore.ieee.org/Xplore/home.jsp`
[3] ACM Digital Library - `http://dl.acm.org/`
[4] Springer Link - `http://link.springer.com/`

According to Freedman [7], "The traditional ITS model contains four components: the domain model, the student model, the teaching model, and a learning environment or user interface." In this research the teacher model is represented by the pedagogical model, the domain model by the domain base, and the user interface by the control device. In the proposed model, two types of agents, called the **Bedel Agent** and the **Tutor Agent** are used, respectively. The Bedel Agent, and its whole knowledge and interaction structure mimics the pedagogical model of the classic abstract model of the ITS, while the Tutor Agent, and its whole structure, constitute the student model of the classic abstract model of ITS. The LMS database, in turn, can be associated with the abstraction of the domain base of the ITS. These correlations are shown in Figure 1. The black blocks represent the interface components in the LMS.

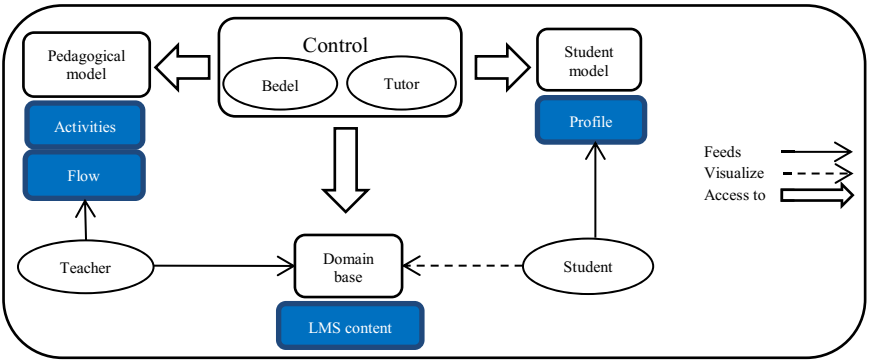


Fig. 1. Classical model with the proposed model

The **Bedel Agent** performs its action as the virtual tutor of the course, setting up the LMS interface, according to this pedagogical model, the published resources in the LMS and the students' performance.

To design the pedagogical model with the proposed teaching strategies, according to these resources, the teacher uses an especially developed tool which is incorporated into the LMS interface. The teacher uses this tool to build a diagram shaped graph, which represents all the possible sequencing for learning, according to the student performance.

The behavior of the Bedel agent is determined by the available resources and activities published in the LMS by the teacher, as well as the pedagogical mediation flow diagram (graph). The plans and beliefs of this agent are influenced dynamically by the changes made in the LMS database insofar as the course takes place and individually for each student based on the performance of each one of them.

The **Tutor Agents** are the agents that have direct contact with the students. They guide the students, indicating changes in their performance, each time an activity is evaluated, encouraging them to improve when they have had a drop in performance or congratulating them when they have achieved better performance.

The agents use available information from the LMS database in order to obtain and store the updated information. The Bedel Agents use this information to update the student profile data and inform the Tutor Agents about the changes in the profile of

the students. The proposed model takes into account a large amount of students, teachers and courses, providing, therefore, the existence of one Bedel Agent for each course and one Tutor Agent for each student. The model works as follow:

- The teacher accesses the course for the first time and the Bedel enabled its configuration block. The teacher inserts the resources and activities in the course and configures the Bedel agent.
- For each course, there is an artifact with an ID that is activated by the Bedel.
- Bedel agent checks the students' achievement in the activities.
- Bedel agent checks the tasks' grades computation held by the teacher, calculates the students' performance and stores in the database the information needed. Then new activities in the LMS become available to the students in a personalized way.
- Tutor agent sends messages to encourage students or congratulate him/her, according to his/her performance.

When the Bedel agent checks the evaluation of the task, made by the teacher, the agent verifies if all the students have been evaluated and, after that, calculates the profile-grade of them, the profile-grade average and the values of each one of the profile levels (basic, intermediate, advanced).

The main goal of the Tutor agent is to verify the change in the student performance and send motivational and feedback messages to him/her. This agent has, in this stage of the project a reactive behavior because it acts after receiving messages from the Bedel agent. However, in the general structure of the model, its performance may be extended, taking into account the overall scenario of the LMS and considering the interaction with Bedel agents from various courses as well as other Tutor agents.

The proposed model assumes that the system comprises four types of actors: the human actor teacher and student; and the agents: Tutor and Bedel. The students are grouped into three different profiles according to their performance (grades) in the accomplished tasks and their access to different resources (teaching materials). This group's separation takes into account the profile-grade of each student (SPG), which is calculated as follows:

$$SPG = \frac{SG + (AG + AcG)}{CN}$$

The sum of the grades (SG) is calculated by multiplying the value of the last profile-grade of the student with the number of times the profile-grade was calculated (last-calculation-number), before the current calculation. The activity-grade (AG) is the grade of the last activity evaluated by the teacher that activated the calculation of the new profile-grade. The access-grade (AcG) is a score that is added to the activity-grade, depending on whether the student read or did not read the content that is a prerequisite of this activity.

The calculation-number (CN) is the amount of times the profile-grade is calculated, including the current calculation. This value is equal to 1 + last-calculation-number.

The last-profile-grade is the student profile-grade at position [last-calculation-number]. The student belongs to the intermediate profile level if his/her profile-grade is between 0.5 more or less than the average of profile-grade of the class. The student who has a higher grade, with a difference of over 0.5 from the average, will have the advanced profile level and the student who has a lower grade, with more than 0.5 of difference, will have the basic profile level.

These values were used due to the fact that in the LMS that was used for the case study, the grades are in the range of 0-10 and students are considered approved with grades larger than 6.

3 Implementation and Tests of the Model

The proposed model integrates concepts from ITS within a well-known LMS that have consolidated use, such as Moodle, which are (by default) not adaptive themselves, and that can be better leveraged using Artificial Intelligence technics, resulting in an intelligent learning environment that are adaptive and are more suitable for the implementation of challenging learning methodologies for the students.

To implement the agents of the system, the Jason tool was used. Jason is an interpreter for an extended version of AgentSpeak language. According to Bordini et al. the most interesting aspects of the AgentSpeak language is that it was inspired and built on the human behavior model, which was developed by philosophers, and is known as the belief-desire-intention (BDI) model. [3].

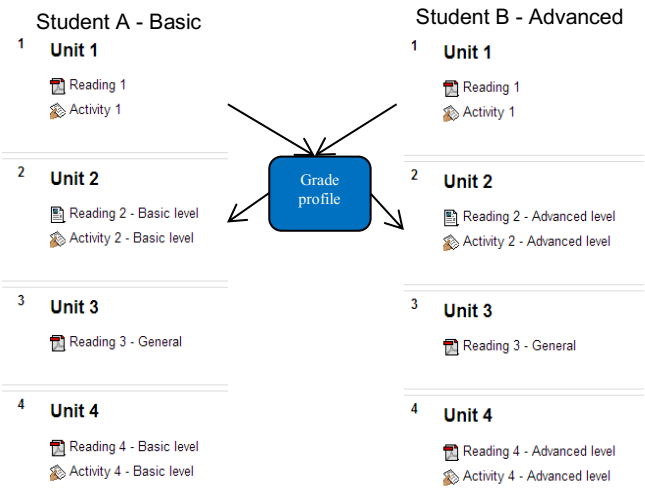


Fig. 2. Student view

According to the research conducted by Al-Ajlan and Zedans [1], The Moodle LMS has a good architecture, robust implementation, interoperability, has nearly the maximum score for functionality expected for an e-learning platform and has the best rating in the adaptation category. For all these reasons, the LMS used in this work, as a case study for the implementation and evaluation of the proposed model, is Moodle.

To test the prototype several adaptations were made in the Moodle LMS code, in order to integrate it with the agents that implement the proposed model. The communication between the LMS and the agents is done through the database, which is updated by both the LMS and the BD artifact, on the agent's side.

Resources and activities are displayed in the LMS in different ways, depending on the students' profile who can change from the basic profile to the intermediate or advanced and vice versa. Figure 2, shows two snapshots of the Moodle screen interface viewed by two students with different profiles (basic and advanced). The general profile is used when an activity or resource is mandatory for the students.

The first resource and the first activity are shown for both students and, after delivering the first activity, a profile is defined for each of them, through the profile grade. Based on this, the resources and activities are shown individually.

4 Remarks and Future Work

In this study an intelligent learning management system based on intelligent tutoring system for large LMS was presented, which could help teachers to provide activities and resources in a customized way according to the student's performance and behavior in the course. Students are continuously assessed by their interaction in the course and through the grades obtained by them in the tasks. According to the results of this assessment, more advanced tasks are provided for students who show a better performance, enabling a more efficient learning, further exploring students' skills, and maintaining a base level for learning the content of the course.

Related Work with LMS and adaptability generally differentiate the students by learning style. In this research, students are differentiated by their performance, taking into account the grades obtained, and their participation (hits) in the various resources available in the course. With this differentiation, it was proved to be possible to create an adaptive environment, based on a conventional LMS, which updates, steadily, the profile of the students, and with it the system behaves adaptively and individually for each student leading the process of teaching and learning by the agents' action, indicating the contents and most appropriate activities.

The main contribution of the presented model is to aggregate the benefits of the learning management system (LMS) at the Intelligent Tutoring Systems (ITS) and vice versa, creating an intelligent learning environment that provides the best of both approaches, combining the robustness and usability of LMS which usually comes from hard learning environments, and the effectiveness of intelligent tutors who offer a much more flexible environment that implements more complex strategies for teaching, but is usually constructed ad hoc for specific areas and with difficulty in reusability.

As future work, we can mention: the improvement of the tutor configuration block, as regards the dependence level between features in order to prevent cyclic dependencies, the reports implementation, to be available to the teacher, with an indicative performance of all the students during the course, performed by agents, and to improve the model so that the teacher doesn't need to input all the possible sequences of the activities and resources, that will be inferred by the agents.

References

1. Ajlan, A.-A., Husein, Z.: Why Moodle. In: International Workshop on Future Trends of Distributed Computgin System. IEEE (2008)
2. Rafael, B., Renata, V., Moreira, A.F.: Fundamentos de sistemas multiagentes. In: Ferreira, C.E. (ed.) Jornada de Atualização em Informática (JAI 2001), vol. 4(1), pp. 3–44. SBC, Fortaleza (2001)

3. Bordini, R.H., Hubner, J.F., Wooldridge, M.: Programming multi-agent systems in AgentSpeak usingJason. John Wiley & Sons (2007)
4. Conati, C.: Intelligent tutoring systems: new challenges and directions. Paper presented at the Proceedings of the 21st International Joint Conference on AI (2009)
5. Mosser, F.: Um ambiente para desenvolvimento de agentes B.D.I. Trabalho de conclusão de curso. Universidade Federal de Pelotas, 2004. Disponível em: http://www.inf.ufsc.br/~silveira/INE602200/Artigos/TCC_Moser.pdf (accessed on December 04, 2011)
6. Claude, F., Thierry, M., Esma, A.: Using pedagogical agents in a multi-strategic intelligent tutoring system. In: Proceedings of the A I-ED 1997 Workshop on Pedagogical Agents, pp. 40–47 (1997)
7. Freeman, R.: What is an Intelligent Tutoring System? Published in Intelligenge, 11(3): 15-16, 2000. Disponível em http://faculty.cs.niu.edu/~freedman/papers/link2000.pdf (accessed on January 20, 2014)
8. Frigo, L.B., Pozzebon, E., Bittencourt, G.: O papel dos agentes inteligentes nos sistemas tutores inteligentes. World Congress on Engineering and Technology Education, São Paulo, Brasil (2004)
9. Lima, R.D., Rosatelli, M.C.: Um sistema tutor inteligente para um ambiente virtual de ensino aprendizagem. Anais do WIE, 2003 (2004)
10. Milligan, C.: Delivering Staff and Professional Development Using Virtual Learning Environments. In: The Role of Virtual Learning Environments in the Online Delivery of Staff Development. Institute for Computer Based Learning, Heriot-Watt University,Riccarton, Edinburgh EH14-4AS. October 1999. Disponível em: http://www.icbl.hw.ac.uk/jtap-573/573r2-3.html (accessed on January 13, 2014)
11. Omicini, A., Ricci, A., Viroli, M.: Artifacts in the A&A metamodel for multi-agent systems. Autonomous agents and multiagent systems 17(3), 432–456 (2008)
12. Pereira Alice, T.C.: Schmitt Valdenise, Álvares Maria R C Dias. Ambientes virtuais de aprendizagem. Livraria Cultura, 2007. Disponível em: http://www.livrariacultura.com.br/imagem/capitulo/2259532.pdf (accessed on November 27, 2011)
13. Santos, C.T., Frozza, R., Dahmer, A., Gaspary, L.P.: Dóris – Um agente de acompanhamento pedagógico em sistemas tutores inteligentes. In: Sbie 2001 Simpósio Brasileiro De Informática Na Educação, 12.,2001, UFES, Vitória-ES (2001)
14. Silveira, R.A.: Ambientes inteligentes distribuídos de aprendizagem. CPGCC da UFRGS, Porto Alegre (1998)
15. Michael, W.: An introduction to multiagent systems, 2nd edn. Jonh Wiley & Sons Ltd (2009)

Designing an Interactive Teaching Tool with ABML Knowledge Refinement Loop

Matej Zapušek[1], Martin Možina[2], Ivan Bratko[2], Jože Rugelj[1], and Matej Guid[2]

[1] Faculty of Education, University of Ljubljana, Slovenia
[2] Faculty of Computer and Information Science, University of Ljubljana, Slovenia

Abstract. Argument-based machine learning (ABML) knowledge refinement loop offers a powerful knowledge elicitation tool, suitable for obtaining expert knowledge in difficult domains. In this paper, we first use it to conceptualize a difficult, even ill-defined concept: distinguishing between "basic" and "advanced" programming style in python programming language, and then to teach this concept in an interactive learning session between a student and the computer. We demonstrate that by automatically selecting relevant examples and counter examples to be explained by the student, the ABML knowledge refinement loop provides a valuable interactive teaching tool.

Keywords: intelligent tutoring, knowledge elicitation, argument-based machine learning, ill-defined concept, programming style, computer programming, python.

1 Introduction

Argument-based machine learning (ABML) knowledge refinement loop offers a powerful knowledge elicitation tool, suitable for obtaining expert knowledge in difficult domains [2,3,6]. Benefits of ABML for knowledge elicitation include: (1) the expert only needs to explain a single example at the time, (2) it enables the expert to provide most relevant knowledge by showing him problematic examples only, and (3) it helps the expert to detect deficiencies in his or her explanations by providing counter examples [3]. In this paper, we would like to verify whether ABML knowledge refinement loop could also be used by students, as an interactive teaching tool based on machine learning and argumentation.

As our case study we selected a difficult, hard to define concept: programming style in python programming language. This language often enables short and elegant solutions. And although the meaning of this latter word is not well defined, it is quite widely accepted in computer programming what has been nicely put by Richard O'Keefe: "Elegance is not optional." [8]

We were particularly interested in distinguishing between "basic" and "advanced" solutions of exercises that typically occur in introductory programming lessons with python as the language of choice. Consider the following solutions: Although both solutions apply to the same exercise, they demonstrate two very different approaches to solve it. In both cases the problem is divided into several

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 575–582, 2014.

```
# Solution 1
def most_different(words):
    most_letters = 0
    for word in words:
        characters = []
        for c in word.lower():
            if not c in characters:
                characters.append(c)
        if len(characters) > most_letters:
            most_letters = len(characters)
            most_diff_word = word
    return most_diff_word
```

```
# Solution 2
def most_different(words):
    return max(words, key=lambda x:len(set(x.lower())))
```

Fig. 1. A "basic" solution (left) and an "advanced" solution (right)

subproblems. However, in Solution 1 each subproblem is expressed separately, while Solution 2 effectively utilizes available built-in functions and mechanisms. The first solution (left) is less sophisticated and clearly a preferred option for the beginners, while the second one (right) is arguably more elegant, more advanced, and perhaps even easier to read by an advanced programmer, but may be difficult to understand for beginners.

While this paper is *not* concerned whether the second solution is better than the first one, our domain expert – a teacher of introductory programming course – labeled solutions such as Solution 1 as "basic," and solutions such as Solution 2 as "advanced." Our goal was to design an interactive tool for supporting students to learn this concept, with respect to distinguishing advanced solutions from the basic ones.

The experts in this domain are generally able to recognize good or bad programming style merely by observing solutions, provided that the solutions are correct, sensible, and complex enough to enable a more advanced approach [1]. The expert should therefore be able to distinguish between "basic" and "advanced" programming style (i.e., between simple and more sophisticated solutions) based on solutions only. In our approach, the text of the exercise did not influence the expert's decisions at all. Our teaching tool should therefore not depend on understanding semantics (or deeper meanings) of the exercise text.

Note that the aim of this paper is *not* to debate what is a suitable programming style and whether the recognition of "elegant" or "advanced" programming style is possible by observing the solutions of programming exercises only (without knowing the instructions of the exercise itself). Nor do we claim that our teacher's views about programming style in python programming language are absolutely correct or indisputable. The goal of this paper is merely to demonstrate the use of argument-based machine learning (ABML) knowledge refinement loop for the purpose of designing an interactive teaching tool. In particular, we intend to demonstrate the use of ABML knowledge refinement loop for: (1) knowledge elicitation of a difficult (even ill-defined) concept from the domain expert – a teacher of introductory computer programming, and (2) student-computer interaction that involves student's argumentation of automatically selected examples and counter examples.

A similar idea, however with a different goal, was explored in a system for smart authoring of automated tutors, *SimStudent*, where students can learn by teaching a live machine-learning agent, using a game-like learning environment [5]. Nan *et* al. showed that an extended version of *SimStudent* successfully learns grammar rules for the difficult task of article selection in English [4].

The paper is organized as follows. In Section 2, we briefly explain the experimental design. Section 3 highlights two important goals of knowledge elicitation from the teacher, namely to obtain (1) relevant description language in the form of new attributes, and (2) consistently labeled learning data. In Section 4, we describe in detail the interactive learning session between a student and the computer, using our (argument-based) teaching tool. Also, the results of an experiment with students learning to distinguish between basic and advanced solutions are presented. We then conclude the paper and point out directions for future work.

2 Experimental Design

From a textbook of introductory programming in python, we selected 121 solutions of 62 different exercises. The teacher labeled each solution as "basic," or "advanced." We randomly selected 91 solutions for learning and 30 solutions for testing (the proportion of positive and negative examples was preserved).

In order to design a successful teaching tool, it was first required to "conceptualize" the domain, that is, to elicitate relevant knowledge from the teacher and transform it into both human- and computer-understandable form. Also, the labels of examples had to be corrected, if necessary. The knowledge in form of attribute values and correct labels had to be incorporated into the teaching tool. Finally, the teaching tool had to be tested by the students. At the end of the interactive session, the students were therefore asked to classify all 30 examples in the test set.

The teaching tool was operated by the teacher. It is essentially based on ABML knowledge refinement loop, and has the following main properties:

1. It is capable of building a rule-based model, using attributes and arguments that are currently included into the domain.
2. It finds "critical examples," i.e. examples that the current model cannot classify successfully, and therefore should be explained.
3. It enables the user to explain given examples in various ways:
   - by introducing (predefined) attributes into the domain,
   - by attaching arguments to selected critical examples,
   - by assigning constraints to particular attributes in the arguments (*high*, *low*, *true*, *false*, *higher*/*lower* than a particular value etc.)
4. It selects appropriate "counter examples," if necessary.
5. It measures the progress of the student (in terms of accuracy of the obtained rules on the unseen test data). However, this information was not disclosed to the students during the experiments.

A detailed description of the ABML knowledge refinement loop can be found in [3] and [7].

3 Knowledge Elicitation from the Teacher

The knowledge elicitation process is described in detail in the next section, where the student-computer interaction is presented. It is actually very similar to that interaction, however, there are two very important differences:

- features (attributes) that would describe the domain well are not yet known,
- labels of examples (given by the teacher) are likely to contain inconsistences.

The goal of the knowledge elicitation from the teacher is therefore not only to obtain a (rule-based) model consistent with his knowledge, but – even more importantly – (1) to obtain relevant description language in the form of new attributes, and (2) to obtain consistently labeled learning data.

This goal is achieved with the help or relevant critical examples and counter examples being presented to the teacher during the interaction. As the teacher is asked to explain given examples or to compare the critical examples to the counter examples, he may introduce new attributes into the domain. The inconsistences are usually found quite easily, since inconsistently labeled examples are likely to appear as critical or counter examples.

In the present case study, the knowledge elicitation process consisted of 9 iterations. Table 1 shows the list of all attributes used: at the beginning of the process 5 of them were included into the domain, and 9 new attributes were introduced by the teacher during the process. Only 1 initial attribute remained in the final model.

Table 1. List of attributes

#	Attribute	Type	Description	Start	Final
1	*cRows*	cont.	number of rows	X	X
2	*cVar*	cont.	number of variables	X	
3	*cFor*	cont.	number of `for` loops	X	
4	*cWhile*	cont.	number of `while` loops	X	
5	*cIf*	cont.	number of conditionals	X	
6	*NeLoop*	T/F	occurrence of nested loop		X
7	*LiCom*	T/F	occurrence of list comprehension		X
8	*cLCbFor*	cont.	number of tokens before the last `for` in list compr.		X
9	*cLCaFor*	cont.	number of tokens after the last `for` in list compr.		X
10	*Zip*	T/F	occurrence of `zip` function		X
11	*cSlice*	cont.	number of list slices		X
12	*Lambda*	T/F	occurrence of `lambda` function		X
13	*cFunc*	cont.	number of built-in functions		X
14	*cMeth*	cont.	number of built-in methods		X

The attributes that occurred in the rules of the final model were included in the interactive teaching tool presented in the next section. The final model contained 9 rules, all of them were found sensible by the expert.

4 A Student-Computer Interactive Learning Session

At the start of the learning session, each student is given the following task: to obtain rules for determining whether a particular solution of (an unknown) programming exercise is an advanced one. The rules must consist of the attributes that remained in the final model obtained by the teacher (see Table 1) only. That is, the goal of the interaction is the student being able to express the target concept using the teacher's expressive language. The instructions were accompanied with a simple example that demonstrates differentiating between a basic and an advanced solution, similar to the one in Fig. 1. In order to facilitate learning, the ABML knowledge refinement loop was used to present the student with relevant examples (and counter examples, if necessary). To accomplish the task in as few iterations as possible, the students are advised to give explanations that:

- contain the most important feature(s) to explain the given example,
- use the smallest possible number of features in a single argument,
- try not to repeat the same arguments.

In the sequel, we demonstrate 4 out of 5 iterations of a typical interaction that actually occurred in one of the learning sessions.

Iteration 1. In the beginning of the interaction, only 5 initial attributes listed in Table 1 were included into the domain, and no arguments were given yet. The solution *A.20-3* (Fig. 2) was the first critical example presented to the student. The student was asked to explain which features speak in favor of this solution being an advanced one. His argument was "the solution is *advanced* because function `zip` is present and the number of rows is low." He also gave an interesting remark that the overall number of different tokens in the solution might have been a more appropriate feature than simply the number of rows.

```
# solution A.20-3 (advanced)
def crossword(word, words):
    return [d for d in words if len(d) == len(word) \\
        and all(c1 == '.' or c1 == c2 for c1, c2 in zip(word, d))]

# solution B.22-3 (basic)
def match(b1, b2):
    b = ""
    for c1, c2 in zip(b1, b2):
        b += c1 if c1 == c2 else "."
    return b
```

Fig. 2. The first "critical example," and the corresponding "counter example"

The student's argument was not sufficiently good: the algorithm selected the (basic) solution *B.14-1* (see Fig. 2) as the counter example. He was asked to compare the counter example *B.14-1* with the critical example *A.20-3* and try to improve the argument. The student noticed an important difference between

the two examples: the advanced solution *A.20-3* contains a list comprehension, whereas the basic solution *B.14-1* does not. He extended the argument to "the solution is *advanced* because `zip` function is present, the number of rows is low, and a list comprehension occurs." There were no more counter examples.

Iteration 2. The (advanced) solution *A.35-1* (see Fig. 3) was then presented to the student. The student now observed a relatively high number of occurring methods (join, split, lower) and chose this as the most important argument. Again he gave an interesting suggestion: namely, that the attribute *cMeth* should have been normalized, taking into account the overall number of tokens in the solution. Another suggestion was to include a new feature: the number of *distinct* methods that occur in the solution.

```
# solution A.35-1 (advanced)
def censorship(text, forbidden):
    return " ".join(word for word in text.split() \\
        if word.lower() not in forbidden)
```

Fig. 3. Solution with "a high number of occurring methods and a list comprehension"

The method now selected a solution from the class "basic" as the counter example. The student quickly noticed several differences between the two solutions, and chose the fact that the advanced one contains a comprehension list to be the most important one among them. The argument was thus extended to "the solution is *advanced* because the number of used methods is high, and a list comprehension occurs.". The algorithm did not find more counter examples.

Iteration 3. was very similar to the second one, thus we skip its description.

Iteration 4. The solution *A.13-3* (see Fig. 4), again an advanced one according to the teacher, was presented to the student. He now selected a new attribute to describe the reasons for the teacher's opinion: the relatively high number of occurring functions.

```
# solution A.13-3 (advanced)
def pairs():
    return [(i, j) for i in range(1, 101) for j in range(i+1, 1001) \\
        if len(str(i)) != len(str(j)) and sum(map(int, str(i))
```

Fig. 4. Another python "one-liner"

After another counter example, the student extended his argument with another unused attribute from the list of available features: a relatively high number of tokens *after* the last `for` statement within the list comprehension. The argument was extended to "the solution is *advanced* because the number of used functions is high, and the number of tokens after the last `for` in the list comprehension is high.". He also suggested that the number of `for` statements within a list comprehension would be another interesting attribute. The extension of the argument worked well: no counter examples were found.

Iteration 5. The student's arguments now resulted in a rule-based model that covered all positive examples for the class "advanced". However, now the algorithm found a problematic example of a different kind: a basic solution that was also covered by one of the obtained rules for the opposite class. The problematic example was the solution *B.34-2*. The key example of the problematic rule was the solution *A.20-2*. The student was now asked to compare these two solutions (see Fig. 5). Namely, what makes the solution *A.20-2* more advanced compared to the solution *B.34-2*.

```
# solution B.34-2 (basic)
def even_vs_odd(s):
    t = sum(e % 2 for e in s) > len(s) / 2
    return [e for e in s if e % 2 == t]

# solution A.20-2 (advanced)
def crossword(word, words):
    return [d for d in words if len(d) == len(word) \\
        and all(c1 == '.' or c1 == c2 for c1, c2 in zip(word, d))]
```

Fig. 5. What makes the solution *A.20-2* more advanced compared to *B.34-2*?

The student chose another yet unused attribute from the list of available features: a relatively high number of tokens *before* the last `for` statement within the list comprehension. After seeing a relevant counter example he extended the argument with the presence of the `zip` function. The argument "the solution is *advanced* because the number of tokens before the last `for` in the list comprehension is high and `zip` function is present." hit upon no counter examples. Moreover, no more critical or problematic examples were detected and thus the learning process concluded.

4.1 Assessment

During the interactive session student therefore also expressed his own (actually very sensible) suggestions on how to introduce new features into the learning domain or how to improve on the existing ones. Such new features can easily be incorporated into the teaching tool. Incidentally, the student's obtained rule model for determining whether a particular solution is advanced even outperformed the teacher in terms of classification accuracy on the testing data (90% vs. 83%; note that the same learning and testing set were used in all experiments). More importantly, the teacher found all given arguments and the obtained rules sensible.

The whole interactive procedure consisted of only 5 iterations and lasted about half an hour. This can be explained by the fact that the student only had to choose among the given attributes (and not yet to discover them). The student was selecting the attributes for his arguments rather skillfully, and in accordance with the given recommendations stated at the beginning of this section.

In the experiment with 7 students, the interactive learning session on average consisted of 7.1 iterations, the classification accuracy of students' final rule models on the testing data were (on average) 87.1% (AUC: 0.74, Brier: 0.25), while they themselves (on average) correctly classified 86.7% examples of the (previously unseen) testing data.

5 Conclusion

We demonstrated the use of argument-based machine learning (ABML) knowledge refinement loop [3,7] for the purpose of knowledge elicitation of a difficult, ill-defined concept of distinguishing between "basic" and "advanced" programming style in python programming language, and used the results of knowledge elicitation for designing an interactive teaching tool. For the first time, ABML knowledge refinement loop was used in an interaction between a *student* and the computer. An interactive learning session between the student and the computer was thus described in detail. The initial experimental results with students are very promising, and suggest that ABML knowledge refinement loop provides a valuable interactive teaching tool. As a line of future work, we consider designing an online multi-domain learning platform based on student's argumentation of automatically selected examples and counter examples.

References

1. Ala-Mutka, K., Uimonen, T., Järvinen, H.M.: Supporting students in C++ programming courses with automatic program style assessment. JITE 3, 245–262 (2004)
2. Groznik, V., Guid, M., Sadikov, A., Možina, M., Georgiev, D., Kragelj, V., Ribarič, S., Pirtošek, Z., Bratko, I.: Elicitation of neurological knowledge with argument-based machine learning. Artificial Intelligence in Medicine 57(2), 133–144 (2013)
3. Guid, M., Možina, M., Groznik, V., Georgiev, D., Sadikov, A., Pirtošek, Z., Bratko, I.: ABML knowledge refinement loop: A case study. In: Chen, L., Felfernig, A., Liu, J., Raś, Z.W. (eds.) ISMIS 2012. LNCS (LNAI), vol. 7661, pp. 41–50. Springer, Heidelberg (2012)
4. Li, N., Tian, Y., Cohen, W.W., Koedinger, K.R.: Integrating perceptual learning with external world knowledge in a simulated student. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 400–410. Springer, Heidelberg (2013)
5. Matsuda, N., Keiser, V., Raizada, R., Tu, A., Stylianides, G., Cohen, W.W., Koedinger, K.R.: Learning by teaching simStudent: Technical accomplishments and an initial use with students. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 317–326. Springer, Heidelberg (2010)
6. Možina, M., Guid, M., Krivec, J., Sadikov, A., Bratko, I.: Fighting knowledge acquisition bottleneck with Argument Based Machine Learning. In: The 18th European Conference on Artificial Intelligence (ECAI), Patras, Greece, pp. 234–238 (2008)
7. Možina, M., Žabkar, J., Bratko, I.: Argument based machine learning. Artificial Intelligence 171(10/15), 922–937 (2007)
8. O'Keefe, R.: The Craft of Prolog. The MIT Press (2009)

Barriers to ITS Adoption: A Systematic Mapping Study

Benjamin D. Nye

University of Memphis
Memphis, TN 38111
benjamin.nye@gmail.com

Abstract. Despite leading to strong learning outcomes, intelligent tutoring systems (ITS) have struggled to reach widescale adoption. However, recent increases in educational technology adoption are slowly leading to larger user bases. Such order-of-magnitude increases have significant research implications for the number and diversity of users. To better understand the problems and solutions that impact this transition, a review of barriers to ITS adoption was performed. This paper significantly extends a prior systematic mapping study of recent ITS literature (2009-2012) focusing on barriers in the developing world. The present study examines research published on possible barriers to adoption related to students, teachers, and school systems. The results indicate that while barriers related to students have received extensive attention, less attention has been given to barriers related to teachers and schools. Successful and innovative approaches to integrating ITS with teacher and school needs are reviewed, with consideration given to both published research papers and successful commercial systems.

Keywords: Intelligent Tutoring Systems, Systematic Mapping Study, ITS Architectures, Barriers to Adoption, Big Data.

1 Introduction

After significant development and investment in intelligent tutoring systems (ITS), current trends indicate that wider adoption may be on the horizon. Overall, educational technology is on the rise: annual investment in educational technology has tripled since 2002 and is becoming more common at all levels of education [7]. Considering that rigorous evaluations of ITS have demonstrated highly significant (0.76σ) learning gains [20], ITS should see an expanded role in educational technology. The rise of Massive Open Online Courses (MOOC's), for example, may present an opportunity for synergy between high-class-size teaching and individualized adaptation (ITS). The growth of educational technology in K-12 schools is another significant trend.

Larger user bases for ITS have sweeping implications. First, more learners could benefit from ITS. Second, a large and sustained user base generates *big data*, potentially orders of magnitude greater than what is currently available.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 583–590, 2014.

Systems like Cognitive Tutor and ASSISTments already have hundreds of thousands of users and store data in open repositories such as the Pittsburgh Science of Learning Center DataShop [8, 13]. One roadblock for ITS has been that evaluating features through lesion studies (i.e., turning features on or off) results in a combinatorial explosion of feature combinations. Larger data sets can be used to test more combinations of features, which offers insight into fundamental questions about the effectiveness of tutoring strategies, multimedia, and individual differences. This ties into a related benefit: greater diversity of users. A limitation for ITS research has been the oversampling of WEIRD (Western, Educated, Industrialized, Rich and Developed) populations [4]. Greater adoption, even if it did not change the distribution of diversity (e.g., percentage of minority users), would increase the raw sample sizes large enough to see if significant effects (e.g., learning gains, motivation) remain significant for subgroups.

These benefits depend on more widespread adoption of ITS, but the transition of educational technology into the hands of learners has often been difficult [3]. In a prior study, a systematic literature review considered barriers to adoption of ITS in the developing world [15]. One takeaway from that study was that barriers found in both most-developed and developing countries are at least as important as those that are unique to developing countries alone. This study follows up on that thread by reviewing a broad range of general barriers to ITS adoption, in an effort to identify barriers that received higher or lower levels of emphasis, barriers that seemed most essential for adoption, and highlighting how existing ITS target these key barriers.

2 Systematic Mapping Study Design

Before conducting this study, a broad set of barriers to adoption related to students, teachers, and school needs was identified. The research question behind this study was: "What fraction of ITS research addresses each barrier to adoption?" In this context, *addressing* a barrier means to direct effort or attention to it within the paper as part of the design or experimental process (e.g., not just part of the background literature review). A systematic mapping design was used, following guidelines from Petersen et al. [16]. The study presented here covers articles and conference papers published no earlier than January 1, 2009 and indexed before January 1, 2013. The inclusion criteria, search methodology, and screening criteria for including papers mirrors Nye [15].

The full text of 2586 papers was reviewed, based a set of citations aggregated from the search phrase "intelligent tutoring system" OR "intelligent tutoring systems" in Thomson-Reuters Web of Science, ACM Digital Library, IEEE Xplore, and ERIC. The primary inclusion criteria for ITS required an inner-loop (i.e., intelligent step-based hints or feedback) as defined by VanLehn [20]. Since this criteria is not always straightforward, a second category of "adaptive learning systems" collected fringe systems with only an outer loop (e.g., selecting the next problem to work on) and possibly rudimentary feedback. The study considered two units of analysis: papers and ITS architecture families.

2.1 Categorization Criteria

Barriers to information and communications technology (ICT) were aggregated from multiple reviews that focus primarily on formal settings in Western countries [3, 14, 17]. From these papers a set of categorization criteria was developed:

1. Independent ICT (Learner): Addresses technologies that enable home or remote use of the ITS, such as web-based ITS.
2. Motivation (Learner): Addresses motivation or employs techniques that are known to impact motivation, such as affect, games, or pedagogical agents.
3. Peer Support (Learner): Addresses peer support or collaborative designs such as computer supported collaborative learning (CSCL).
4. Beliefs (Teacher): Addresses teacher beliefs about ITS utility or expectations.
5. ITS-Integrated Curricula (Teacher): States that the ITS is integrated into an established curricula that teachers might adopt or adapt.
6. Pedagogy Match (Teacher): Addresses the fit of the ITS to teacher pedagogy.
7. Peer Support (Teacher): Describes integration with communities of practice or support for teacher collaboration using the ITS.
8. Time (Teacher): Measures or discusses time costs for adoption, time savings due to adoption, or notes barriers due to lack of time.
9. Training (Teacher): Describes a process for training teachers to use the system or barriers due to lack of training.
10. Administrative Support (School): Addresses administrator buy-in or needs.
11. Technical Support (School): Addresses technical needs or technical support.
12. Assessments (Exosystem): Addresses standardized assessments and tests.
13. Software Cost (Exosystem): States or implies that it is free or low cost.

These factors are broken down by the stakeholder involved: student/learner, teacher, school administration, and exosystem (e.g., geographic or country-level). When examining families of ITS families as units of analysis, a family qualified as addressing a barrier if at least one of their papers did so.

3 Mapping Study Results

The study identified 815 ITS papers on ITS and 240 adaptive learning system (ALS) papers. 373 families of ITS architectures were identified. 36% of ITS papers belonged to 12 major ITS families that had 10 or more papers. 35% of ITS papers described architectures that were discussed only once. ALS tended to be single-paper architectures (80%), which is not surprising since these were not the focus of the search criteria. Adaptive systems mainly described e-learning (over 80% were web-based) or game-based systems (13% were games).

3.1 Student Needs and Barriers

Table 1 shows the percentage of papers and architectures that addressed each student-related barrier. The “Major ITS Families” sample refers to architectures

with more than 10 papers published during the study period. Student motivation received significant attention, particularly among major ITS families. A number of topics related to motivation were also considered by many ITS papers, such as pedagogical agents (31%), affective interaction (18%), games (15%), and metacognition, such as self-regulated learning (4%). Web-based ITS were also a major focus, which can enable student access in a variety of contexts (computer labs, home PC's, etc.). Over the last 4 years, almost all major ITS families reported a web-based version, which could be accessed by students inside or outside of the classroom. Systems such as Andes and AutoTutor Lite explicitly noted that easier setup and access were reasons for building a web-based version and a key use-case for ASSISTments is web-based homework [21, 10, 5]. With that said, few systems focused on increasing access for students who lack home computers, so additional challenges remain in this area (e.g., mobile-only internet users). Peer support and interaction received a moderate level of focus, with over 10% of papers describing a mechanism for student collaboration (e.g., computer-supported collaborative systems), competitive games, teamwork, or content sharing. Overall, barriers related to students received moderate to high emphasis.

Table 1. Research Focus on Student-Centric Barriers

Sample	Independent/ Web ITS	Motivation	Peer Support
All ITS Papers	41.0%	47.2%	11.3%
ITS Families	53.6%	48.0%	17.2%
Major ITS Families	83.3%	91.7%	50%
All ALS Papers	84.6%	31.3%	16.3%

3.2 Teacher Needs and Barriers

Less emphasis was placed on teacher-related barriers, as shown in Table 2. Despite significant attrition by teachers discussed by VanLehn et al. [21], where only 10% continued using Andes, few ITS papers directly researched or addressed teacher factors. Even major ITS families seldom addressed teacher factors in their papers, though many did note them on their project websites.

The barrier that received greatest amount of attention was pre-made curricula: a curriculum unit or course integrated with the ITS. Integrating the ITS into curricula units reduces the burden on teachers, allowing them to adopt or adapt that existing curriculum. Three distinct approaches to this problem were observed: building a full curriculum around the ITS, specifying alignment mappings to existing materials, and building the ITS around a specific popular curriculum. Cognitive Tutor for Algebra took the first approach, supplementing the ITS with an accompanying textbook and full course that are steadily being approved by state curriculum bodies [12]. ALEKS (Assessment and Learning in Knowledge Spaces), an adaptive learning system for math, does not have its own textbook but instead stores mappings that align to curricula (e.g., Common Core) and can embed sections of multiple existing textbooks to align with them [1]. The My Science Tutor

Table 2. Research Focus on Teacher-Centric Barriers

Sample	Beliefs	ITS Curricula	Pedagogy Match	Peer Support	Time	Training
All ITS Papers	1.8%	7.5%	2.5%	0.5%	2.9%	1.6%
ITS Families	3.5%	9.4%	4.6%	0.8%	4.6%	2.7%
Major ITS Families	41.7%	58.3%	33.3%	16.7%	25.0%	25%
All ALS Papers	2.1%	7.1%	1.7%	1.7%	3.3%	3.3%

project took the final approach, building its content around the existing Full Option Science System (FOSS) curricula [22].

Barriers such as training, peer support, beliefs, time, and interactions with pedagogy were mentioned less frequently. These are important, since teacher attitudes and engagement with ICT impact learning gains [18]. Training was not mentioned as a major barrier, though papers noted that teachers received training sessions. Papers that addressed peer support systems between teachers were rare. When mentioned, they were usually noted as a feature rather than a roadblock. For example, ASSISTments tried to increase parental involvement by emailing parents with updates on students' performance on concepts that they were studying [5]. Teacher beliefs about the ITS were stated as a major barrier when they were discussed (i.e., some teachers refused to use an ITS). In some reports, teachers' apriori reactions were influential (e.g., they never even tried the system once). As such, teacher beliefs may tie into general educational technology issues rather than ITS design.

Time and pedagogy issues were mentioned more centrally and were tied to ITS features. A recurring theme from the reviewed papers was that teachers valued saving time and monitoring student outcomes to support their own pedagogy [19]. Systems with high levels of adoption (e.g., ALEKS, Cognitive Tutor, ASSISTments) tend to offer well-developed interfaces for managing class rosters and monitoring estimates of student knowledge. Some go further, using the student model to notify teachers of student impasses or acting as a teacher assistance agent to identify student problems in online classes [6]. Authoring tools that give teachers more control over content have also been developed. In theory, this gives the teacher more control over the pedagogy. Authoring tools were most common in adaptive e-learning systems, but are also available in ASSISTments, the Cognitive Tutor Authoring Tools (CTAT), and various other systems [11]. However, overall use and demand for authoring tools by teachers is generally low: due to time constraints, most teachers probably prefer to select from existing tutoring problems rather than develop new content.

3.3 School and Exosystem

Table 3 displays the percentage of the reviewed papers that consider school-level and external issues. Administrative or technical support factors were seldom mentioned in the reviewed papers. VanLehn et al. [21] was a notable exception, proposing a transition to a web-based system to facilitate multi-platform support

and setup. From a technical standpoint, web-based platforms reduce the burden for installing and updating an ITS. It is unclear how much of the shift toward web-based ITS is motivated by simplifying access and reducing technical support barriers, but they are probably related. Interfaces to help administrators view student and class performance were noted as a desirable feature at the school level by a few papers, but were not discussed as frequently as teacher interfaces.

Table 3. Research Focus on School and Exosystem Barriers

Sample	Administrative Support	Technical Support	Standardized Assessments
All ITS Papers	0.7%	1.8%	2.1%
ITS Families	1.1%	3.0%	3.2%
Major ITS Families	8.3%	25.0%	33.3%
All ALS Papers	1.3%	2.1%	1.3%

Standardized tests were also noted by only a few papers, but systems that mentioned such tests considered them in depth. For example, ASSISTments and Wayang Outpost both support MCAS (Massachusetts Comprehensive Assessment System) and SAT preparation, and have even included mechanisms to project student's scores [9, 2]. These projections model transfer learning and can be better predictors for test item performance than practice items alone, such as a practice test [8]. Since standardized tests are now commonly included in state requirements, projecting their scores may be an increasingly desirable feature for ITS.

The amount of free or low-cost software is not reported in Table 3 because data was incomplete: only about 2% of all publications discussed the topic (0.6% commercial ITS against 1.23% free, open-source, or low-cost ITS). In general, cost does not appear to be a primary barrier. The Cognitive Tutor and ALEKS, two of the most popular systems, are sold commercially. VanLehn et al. [21] noted that teacher perceptions were actually biased toward commercial software rather than free academic software. As such, bias against free software may be a larger barrier than a reasonable fee. Additionally, paid software implies technical support, which is less commonly offered by free or open-source projects.

4 Discussion and Future Directions

To summarize, a large number of papers discussed student-related barriers, while attention to teacher- and school-related barriers was mixed. Compared to the typical ITS, systems with high adoption addressed ITS-integrated curricula, teacher monitoring and customization tools to help teachers include the ITS in their pedagogy, and minimizing teachers' time costs. These may be key barriers that impact adoption and attrition. Features that required serious time investment (e.g., authoring tools) were used infrequently. A number of other barriers were mentioned infrequently overall (training and technical support) or were not

necessarily tied to the ITS design (teacher beliefs on ICT). A few systems researched standardized tests as a type of transfer learning, turning a potential barrier into a feature. This research offers significant value, since standardized tests have real consequences for learners (e.g., graduation, college acceptance) and other educational stakeholders. Finally, cost barriers may work in the opposite direction than expected, with some educators wary of "free software."

A unifying theme for ITS adoption in formal learning settings is the need to communicate with the teacher, administration, or even parents. This typically involves opening the student model to provide reports, predictions, or views that other stakeholders can use to improve educational outcomes. In particularly, teachers "in-the-loop" is critical, since overall learning gains depend on both teacher and ITS interventions. Effective reports from the ITS can save teachers time and help them work the system into their pedagogy. Additionally, understanding the granularity of knowledge that helps a teacher apply their pedagogical strategies should offer insight into both computer and human instruction. Communication in the opposite direction, with the ITS requesting or receiving information directly from teachers, could also help ITS harness judgments that a human can easily make but a computer cannot (e.g., two students shared a computer). Tightening the loop between ITS and teachers may be a step forward for both ITS adoption and effectiveness. By focusing on these adoption barriers, ITS should be able to reach more students, provide better outcomes, and collect more data to help understand fundamental learning processes.

References

[1] ALEKS: K-12 textbook integration (2014), http://www.aleks.com/k12/textbook_integration

[2] Arroyo, I., Woolf, B.P., Royer, J.M., Tai, M., English, S.: Improving math learning through intelligent tutoring and basic skills training. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 423–432. Springer, Heidelberg (2010)

[3] Bingimlas, K.: Barriers to the successful integration of ICT in teaching and learning environments: A review of the literature. Eurasia Journal of Mathematics, Science and Technology Education 5(3), 235–245 (2009)

[4] Blanchard, E.G.: On the WEIRD nature of ITS/AIED conferences: A 10 year longitudinal study analyzing potential cultural biases. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 280–285. Springer, Heidelberg (2012)

[5] Broderick, Z., O'Connor, C., Mulcahy, C., Heffernan, N., Heffernan, C.: Increasing parent engagement in student learning using an intelligent tutoring system. Journal of Interactive Learning Research 22(4), 523–550 (2011)

[6] Casamayor, A., Amandi, A., Campo, M.: Intelligent assistance for teachers in collaborative e-learning environments. Comput. Educ. 53(4), 1147–1154 (2009)

[7] DeSantis, N.: A boom time for education start-ups (2012), http://chronicle.com/article/A-Boom-Time-for-Education/131229/

[8] Feng, M., Heffernan, N.: Can we get better assessment from a tutoring system compared to traditional paper testing? Can we have our cake (Better assessment) and eat it too (Student learning during the test)? In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part II. LNCS, vol. 6095, pp. 309–311. Springer, Heidelberg (2010)
[9] Feng, M., Heffernan, N., Koedinger, K.: Addressing the assessment challenge with an online system that tutors as it assesses. User Modeling and User Adapted Interaction 19(3), 243–266 (2009)
[10] Hu, X., Cai, Z., Han, L., Craig, S.D., Wang, T., Graesser, A.C.: AutoTutor Lite. In: AIED 2009, Amsterdam, The Netherlands, p. 802. IOS Press (2009)
[11] Katuk, N., Sarrafzadeh, A., Dadgostar, F.: Effective ways of encouraging teachers to design and use ITS: Feature analysis of ITS authoring tools. In: Innovations in Information Technology (IIT) 2009, pp. 100–104 (2009)
[12] Koedinger, K.R., Corbett, A.T.: Cognitive tutors: Technology bringing learning science to the classroom. In: Sawyer, R.K. (ed.) The Cambridge Handbook of the Learning Sciences, pp. 61–77. Cambridge University Press, New York (2006)
[13] Koedinger, K.R., Cunningham, K., Skogsholm, A., Leber, B.: An open repository and analysis tools for fine-grained, longitudinal learner data. In: Baker, R., Barnes, T., Beck, J. (eds.) EDM 2008. vol. 157-166 (2008)
[14] Lowther, D.L., Inan, F.A., Strahl, J.D., Ross, S.M.: Does technology integration "work" when key barriers are removed? Educational Media International 45(3), 195–213 (2008)
[15] Nye, B.D.: ITS and the digital divide: Trends, challenges, and opportunities. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 503–511. Springer, Heidelberg (2013)
[16] Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic mapping studies in software engineering. In: EASE 2008, pp. 1–8. IET Publications (2008)
[17] Riasati, M.J., Allahyar, N., Tan, K.E.: Technology in language education: Benefits and barriers. Journal of Education and Practice 3(5), 25–30 (2012)
[18] Rosenthal, R.: Teacher expectancy effects: A brief update 25 years after the Pygmalion experiment. Journal of Research in Education 1(1), 3–12 (1991)
[19] Shanabrook, D.H., Arroyo, I., Woolf, B.P., Burleson, W.: Visualization of student activity patterns within intelligent tutoring systems. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 46–51. Springer, Heidelberg (2012)
[20] VanLehn, K.: The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist 46(4), 197–221 (2011)
[21] VanLehn, K., van de Sande, B., Shelby, R., Gershman, S.: The Andes physics tutoring system: An experiment in freedom. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) Advances in Intelligent Tutoring Systems. SCI, vol. 308, pp. 421–443. Springer, Heidelberg (2010)
[22] Ward, W., Cole, R., Bolaños, D., Buchenroth-Martin, C., Svirsky, E., Vuuren, S.V., Weston, T., Zheng, J., Becker, L.: My science tutor: A conversational multimedia virtual tutor for elementary school science. ACM Trans. Speech Lang. Process. 7(4), 18:1–18:29 (2011)

Towards Scalable Assessment of Performance-Based Skills: Generalizing a Detector of Systematic Science Inquiry to a Simulation with a Complex Structure

Michael A. Sao Pedro[1,2], Janice D. Gobert[1,2], and Cameron G. Betts[2]

[1] Learning Sciences & Technologies Program,
Worcester Polytechnic Institute, Worcester, MA
Apprendis LLC, Stow, MA USA
{mikesp,jgobert}@wpi.edu
[2] Apprendis LLC, Stow, MA USA
cam@apprendis.com

Abstract. There are well-acknowledged challenges to scaling computerized performance-based assessments. One such challenge is reliably and validly identifying ill-defined skills. We describe an approach that leverages a data mining framework to build and validate a detector that evaluates an ill-defined inquiry process skill, designing controlled experiments. The detector was originally built and validated for use with physical science simulations that have a simpler, linear causal structure. In this paper, we show that the detector can be used to identify demonstration of skill within a life science simulation on Ecosystems that has a complex underlying causal structure. The detector is evaluated in three ways: 1) identifying skill demonstration for a new student cohort, 2) handling the variability in how students conduct experiments, and 3) using it to determine when students are off-track before they finish collecting data.

Keywords: science simulations, science inquiry, inquiry assessment, performance assessment, behavior detector, reliability, educational data mining.

1 Introduction

Performance-based assessment tasks, complex tasks that require students to create work artifacts and/or follow processes, are being seen as alternatives to multiple-choice questions because the latter have been criticized as not capturing authentic and relevant "21st century skills" such as critical and creative thinking (e.g. [1]), and scientific inquiry (e.g. [2]). When implemented using computerized simulations [3], games [1] and virtual worlds [2], they have the potential to be scaled because they can be deployed consistently, can automatically evaluate students' work products and processes they follow to create those work products [1], [2], [3], [4], and by virtue of automatic assessment, can provide real-time feedback to students and educators [1], [3]. However, an assessment challenge arises when skills are ill-defined (cf. [1]), meaning that there are many correct or incorrect ways for students to demonstrate skills [5]. How can assessment designers guarantee that the evaluation rules or models

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 591–600, 2014.

they author [1] to identify demonstration of skill within a given task are consistently and accurately doing so? Furthermore, how can they guarantee models will work across different contexts (tasks)?

In this paper, we explore the challenge of creating reliable, scalable evaluation of an ill-defined scientific inquiry process skills in the context of Inq-ITS [3], a simulation-based intelligent tutoring system that also acts as a performance assessment of students' inquiry skills. We determine whether an evaluation model (detector) of an inquiry process skill already shown to generalize for physical science simulations with simple, linear causal structures [6], [7], [8], [9] can also identify the skill in a Life Sciences simulation on Ecosystems that has a complex causal structure (cf. [10]).

2 Prior Work: Validating a Designing Controlled Experiments Detector for Inq-ITS Physical Science Activities

Inq-ITS [3] is a web-based virtual lab environment in which students conduct inquiry with interactive simulations and inquiry support tools. The simulations were designed to tap content areas aligned to middle school Physical, Life, and Earth Science described in Massachusetts' curricular frameworks. Each Inq-ITS activity provides students a driving question, and requires them to investigate that question using the simulation and tools (see Figure 1 for an example Ecosystems activity) in a semi-structured inquiry. More specifically, students attempt to form a testable hypothesis using a pulldown menu-based sentence builder, collect data by changing the simulation's variables and running trials (Figure 1), analyze their data using pulldown menus to construct a claim and by selecting trials as evidence, and communicate findings in an open text field (see [3]). A key aspect of the system is that activities are performance assessments of inquiry skill, because skills are inferred from the inquiry processes they follow and the work products they create with the support tools.

The process skill of focus in this paper is *designing controlled experiments* when collecting data with the simulation. Students design controlled experiments when they generate trials that make it possible to infer how changeable factors (e.g. seaweed, shrimp, small fish, and large fish within an Ecosystem) affect outcomes (e.g., the overall balance of the ecosystem) [6]. This skill relates to application of the Control of Variables Strategy (CVS; cf. [11]), but unlike CVS, it takes into consideration *all* the experimental design setups run with the simulation, not just isolated, sequential pairs of trials [6], [3]. The challenge in assessing this skill is that it is ill-defined; students' data collection patterns can vary widely and there are many ways to successfully demonstrate (or not demonstrate) this process skill [12]. The added difficulty of conducting inquiry in a complex system whose variables interact in nonlinear ways (as opposed to simpler linear systems in which variables have more straightforward dependencies [13]) also contributes to the multitude of ways in which students collect data. This in turn also affects the complexity of assessing this skill.

To address this assessment difficulty, we developed and validated a data-mined detector to determine whether students designed controlled experiments within Inq-ITS physical science activities [6], [7], [8], [9]. We chose a data mining approach to overcome limitations of other models that could under- or over-estimate students' mastery of this skill (e.g. [14]), and to enable easier validation of how well it would perform

by testing it against data not used to build it, thereby addressing issues of reliability and scalability (see [12], [9] for a discussion). Data mining was applied to build models that could replicate human judgment of whether or not students designed controlled experiments. Training and testing labels were generated using text replay tagging of students' log files [15], [6], a process in which human coders tag segments of logfiles (clips) with behaviors or skills. This detector was originally built for a physical science topic on Phase Change as a J48 decision tree. In subsequent work, the decision tree was further improved by choosing features that increased the theoretical construct validity of the detector, and by iterative refinement of the decision tree to find an optimal feature set [7], [9]. Examples of chosen features included the number of data trials collected, how many times the simulation variables were changed, various counts of controlled trials in which only one variable was changed, and various counts for repeated trials with the exact same simulation setup. The detector uses cutoffs of feature values to predict if a student designs controlled experiments.

Overall, we have strong evidence for using this detector to evaluate the designing controlled experiments skill for physical science inquiry activities at scale. For example, as well as being able to predict skill demonstration on held-out test data for Phase Change (the same student sample and simulation from which it was constructed [7]), the models also generalized to predict the same skill within two other physical science topics on energy during free fall [8] and density [9]. The generalization test to the Energy activities also addressed how well the model could handle both new students, and the variability in how they collect data and demonstrate skill [8]. The detector was also validated for a second purpose, determining if a student was off-track when collecting data [7]. In follow-on work, the detector was deployed in Inq-ITS to drive proactive interventions, *before they finished collecting data* in the Phase Change simulation [16], [12]. Thus, the detector could both assess the skill when students finish collecting data, and to drive interventions.

The present study extends this prior work to determine if this detector built and validated for physical science simulations can evaluate the skill and drive interventions for a more complex Life Science simulation on Ecosystems. We adapt our former analytical techniques [6], [7], [8], [9] to address this question.

3 Inq-ITS EcoLife Ecosystems Activities

The EcoLife simulation assesses students' inquiry skills and hones their knowledge of ecosystems. It addresses the two strands of the Massachusetts Curricular Frameworks: 1) the ways in which organisms interact and have different functions within an ecosystem to enable survival, and 2) the roles and relationships among producers, consumers, and decomposers in the process of energy transfer in a food web. The EcoLife simulation (Figure 1) consists of an ocean ecosystem containing big fish, small fish, shrimp, and seaweed. Two inquiry scenarios were developed for this simulation. In the first, students are explicitly told to stabilize the ecosystem. In the second, students are to stabilize the shrimp population (or alternatively, ensure that the shrimp population is at its highest). Students then address the questions by engaging in the inquiry process described earlier.

There are key differences between our physical science simulations and the Ecosystems simulation that can make assessing the designing controlled experiments skill more difficult. For example, unlike the physical science simulations that have discrete choices for variable values [3], in Ecosystems students add and remove organisms with varying numbers. The Ecosystems simulation model is also complex causal system whose multiple variables are interconnected in a non-linear fashion [13], [10], unlike the physical science simulations which have simple linear dependencies [3]. This added complexity increases the hypothesis search space [17], and makes understanding the effects of the independent variables on dependent variable(s) more challenging. As such, the simple control for variables strategy (cf. [11]) may not be applied in a straightforward manner for this task.

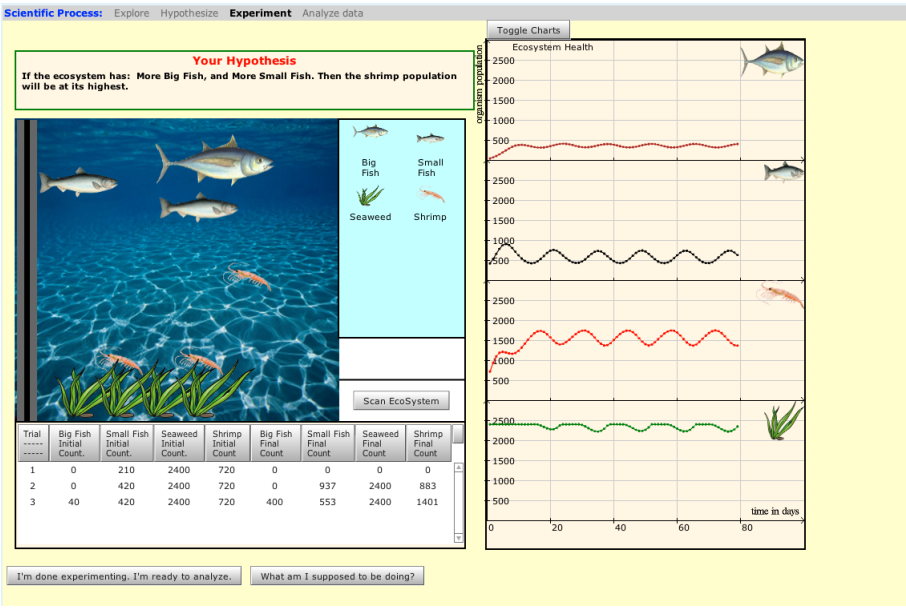


Fig. 1. EcoLife experiment stage. Here, students add and remove organisms, and scan the ecosystem to determine how the population changes over time.

4 Dataset: Distilling Clips from Ecosystems Activities

We collected interaction data from 101 students from a Central Massachusetts middle school who engaged in inquiry with the Ecosystems activities. Then, text replay tagging of log files (clips) [15] was again used to generate a test set for evaluating the applicability of the detector to Ecosystems. A clip contains all actions associated with formulating hypotheses (hypothesize phase actions) and all actions associated with designing and running experiments (experiment phase) [6].

One human coder (the third author) tagged all the clips distilled from the Ecosystems logfiles. A second coder who originally tagged clips in physical science also tagged the first 50 clips to test for agreement. Aside from training the first coder,

determining inter-rater reliability was particularly important because, in addition to its complexity, the Ecosystems environment has a substantially different UI and interaction pattern than the previous physical science simulations [3]. Agreement for the 50 clips tagged by both coders was high overall, $\kappa = .71$, on par with our prior work coding for this skill [6]. In total, 226 clips were tagged, and of those, 52.2% were tagged as the student having demonstrated skill at designing controlled experiments.

5 Results: Generalizability of the Detector to Ecosystems

The overarching goal of this paper is to determine how well the designing controlled experiments detector built and validated for physical science simulations with a simpler, linear causal model, generalizes to predict skill demonstration in a second topic, Ecosystems with a more complex simulation. This goal is important to ensure the model can correctly identify skill in multiple simulation contexts, students and students' experimentation patterns. To do so, three questions are addressed: First, acknowledging that there might be individual differences in how students conduct inquiry in general, can the detector be applied to new students who used the Ecosystems simulation [8]? Second, can the detector handle the variability in how students collect data in Ecosystems [8]? Finally, can the detector be used to determine when scaffolding could be applied when a student is "off-track" [7]?

Commensurate with our prior work on testing the goodness of detectors [6], [7], [8], [9], the degree to which the detector agrees with human judgment (the clip labels described previously) is summarized using two metrics, A' computed as the Wilcoxon statistic [18] and Cohen's Kappa. Briefly, A' is the probability that the detector can distinguish a clip where skill is demonstrated from a clip where skill is not demonstrated, given one clip of each kind. The chance value of A' is .50. Cohen's Kappa (κ) estimates whether the detector is better than chance ($\kappa = 0.0$) at agreeing with the human coder's judgment. A' and Kappa were chosen because, unlike accuracy, they attempt to compensate for successful classifications occurring by chance (cf. [19]). A' can be more sensitive to uncertainty in classification than Kappa, because Kappa looks only at the final label, whereas A' looks at the classifier's degree of confidence.

5.1 Can the Detector Be Applied to New Students in Ecosystems?

The following analysis benchmarks how well the detector handles new students in the new science domain with a more complex simulation [8]. As mentioned earlier, this cohort of students came from a different school than those from which the original detector was built. As shown in Table 1, the detector's performance was quite high and indicate that the detector can be used to evaluate new students' performance in the Ecosystems activities [8]. It could distinguish when a student designed controlled experiments in Ecosystems from when they did not A' = 75% of time. The detector's overall agreement with human judgment of whether a student designed controlled experiments was also quite high, $\kappa = .61$. This performance is on par with previous metrics computed at the student-level across three physical science topics, A' ranging from .82 to .94 and κ ranging from .45 to .65 across studies [7], [8], [9].

Table 1. Confusion matrix and performance metrics computed when applying the designing controlled experiments detector to the Ecosystems clips

	True N	True Y
Pred N	91	27
Pred Y	17	91

Pc = .84, Rc = .77

K = .61, A'=.75

* Pc = precision; Rc = recall

Table 2. Performance metrics for the designing controlled experiments detector disaggregated by number of trials in students' experimentation

Runs	# Clips	A'	K	Pc	Rc
[2,3]	40	.90	.76	.83	.83
[4,5]	39	.64	.44	.78	.67
[6-10]	38	.53	.07	.82	.60
>10	65	.66	.20	.88	.89

* Pc = precision; Rc = recall

5.2 Can the Detector Handle the Variability in How Students Collect Data?

Though the previous results are highly encouraging, they only reveal one aspect of generalizability. We found in prior work that by sampling data according to the variability in students' experimentation patterns, specifically how many trials they collected, we could reveal weaknesses in the detector [8]. We follow a similar process here to characterize how well the detector handles the experimentation variability within Ecosystems. Unlike [8] in which clips were sampled to balance exact counts of trials collected by students (e.g. clips where students collected exactly 4 trials, clips with exactly 5 trials, etc.), here clips were binned into different groups of variability. As an example, one bin contained 40 clips where students collected exactly 2 or 3 trials (Table 2). This deviation was performed because there was greater variability in the number of trials run by students in Ecosystems than in Physical Science. In addition, the number of clips for any specific number of runs was not large enough to generate valid performance metrics. Bins were chosen to both balance the number of clips per bin and to ensure each had enough set of clips for generating metrics.

As shown in Table 2, the detector handled the variability in students' experimentation reasonably well. Performance was high for clips with 2 or 3 simulation runs (A' = .90, κ = .76) and clips with 4 or 5 runs (A' = .64, κ = .44). The detector did, however, struggle on predicting clips with 6 to 10 runs as indicated by A' = .53 and κ = .07 values close to chance. It also did not perform as well for clips with more than 10 runs, A' = .66 and κ = .20, albeit better than chance.

5.3 Can the Detector Identify When Students Are "Off-Track" When Designing Controlled Experiments so That Scaffolds Can Be Effectively Applied?

As mentioned, it is also of interest to determine if the detector can be used to identify when students are off-track by not designing controlled experiments. This is important so that a timely intervention can be given *before* they finish collecting data to prevent floundering [16]. We can determined this by measuring how well the detector can identify skill using less data than was used by the human coder to identify skill [7]. More specifically, we can use a subset of a student's interaction data up to and including the *nth* time the student ran the simulation to predict if a student ultimately did/did not design a controlled experiment. The grain size of "simulation run" was chosen because an intervention given at this point may prevent students from floundering and collecting more confounded data [3], [16].

Like [7], detector performance was measured using data up to a given number of simulation runs. Since there was more variation in how many times the simulation was run in Ecosystem and its increased complexity, detector performance was measured by varying the number of simulation runs from 1 to 10. Again, A' and κ were computed for each simulation run. As shown in Figure 2, the detector can predict if a student is "off-track" when collecting data in Ecosystem in as few as 3 simulation runs, indicated by A' and κ values well above chance, replicating earlier findings [7]. We note the detector performs at chance level for exactly one simulation run because the designing controlled experiments skill can be only identified after the student has collected two or more trials with the simulation (cf. [11]). We also note, however, that as the number of runs exceeds 6, the detector has difficulty distinguishing positive from negative examples. This is indicated by A' values ranging from .58 to .66. The detector, though, still agrees with human judgment fairly well, κ = .41 to .52. The implications of this finding are discussed in the next section.

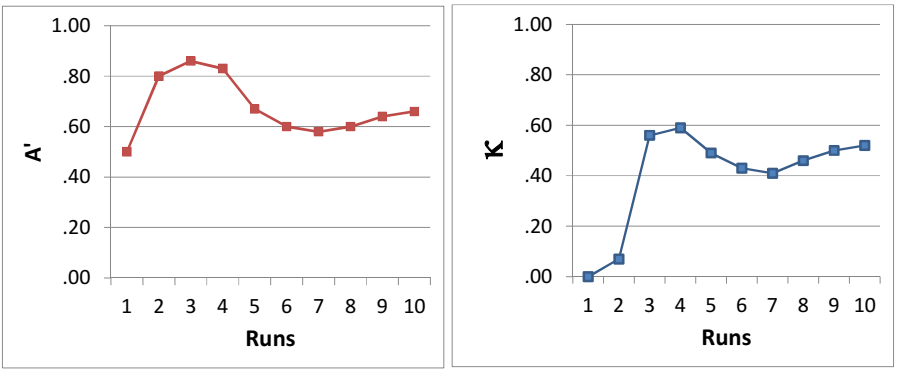


Fig. 2. Designing controlled experiments performance (A' and κ) predicting skill demonstration using data up to and including the *nth* simulation run, *n* = [1,10]. As shown, the detector can be applied in as few as three simulation runs. However, as the number of runs exceeds 6, the detector has difficulty time distinguishing positive from negative examples (indicated by A' closer to chance = .5) even though it still agrees well with human labels (κ >= .40).

6 Discussion and Conclusions

Performance-based assessments (e.g. [1], [2], [3]) present added assessment challenges when the underlying skills they tap are ill-defined (cf. [1]). The main challenge is that such skills may be demonstrated in many correct or incorrect ways by the student (e.g. [5]) which calls to question the reliability and applicability of the underlying assessment models aimed at identifying such skills. Towards the goal of providing reliable, scalable performance-based assessment of inquiry, we determined if a data-mined detector for designing controlled experiments [6], originally built for Physical Sciences simulations [7], [8], [4] that have simpler, linear dependencies between simulation variables, could be applied to the same skill in Ecosystems, a more complex simulation. In brief, we addressed if the detector could: 1) handle student-level validation, 2) assess the multi-faceted ways in which students' conduct inquiry in a complex system, and 3) predict when scaffolding in this domain is needed, a question of importance since the system aims to provide feedback to students as they experiment to prevent them from floundering [3], [16].

The results indicated that the detector had broad generalizability (cf. [20]) given that it could reliably assess the skill within Ecosystems and given its prior success at doing so for physical science simulations [7], [8], [4]. Its performance on the Ecosystems data was akin to that of the physical science simulations [7], [8], [4] under student-level validation. When assessing variability of how students experimented, the detector could identify skill demonstration well when students ran between 2 and 5 trials, but performance dropped when students collected more data than 5 trials. Finally, we found evidence that the detector could detect if a student was "off track" in as few as three simulation runs, commensurate with prior findings within a physical science simulation [7], but also had lower performance as the number of runs increased above 5. One possible way to overcome this limitation as the number of runs increases is to reset the 'window' of students' experimentation patterns after they receive scaffolding, i.e., after a student receives scaffolding, the system could treat the student as if they had not conducted any actions with the simulation. Then, after three more data collections, the system could again determine if the student is still off-track.

This work makes two contributions towards performance-based assessment and generalizability of EDM detectors. First, this study complements prior work on building generalizable detectors of affect (e.g. [21]) and other undesirable behaviors within ITS's (e.g. [20], [22]) with its focus on skill assessment. The power of using the EDM approach to build models that identify skill demonstration is in the ability to *learn* evaluation rules (cf. [1]) from student data, and the ability to quantify how reliable the model is at identifying skills for new students and within different tasks (e.g. physical science vs. life science) by testing detector performance with new student data. Second, as in [7], [8], [9], this study employs additional validation techniques in addition to student-level generalizability tests (e.g. [21], [22]) to determine the extent to which the detector can be used to evaluate skill and drive scaffolding in the more complex domain of Ecosystems. While student-level validation is important, other aspects specific to assessment such as handling variability in how students engage in performance-based tasks and specific to formative assessment such as students get timely feedback so they do not flounder [3] are also necessary if such models are to

generalize to multiple situations. Overall, these results are promising towards realizing scalable assessment and real-time formative feedback of inquiry skill development across science topics. In particular, our computer-based approach complements other assessments of deep science knowledge (e.g. [23]) by focusing on inquiry skills. In addition, since our assessments are performance-based, they may help overcome the limitations associated with assessing inquiry via traditional methods [2].

The generalizability and reusability of the detector has been hypothesized to be due to judicious feature engineering [7]. As such, including other types of features may improve prediction and generalizability. For example, [8] suggests that using ratio-based features instead of a raw counts for features may improve generalizability. For future work, issues such as improved feature engineering will be explored to ensure this detector can work for new students, handle the variability in students experiment, and ensure that scaffolding will be applied at an appropriate time across all Inq-ITS activities for physical, life, and earth science.

Acknowledgements. This research is funded by the National Science Foundation (NSF-DRL#0733286, NSF-DRL#1008649, and NSF-DGE#0742503) and the U.S. Department of Education (R305A090170 and R305A120778). Any opinions expressed are those of the authors and do not necessarily reflect those of the funding agencies.

References

1. Shute, V.: Stealth Assessment in Computer-Based Games to Support Learning. In: Computer Games and Instruction, Charlotte, NC, pp. 503–523. Information Age Publishing (2011)
2. Clarke-Midura, J., Dede, C., Norton, J.: The Road Ahead for State Assessments, Cambridge, MA. Policy Analysis for California Education and Rennie Center for Educational Research & Policy (2011)
3. Gobert, J., Sao Pedro, M., Baker, R., Toto, E., Montalvo, O.: Leveraging educational data mining for real time performance assessment of scientific inquiry skills within microworlds. Journal of Educational Data Mining 4(1), 111–143 (2012)
4. Rupp, A.A., Gushta, M., Mislevy, R.J., Shaffer, D.W.: Evidence-centered Design of Epistemic Games: Measurement Principles for Complex Learning Environments. The Journal of Technology, Learning, and Assessment 8(4), 1–45 (2010)
5. Shute, V., Glaser, R., Raghavan, K.: Inference and Discovery in an Exploratory Laboratory. In: Learning and Individual Differences: Advances in Theory and Research, pp. 279–326. W.H. Freeman, New York (1989)
6. Sao Pedro, M.A., Baker, R.S.J.D., Gobert, J.D., Montalvo, O., Nakama, A.: Leveraging Machine-Learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skil. User Modeling and User-Adapted Interaction 23, 1–39 (2013)
7. Sao Pedro, M., Baker, R., Gobert, J.: Improving Construct Validity Yields Better Models of Systematic Inquiry, Even with Less Information. In: Proc. of the 20th Conf. on User Modeling, Adaptation, and Personalization, Montreal, QC, Canada, pp. 249–260 (2012)
8. Sao Pedro, M.A., Baker, R.S.J.D., Gobert, J.D.: What Different Kinds of Stratification Can Reveal about the Generalizability of Data-Mined Skill Assessment Models. In: Proc. of the 3rd Conference on Learning Analytics and Knowledge, Leuven, Belgium (2013)

9. Gobert, J., Sao Pedro, M., Raziuddin, J., Baker, R.: From Log Files to Assessment Metrics for Science Inquiry using Educational Data Mining. Journal of the Learning Sciences 22(4), 521–563 (2013)
10. Greiff, S., Wustenberg, S., Funke, J.: Dynamic Problem Solving: A New Measurement Perspective. Applied Psychological Measurement 36, 189–213 (2012)
11. Chen, Z., Klahr, D.: All Other Things Being Equal: Acquisition and Transfer of the Control of Variables Strategy. Child Development 70(5), 1098–1120 (1999)
12. Sao Pedro, M.: Real-time Assessment, Prediction, and Scaffolding of Middle School Students' Data Collection Skills within Physical Science Simulations. Ph.D. Dissertation etd-042513-062949, Worcester Polytechnic Institution, Worcester, MA (2013)
13. Yoon, S.: An Evolutionary Approach to Harnessing Complex Systems Thinking in the Science and Technology Classroom. Int'l Journal of Science Education 30(1), 1–32 (2008)
14. McElhaney, K., Linn, M.: Helping Students Make Controlled Experiments More Informative. In: Learning in the Disciplines: Proceedings of the 9th International Conference of the Learning Sciences, Chicago, IL, pp. 786–793 (2010)
15. Baker, R. S. J. D., Corbett, A. T., Wagner, A. Z.: Human Classification of Low-Fidelity Replays of Student Actions. In : Proceedings of the Educational Data Mining Workshop held at the 8th International Conference on Intelligent Tutoring Systems, ITS 2006, Jhongli, Taiwan, pp.29-36 (2006)
16. Sao Pedro, M., Baker, R., Gobert, J.: Incorporating Scaffolding and Tutor Context into Bayesian Knowledge Tracing to Predict Inquiry Skill Acquisition. In: Proc. of the 6th International Conference on Educational Data Mining, Memphis, TN, pp. 185–192 (2013)
17. van Joolingen, W.R., de Jong, T.: An Extended Dual Search Space Model of Scientific Discovery Learning. Instructional Science 25, 307–346 (1997)
18. Hanley, J.A., McNeil, B.J.: The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. Radiology 143, 29–36 (1982)
19. Ben-David, A.: About the Relationship between ROC Curves and Cohen's Kappa. Engineering Applications of Artificial Intelligence 21, 874–882 (2008)
20. Baker, R.S.J.D., Corbett, A.T., Roll, I., Koedinger, K.R.: Developing a Generalizable Detector of When Students Game the System. User Modeling and User-Adapted Interaction 18(3), 287–314 (2008)
21. Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., Heffernan, C.: Population Validity for Educational Data Mining Models: A Case Study in Affect Detection. To appear in the British Journal of Educational Technology (accepted)
22. San Pedro, M.O.C.Z., Baker, R.S.J.D., Rodrigo, M. M.T.: Detecting Carelessness through Contextual Estimation of Slip Probabilities among Students Using an Intelligent Tutor for Mathematics. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS (LNAI), vol. 6738, pp. 304–311. Springer, Heidelberg (2011)
23. Liu, O., Lee, H., Linn, M.C.: Multifaceted Assessment of Inquiry-Based Science Learning, pp. 69–86 (2010)

Automatic Scoring of an Analytical Response-To-Text Assessment

Zahra Rahimi[1], Diane J. Litman[2,4], Richard Correnti[3,4], Lindsay Clare Matsumura[3,4], Elaine Wang[3,4], and Zahid Kisa[3,4]

[1] Intelligent Systems Program
[2] Department of Computer Science
[3] School of Education
[4] Learning Research and Development Center
University of Pittsburgh, Pittsburgh, PA, 15260
{zar10,dlitman,rcorrent,lclare,elw51,zak9}@pitt.edu

Abstract. In analytical writing in response to text, students read a complex text and adopt an analytic stance in their writing about it. To evaluate this type of writing at scale, an automated approach for Response to Text Assessment (RTA) is needed. With the long-term goal of producing informative feedback for students and teachers, we design a new set of interpretable features that operationalize the Evidence rubric of RTA. When evaluated on a corpus of essays written by students in grades 4-6, our results show that our features outperform baselines based on well-performing features from other types of essay assessments.

Keywords: Automatic Essay Assessment, Analytical Writing in Response to Text, Feedback, Natural Language Processing.

1 Introduction

Automatic Essay Assessment can provide a fast, effective and affordable solution to the problem of assessing student writing at scale. The 2010 Common Core State Standards for student learning emphasize the ability of students as young as the fourth grade to construct essays where they interpret and evaluate a text, construct logical arguments based on substantive claims, and marshal appropriate evidence in support of these claims [4]. The Response to Text Assessment (RTA) is developed for research purposes to assess skills at generating analytical text-based writing, and to provide an outcome measure that is independent of a state's accountability test. Specifically, the RTA, unlike available large-scale assessments, is designed to evaluate the integration of reading comprehension and writing skills [4]. Our research takes a first step towards developing an automatic essay assessment system for the RTA. Our goal is to develop a tool that can further large-scale research on the impact of instruction, interventions, and policies that influence the development of this writing skill.

A second goal of our work is to develop a system that could ultimately generate information about students' writing that might be useful for informing instruction. One of the important aspects of the RTA is its multi-dimension rubric,

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 601–610, 2014.

which is used to evaluate students' thinking about the text, their skill at finding evidence to support their claims, and other well-studied criteria associated with effective analytical writing. Such detailed information about students' analytical writing skills is critical in providing informative feedback to students, or giving instructors diagnostic insights into the strengths and weaknesses of students. Thus, an important aspect of our research is designing features for automated assessment that are interpretable given the rating rubrics. While many features previously used in scoring (e.g., Ngrams, part of speech tags, content vectors, Latent semantic analysis, etc.) might yield an automated RTA scoring system with high accuracy, their disconnect from the rubric render them difficult to use as the basis of tutoring or learning analytic systems.

The contributions of our work are as follows. First, analytical response-to text writing is a relatively new domain for the task of automatic assessment. We particularly focus on automatically assessing *Evidence*, which is one of the substantive dimensions of the RTA. Second, we focus on the use of the RTA at the upper elementary level. As such, we tackle the challenge of using computational Natural Language Processing techniques for automation on data that is particularly noisy given the stage of writing development of the students. Finally, our scoring models are based on a new set of features that we designed to reflect the detailed criteria of the rubric related to how students use the reference text. One advantage is that our features are meaningful and interpretable, which should make them useful for producing informative feedback for students and instructors in downstream applications. A second advantage is that our features in fact outperform two baselines based on well-performing features from other types of essay assessment, suggesting the suitability of our approach for the RTA.

2 Related Work

Many essay assessment systems rely on holistic rubrics [1,13,7]. Holistic scoring methods assess the overall quality of an essay by considering multiple criteria simultaneously in order to assign a single score. In contrast, trait-based scoring methods [10,8] can provide multiple scores, as they separately consider component parts or writing purposes when scoring an essay. While holistic methods are typically more efficient and provide more reliable scores, trait-based methods are better at providing diagnostic insight on student performance [16,2]. However, most trait-based scoring systems focus on surface and organizational aspects of writing. In the RTA, substantive dimensions of writing such as Analysis and Evidence[1] are more important[2] [4]. In this paper we focus on assessing the Evidence dimension of the RTA rubric, which is shown in Table 2. The Evidence dimension evaluates how well students use selected details from the text to support and extend a key idea.

[1] There is only a correlation of 0.37 on these dimensions in our data.

[2] The RTA has 5 different rubrics to score the 5 different dimensions: Analysis, Evidence, Organization, Style, MUGS (Mechanics, Usage, Grammar, Spelling).

In terms of writing tasks, most systems (whether holistic or trait-based) focus on assessing writing in response to open-ended prompts [1,13,7,10,5] rather than in response to text. In contrast to the RTA, available assessments tend not to directly measure complex writing skills in which critical thinking and reading are deeply embedded [6,5]. They usually use more generic rubrics instead of task-specific ones. They also do not explicitly evaluate the quality of reasoning based on information from only the text, and instead evaluate dimensions such as structure, elaboration, and vocabulary sophistication [14]. Furthermore, most writing is typically generated by upper elementary, secondary, or post-secondary students [3,6], rather than the younger students targeted by RTA. Our research, which uses the RTA and its task-specific rubrics, takes a step toward evaluating substantive dimensions of analytical writing in response to text.

3 Data

Our research uses the dataset introduced in [4], which is a corpus of essays written by students in grades 4–6. The students first read an article from *Time for Kids* about a United Nations effort to eradicate poverty in a rural village in Kenya, then wrote an essay in response to a prompt. The prompt as well as two student essays are shown in Table 1. Our dataset has a number of properties that may increase the difficulty of the automatic assessment task. The essays in our dataset are short: The average number of words is 161.25 (SD=92.24), while the average number of unique words is 93.27 (SD=40.57). The essays also have many spelling and grammatical errors, and are not well-organized.

The essays are assessed by raters on a scale of 1-4 [4]. Half of the assessments are scored by an expert. The rest are scored by undergraduates trained to evaluate the essays based on the criterion. The currently available corpus contains 1569 essays with 603 of them double-scored for inter-rater reliability checks. Inter-rater agreement (Kappa) on the double-scored part of the corpus on Evidence is 0.42 and Quadratic Weighted Kappa is 0.67. In this paper we only focus on predicting the Evidence ratings, which were produced using the rubric shown in Table 2. An example of a high and low-scoring student essay based on this rubric are shown in Table 1. The distribution of Evidence scores is 469 ones, 594 twos, 335 threes and 171 fours on the full dataset, and 133 ones, 131 twos, 54 threes and 35 fours on the doubly-coded portion where both raters agreed.

4 Features

As discussed above, one goal of our research in predicting Evidence scores is to design a small set of rubric-based meaningful features that perform acceptably and model what is actually important in an essay. In order to help us better understand the process of scoring, our experts first derive a decision tree from the rubric, shown in Fig. 1. To operationalize key decision points in this tree, we develop methods for extracting the following four features from every essay.

Table 1. Sample high and low-scoring essays with highlighted supporting evidence

Prompt: The author provided one specific example of how the quality of life can be improved by the Millennium Villages Project in Sauri, Kenya. Based on the article, did the author provide a convincing argument that winning the fight against poverty is achievable in our lifetime? Explain why or why not with 3-4 examples from the text to support your answer.
Essay with score of 1 on Evidence dimension: *Yes, because even though proverty is still going on now it does not mean that it can not be stop. Hannah thinks that proverty will end by 2015 but you never know. The world is going to increase more stores and schools. But if everyone really tries to end proverty I believe it can be done. Maybe starting with recycling and taking shorter showers, but no really short that you don't get clean. Then maybe if we make more money or earn it we can donate it to any charity in the world. Proverty is not on in Africa, it's practiclly every where! Even though Africa got better it didn't end proverty. Maybe they should make a law or something that says and declare that proverty needs to need. There's no specific date when it will end but it will. When it does I am going to be so proud, wheather I'm alive or not.*
Essay with score of 4 on Evidence dimension: *I was convinced that winning the fight of poverty is achievable in our lifetime. Many people couldn't afford medicine or bed nets to be treated for malaria. Many children had died from this dieseuse even though it could be treated easily. But now, bed nets are used in every sleeping site. And the medicine is free of charge. Another example is that the farmers' crops are dying because they could not afford the nessacary fertilizer and irrigation. But they are now, making progess. Farmers now have fertilizer and water to give to the crops. Also with seeds and the proper tools. Third, kids in Sauri were not well educated. Many families couldn't afford school. Even at school there was no lunch. Students were exhausted from each day of school. Now, school is free. Children excited to learn now can and they do have midday meals. Finally, Sauri is making great progress. If they keep it up that city will no longer be in poverty. Then the Millennium Village project can move on to help other countries in need.*

Table 2. Rubric for the Evidence dimension of RTA

1	2	3	4
Features one or no pieces of evidence	Features at least 2 pieces of evidence	Features at least 3 pieces of evidence	Features at least 3 pieces of evidence
Selects inappropriate or little evidence from the text; may have serious factual errors and omissions	Selects some appropriate but general evidence from the text; may contain a factual error or omission	Selects appropriate and concrete, specific evidence from the text	Selects detailed, precise, and significant evidence from the text
Demonstrates little or no development or use of selected evidence	Demonstrates limited development or use of selected evidence	Demonstrates use of selected details from the text to support key idea	Demonstrates integral use of selected details from the text to support and extend key idea
Summarize entire text or copies heavily from text	Evidence provided may be listed in a sentence, not expanded upon	Attempts to elaborate upon Evidence	Evidence must be used to support key idea / inference(s)

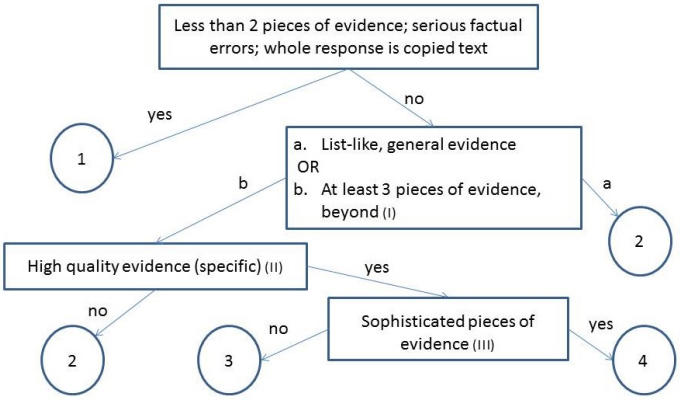


Fig. 1. Decision Tree. I. The evidence is beyond list-like if at least 3 pieces are provided and the student tries to explain the use of evidence in his/her own words, or attempts to connect evidence to his/her thesis. II. High quality evidence includes specific examples from different parts of the text, or an explanation of why the evidence is important. III. The evidence is sophisticated if it is used to support the key idea, and to make inference(s).

Number of Pieces of Evidence (NPE) is defined to capture the first part of the root node of the decision tree: If there are fewer than 2 pieces of evidence, score the essay as 1. For calculating NPE, we use a list of important words for each of the main topics, where the topics and words are defined based on the text and by experts. Any information in the essays that is related to these text-based topics will be considered as a piece of evidence. We use a simple window-based algorithm with fixed window-size[3] to calculate NPE. A window contains evidence related to a topic if there are at least two words from the list of words for that topic. Each topic is only counted as a piece of evidence once to avoid redundancy. NPE is also used by part "b" of the second node of the tree.

Concentration (CON) captures part "a" of the second node of the decision tree. If the essay consists of a not specific, brief list of different pieces of evidence without any elaboration, it has a high concentration and should get the score of 2. We define concentration as a binary feature which indicates if the essay has a high concentration. The high concentration essays have fewer than 3 sentences with topic words. In the case of elaborated evidence, there should be at least three sentences addressing topic words. To calculate this feature, we count the number of sentences that have at least one topic word. If there are less than three sentences with topic words, the concentration is high which means the distribution of topic words in different sentences is low.

Specificity (SPC) is defined to capture the information in the third node of the decision tree. High quality evidence includes specific examples from different parts of the text, or an explanation of why the evidence is important. We extract

[3] For all window-based features, we set the window size value to 6 by trying some different values on a small subset of the dataset and choosing the best value.

a comprehensive list of topics which includes every specific example from the text related to each topic. For each of the examples we need to answer this question of whether the student talked about this specific example or not. So the specificity feature is a vector of integer values. Each value shows the number of examples from the text mentioned in the essay for a single topic. We use the same window based algorithm which we use for NPE to calculate each value of the vector.

Word Count (WOC) is used as a feature because in prior work and in our own data, longer essays tend to receive higher scores. Although word count is not rubric-based, we have not yet defined features to discriminate score 4 due to the difficulty of operationalizing "sophisticated." Until we define such features, we temporarily include word count as a potentially helpful fallback feature.

Based on the defined features, we imagine generating feedback that points students to alternative sources of evidence, that highlights the need to elaborate on the included evidence, or that suggests that students be more specific in their usage of evidence. For example, a student could be given feedback such as "You provided evidence about malaria as condition of poverty that was improved, but there are other relevant evidence in the text that you also need to focus on, such as lack of fertilizer for crops." For teachers, we envision providing summary information such as students' weakness in elaborating on the evidence they provided.

5 Experimental Setup

We configure a series of experiments to test the validity of three hypotheses: H1) the new features will outperform or at least perform equally well as baselines, H2) due to noisy data, spelling correction will improve predictive performance, and H3) word count will be helpful in discriminating the score of 4 from the rest as we have not yet defined features for that part of the decision tree.

In our experiments, we do 10-fold cross validation using 3 different classification methods: Naive Bayes, Random Forest (max depth = 5) and Logistic Regression.[4] Since Naive Bayes is used in [11] (which is one of our evaluation baselines, as discussed below), for comparability we include Naive Bayes as one of our classification methods. Since Random Forest is a decision tree based model and our features are motivated by the decision tree of Fig. 1, we expect this approach to be well-suited for our task. We also include logistic regression to determine whether any observed differences are due to changing features or changing classifiers. Unless otherwise noted, the performance measures reported below are calculated by comparing the baseline and new classifier results with the first human rater's scores. We chose the first human rater because we do not have the scores of the second rater for the entire dataset. The performance measures we report are Accuracy, Kappa and Quadratic Weighted Kappa, which are standard evaluation measures for essay assessment systems.

[4] While we also tried other classifiers like SVM, due to space limitation we only report results for the classification methods that yielded comparable results to the baselines.

For comparing our models and features with existing methods, we consider two different baselines. Baseline1 is one of the best performing methods [11] used in the Hewlett Foundation automated essay scoring competition [15], which was mainly about holistic scoring both on source-based and free-text writing tasks. We choose this baseline because it is an easy-to-implement and open source method: Unigrams and part-of-speech bigrams are extracted and filtered down to the top 500 features by the chi-squared statistic, then a Naive Bayes model is trained on the resulting feature set. Based on some experiments with different Ngram-based features, however, we found that removing part-of-speech bigrams from this model improved performance on our data; therefore, we only use unigrams as features in our experiments. Baseline2 is LSA [9] trained on pre-scored essays and the text. While our first baseline came from the holistic scoring literature, LSA has been successfully used in trait-based systems to score content and ideas [8,12], which seems more similar to our task of scoring Evidence. Since we do not have a separate pre-scored set of training essays, we do cross-validation in our experiments. Scores are assigned based on the scores of the 10 most similar essays, weighted by their semantic similarity based on [12].

6 Results and Discussion

We first examine the hypothesis that our new features will outperform or at least perform equally well as the baselines (H1).[5] The 'comp' columns of Table 3 show the results on the complete dataset. Runs 6 and 7 show that using all 4 new features with either a Random Forest or Logistic Regression classifier yield significantly higher performance than either baseline. Random Forest yields the highest means overall. Run 3 shows that using only the features of Baseline1 (unigrams) with Random Forest does not match the performance of Random Forest and our features, suggesting that our improvements are not just due to changing the classifier of Baseline1. The last three runs show that adding unigrams to our 4 features also do not improve our results. We repeat this experiment using the subset of the doubly-coded portion of the dataset where the 2 raters agreed (353 essays). The 'sub' columns of Table 3 show that these results yield the same conclusions as the 'comp' columns, although the absolute performance figures are even higher on this less noisy part of the dataset (with QWKappa close to the human .67 figure noted in Section 3).

We also examine whether any subsets of our complete 4 feature set could yield comparable predictive performance to using all features. In this experiment we only use Random Forest, as it is the best performing classifier in the experiments above. In each run, we omit one of the features to see if the absence of the feature significantly impacts performance. The results in Table 4 show that removing any of the 4 features significantly degrades model performance compared to using

[5] Since Baseline1 outperforms Baseline2 with one exception (see runs 1 and 2 in Table 3), we focus on comparing our results to Baseline1. Both baselines, in turn, outperform predicting the majority class scores (accuracies of .38 and .37 for the 'comp' and 'sub' portions of the data, respectively).

Table 3. Evaluating performance using 10-fold cross evaluation on both the *complete* (comp) dataset (n=1569), and the *subset* (sub) of the double-coded portion of the dataset (n=603) where the 2 raters agreed (n=353). Significantly better results than Baseline1 are marked by * ($p < 0.05$). The best results are **bolded**.

RUN	Method	Accuracy		Kappa		QWKappa	
		comp	sub	comp	sub	comp	sub
1	Baseline1 (NB + unigrams)	0.52	0.52	0.32	0.28	0.53	0.43
2	Baseline2 (LSA)	0.45	0.43	0.21	0.19	0.47	0.48*
3	RF +unigrams	0.52	0.59*	0.28	0.39*	0.50	0.47*
4	logistic + unigrams	0.49	0.59*	0.27	0.37*	0.52	0.55*
5	NB + 4 features	0.48	0.56*	0.26	0.31*	0.48	0.46*
6	**RF + 4 features**	**0.57***	**0.62***	**0.37***	**0.43***	**0.62***	**0.64***
7	logistic + 4 features	0.55*	0.61*	0.36*	0.41*	0.59*	0.56*
8	NB +unigrams + 4 features	0.52	0.53	0.33	0.29	0.58*	0.45
9	RF +unigrams + 4 features	0.54	0.61*	0.31	0.40*	0.52	0.56*
10	logistic +unigrams + 4 features	0.50	0.60*	0.28	0.40*	0.53	0.60*

Table 4. Performance evaluation of feature subsets on the complete dataset (n=1569). Significantly worse results compared to using all features are marked by $\otimes$ ($p < 0.05$).

Method	Accuracy	Kappa	QWKappa
All(NPE,CON,SPC,WOC)	**0.57**	**0.37**	**0.62**
NPE,CON,SPC	$0.53^{\otimes}$	$0.31^{\otimes}$	$0.57^{\otimes}$
CON,SPC,WOC	$0.54^{\otimes}$	$0.34^{\otimes}$	$0.60^{\otimes}$
NPE,SPC,WOC	$0.55^{\otimes}$	0.35	$0.60^{\otimes}$
NPE,CON,WOC	$0.53^{\otimes}$	$0.32^{\otimes}$	$0.58^{\otimes}$

all 4 features. This suggests that the 4 features capture complementary rather than redundant information.

To evaluate our hypothesis regarding the positive effect of first spell correcting the essays (H2), we repeat the best experimental setting from Table 3 using a 630 essay subset of our dataset where both the original and a manually spell-corrected version of each essay is available; the majority class accuracy for this subset is 0.39. Table 5 shows that spelling correction did indeed improve performance significantly, particularly accuracy by 4%.

Finally, our last hypothesis (H3) is that word count is useful for discriminating score 4 from the rest, as we have not yet defined any rubric-based features for that discrimination. To test this hypothesis, we use Random Forest with all features (All) and after removing word count (All minus WOC) to predict the ratings for 3 different data subsets defined by Evidence ratings: 1) essays rated as 1 and 2; 2) essays rated as 1, 2 or 3; and 3) essays rated as 3 and 4. We also do this comparison using all essays. The results are in Table 6. As can be seen, including word count only significantly improves performance for the data subset that included score 4 (as well as for the complete dataset).

Table 5. The effect of spelling correction (n=630)

Method	Accuracy	Kappa	QWKappa
RF + 4 features	0.52	0.33	0.62
RF + 4 features (spell checked)	**0.56***	**0.36***	**0.65**

Table 6. Performance evaluation of the word count feature. Significant improvements when including word count are marked by * ($p < 0.05$)

Dataset	Features	Majority	Accuracy	Kappa	QWKappa
1,2	All	0.56	0.75	0.48	0.48
	All minus WOC	0.56	0.75	0.49	0.49
1,2,3	All	0.42	0.60	0.36	0.55
	All minus WOC	0.42	0.59	0.35	0.54
3,4	All	0.66	0.66*	0.19*	0.19*
	All minus WOC	0.66	0.63	0.1	0.1
1,2,3,4	All	0.38	0.57*	0.37*	0.62*
	All minus WOC	0.38	0.53	0.31	0.57

7 Conclusion and Future Work

We present results for predicting the Evidence dimension of a rubric developed for the new assessment task of analytical writing in response to text (RTA) using a dataset of essays written by upper elementary school students. We design a new set of rubric-based features that we believe will be more meaningful and interpretable than prior well-performing but generic features like Ngrams and LSA, and compared the predictive utility of our features with these prior baseline features. Our results show that for assessing Evidence, our new methods significantly outperforms baseline methods that performed well on other kinds of automatic essay assessment tasks, and that all 4 features are needed to achieve the best results. We also investigate the impact of one source of noise in the data and find that (manually) correcting spelling errors further improves our results. Finally, we demonstrate that the rubric-based features are particularly valuable for predicting scores when there is a correspondence between the features and where they are used in the decision tree; however, a simple wordcount feature adds value when predicting decisions involving sophisticated evidence, which we have not yet operationalized.

There are still several ways in which our work can be enhanced. Based on our results, we plan to preprocess our data using automated spelling correction as this type of noise was shown to impact Evidence assessment. We would also like to explore using natural language processing techniques to extract topics and words automatically, as our current approach requires these to be manually defined by experts (although this task needs only be done once for each new text and prompt). In addition, we need to improve our implementation of the Specificity feature as well as develop additional features to fully operationalize the Evidence decision tree. We also plan to use natural language processing guided by the RTA rubrics to develop features for predicting the other scoring dimensions. Finally, we plan to examine the generalizability of our current

results, by applying our best-performing model to a new dataset obtained from higher grade levels. Our long term goal is to develop downstream applications based on automated RTA, such as intelligent tutoring systems that can produce informative feedback.

Acknowledgments. This work was supported by the Learning Research and Development Center at the University of Pittsburgh. We thank Huy V. Nguyen who kindly gave us valuable feedback while writing this paper.

References

1. Attali, Y., Burstein, J.: Automated essay scoring with e-rater v.2. Journal of Technology, Learning, and Assessment 4(3) (2006)
2. Bacha, N.: Writing evaluation: What can analytic versus holistic essay scoring tell us? System 29, 371–383 (2001)
3. Burstein, J.C., Braden-Harder, L., Chodorow, M., Hua, S., Kaplan, B., Kukich, K., Lu, C., Nolan, J., Rock, D., Wolff, S.: Computer analysis of essay content for automated score prediction. TOEFL Monograph Series Report No. 13 (1999)
4. Correnti, R., Matsumura, L.C., Hamilton, L.H., Wang, E.: Assessing students' skills at writing in response to texts. Elementary School Journal 114(2), 142–177 (2013)
5. Crossley, S.A., Varner, L.K., Roscoe, R.D., McNamara, D.S.: Using automated indices of cohesion to evaluate an intelligent tutoring system and an automated writing evaluation system. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 269–278. Springer, Heidelberg (2013)
6. Deane, P., Williams, F., Weng, V., Trapani, C.S.: Automated essay scoring in innovative assessments of writing from sources. Writing Assessment 6 (2013)
7. Elliot, S.: Intellimetric: from here to validity. In: Shermis, M.D., Burstein, J. (eds.) Automated Essay Scoring: A Cross Disciplinary Perspective (2003)
8. Foltz, P.W., Streeter, L.A., Lochbaum, K.E., Landauer, T.K.: Implementation and applications of the intelligent essay assessor. In: Shermis, M.D., Burstein, J. (eds.) A Handbook of Automated Essay Evaluation: Current Applications and New Directions, pp. 68–88 (2013)
9. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse Processes 25, 259–284 (1998)
10. Lee, Y.W., Gentile, C., Kantor, R.: Analytic scoring of toefl cbt essays: Scores from humans and e-rater. TOEFL Research Report No. RR 81 (2008)
11. Mayfield, E., Rose, C.: Lightside: Open source machine learning for text. In: Shermis, M.D., Burstein, J. (eds.) A Handbook of Automated Essay Evaluation: Current Applications and New Directions, pp. 124–135 (2013)
12. Miller, T.: Essay assessment with latent semantic analysis. Journal of Educational Computing Research 28(3) (2003)
13. Page, E.B.: Project essay grade: Peg. In: Shermis, M.D., Burstein, J. (eds.) Automated Essay Scoring: A Cross Disciplinary Perspective, pp. 43–54 (2003)
14. Shermis, M.D., Burstein, J.: Automated essay scoring: A cross disciplinary perspective (2003)
15. Shermis, M.D., Hammer, B.: Contrasting state-of-the-art automated scoring of essays: Analysis. In: Annual National Council on Measurement in Education Meeting (2012)
16. Weigle, S.C.: Assessing writing. Cambridge University Press, New York (2002)

Towards Flow Theory on the Design of a Tutoring System for Improving Affective Quality

Po-Ming Lee[1,3], Sin-Yu Jheng[2], and Tzu-Chien Hsiao[2,3,4,*]

[1] Institute of Computer Science and Engineering, National Chiao Tung University, Taiwan (R.O.C.)
[2] Institute of Biomedical Engineering, National Chiao Tung University, Taiwan (R.O.C.)
[3] Department of Computer Science, National Chiao Tung University, Taiwan (R.O.C.)
[4] Biomedical Electronics Translational Research Center and Biomimetic Systems Research Center, National Chiao Tung University, Taiwan (R.O.C.)
labview@cs.nctu.edu.tw

Abstract. Csikszentmihalyi's flow theory states that the components that lead to an optimal state of intrinsic motivation and personal experience may further lead to optimal learning. However, little evidence suggests that a tutoring system (TS) aimed at providing flow preconditions impacts student learning when the contents are the same. Therefore, this study tests this hypothesis by modifying a TS used in an international English language institute (IELI) to provide flow preconditions of students and maintain a balance between the skill level of students and the difficulty level of learning tasks. Fifty-five students in the IELI were separated into two groups to use the modified TS and the original TS. Analysis results indicate an improved engagement and affective quality, as well as reduced frustration levels of the students who used the proposed TS.

Keywords: Tutoring System, Affective Quality, Engagement.

This study examines the influence on providing students with a Tutoring System (TS) that supports the inherent task-related features of flow preconditions [1, 2]; for example, clear goals, immediate feedback, and a balance between challenge and skill [3, 4]. The variables normally measured in association with Csikszentmihalyi's construct of flow are operationalized. This study examines whether students improve in engagement, improve in affective quality, decrease in frustration, and improve in learning performance; when a TS provides flow preconditions with learning contents controlled to be the same.

This study modifies a TS for vocabulary learning that was normally used by English as a second language (ESL) students in an international English language institute (IELI), in order to provide flow preconditions during learning. The flow preconditions were provided to students by loosely incorporating factors that are stated in [3, 4] as follows: 1) a clear goal (the TS instructed the students to answer as many training problems correctly as possible); 2) feedback that is given immediately (although the original TS already included this feature); 3) adaptive tasks, which were implemented

* Corresponding Author.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 611–612, 2014.

by using a task-selection controller that balanced the personalized difficulty level of a selected task with a student's skill level at each task loop; and 4) enhanced concentration of the students, by applying a time limit to each task loop.

Study participants consisted of 55 ESL students from the IELI Reading 3 and Reading 4 classes (intermediate ESL students). The participants had not learned the given vocabulary words at IELI. Each student completed the study in one to two weeks. In total, 43 students completed the study. The experiment and the manner in which data obtained from human subjects was used received approval from the local Institution Review Board (IRB).

Learning performance was evaluated using one-way ANOVA. Random assignment appears to achieve a balance across all groups in terms of the incoming student competency. No statistically significant differences were found between the two groups in the pretest scores ($p = 0.19$). Additionally, the two groups did not significantly differ in the total training time spent on the TSs. Differences in learning performance between the pre- and posttests were also evaluated using one-way ANOVA. Both groups made significant gains from pretest to posttest $p = 0.01$ for the experimental group and $p = .009$ for the control group.

However, statistically significant differences were found between the two groups in the ratings of engagement (items "The activity is fun" and "I find the activity pleasurable": $p = .01$ and $p = .03$ respectively), affective quality (see item "The activity is adequate, neither too difficult nor too easy" and "I enjoy the activity without feeling bored or anxious": $p = .003$ and $p = .001$, respectively), and frustration (items "The activity makes me tired", "The activity is difficult", and "The activity is boring": $p = .003$; $p = .03$; and $p = .04$, respectively). Based on the measurement results, the experimental group may have had a significantly better experience than that of their controlled peers.

Acknowledgements. This work was fully supported by the Taiwan Ministry of Science and Technology under grant numbers NSC-102-2220-E-009-023 and NSC-102-2627-E-010-001. This work was also supported in part by the UST-UCSD International Center of Excellence in Advanced Bioengineering sponsored by the Taiwan Ministry of Science and Technology I-RiCE Program under grant number NSC-101-2911-I-009-101; and in part by "Aim for the Top University Plan" of the National Chiao Tung University and Ministry of Education, Taiwan, R.O.C.

References

1. Csikszentmihalyi, I.S.: Optimal Experience: Psychological Studies of Flow in Consciousness. Cambridge University Press (1992)
2. Csikszentmihalyi, M.: Flow: The Psychology of Optimal Experience. Harper Perennial (1991)
3. Kiili, K.: Digital Game-Based Learning: Towards an Experiential Gaming Model. The Internet and Higher Education 8, 13–24 (2005)
4. Sweetser, P., Wyeth, P.: Gameflow: A Model for Evaluating Player Enjoyment in Games. Computers in Entertainment 3, 3 (2005)

Engaging Higher Order Thinking Skills with a Personalized Physics Tutoring System

Matthew Bojey[1], Bowen Hui[1], and Robert Campbell[2]

[1] Department of Computer Science, University of British Columbia
[2] Faculty of Education, University of British Columbia

Abstract. Recent research shows a lack of student interest and declined enrollment in physics. Our system offers four levels of difficulty with activities that enable students to exercise a range of lower and higher order cognitive skills. Moreover, we adopt existing methods in probabilistic user modeling to provide personalized help. Our work models both domain concepts as well as user attitudes.

Keywords: Probabilistic user modeling, dynamic Bayesian networks, higher order cognitive skills, Bloom's taxonomy, physics education.

1 Introduction

Research shows a declined enrollment in university physics programs and that physics as taught in schools do not seriously take student interests into account [4]. As a way to overcome this challenge, attempts to integrate physics material in an interactive and individualized manner have shown to increase student interest and performance (e.g., [3,5]). However, students' interest in physics is closely related to their self-esteem and sense of academic achievement [4]. Fostering student interest requires teachers to pay close attention to students and guide them. Unfortunately, large-sized classes make it logistically infeasible to realize this. We propose an intelligent tutoring system (ITS) that provides individualized feedback and aims to increase student interest in physics by providing a variety of activities that exercise different levels of thinking skills.

Originally devised as handbooks to systematize learning objectives and assessment, Bloom's taxonomy has become a foundational structure in Education [1]. The taxonomy reprsents the process of mastering a subject through several levels of cognitive activities starting wtih remembering at the lowest level and creating at the highest.Our objective in this work is to adopt Bloom's taxonomy to create interactive activities that enable students with varying expertise to apply a range of cognitive skills.

2 System Overview

Our system is called Kirchhoff's Rules Intelligent Tutor (KRIT) as it focuses on helping students with the application of Kirchhoff's rules. The complexity of a

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 613–614, 2014.

circuit is determined by the layout parameters (number of batteries, resistors, and junctions). The objective of these exercises is to apply Kirchhoff's rules and algebraically solve for one of three variables (voltage, resistance, current). KRIT has four activity types displayed as separate levels to the student, multiple choice questions, coached exercises, guided exercises and a create and share activity.

Our student model represents how much help a student currently needs as well as thier current level of understanding. These factors are crucial in developing a personalized ITS because different types of students prefer different levels of assistance regardless of their level of understanding. These preferences may also vary as a function of the exercises' difficulty. Hence, a personalized tutor must provide support suited for the individual's needs.

Since this type of information is unknown to KRIT, it must be inferred indirectly. Due to the inherent uncertainty of the inference problem, we adopt a probabilistic approach by using a dynamic Bayesian network (DBN) [2]. Our DBN reflects how the student's domain knowledge (K for short) influences her performance on applying algebra and physics concepts. We use observable events to estimate the student's understanding of these concepts thus, the DBN incorporates all the events observed from one response at each stime step.

To model student characteristics, our DBN includes the current need for help and receptiveness to help (N and R for short respectively). We estimate these variables using passively collected behavioral observations such as pausing, undoing what was typed, and making use of hints and system explanations. Additionally, note that K influences N to model advanced students are less likely to need help. In turn, N influences R to model the correlation between neediness and receptiveness to automated help. Together, K and N define the student's *current state* which is used to inform KRIT in its decision making.

3 Conclusions and Future Work

Pilot studies have shown promising results and we are currently in the process of performing a large scale usablity test. We also have plans to develop the system for mobile platforms and perform a longitudinal study.

References

1. Bloom, B.S.: Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. David McKay Company, Inc., New York (1956)
2. Dean, T., Kanazawa, K.: A model for reasoning about persistence and causation 5(3), 142–150 (1989)
3. Gertner, A.S., VanLehn, K.: Andes: A coached problem solving environment for physics. In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 133–142. Springer, Heidelberg (2000)
4. Häussler, P., Hoffmann, L.: A curricular frame for physics education: Development, comparison with students interests, and impact on students achievement and self-concept. Science Education 84, 689–705 (2000)
5. Myneni, L.S., Narayana, N.H.: An Intelligent Tutoring and Interactive Simulation Envioronment for Physics Learning, pp. 250–255 (2012)

Scaffolding Reflection for Collaborative Brainstorming

Andrew Clayphan[1], Roberto Martinez-Maldonado[1], Judy Kay[1], and Susan Bull[2]

[1] School of Information Technologies, University of Sydney, NSW 2006, Australia
[2] Electronic, Electrical and Computer Engineering, University of Birmingham, UK
{ajc,roberto,judy}@it.usyd.edu.au,
s.bull@bham.ac.uk

Abstract. We present a reflection-on-action system supporting students' reflection and self-assessment after a tabletop brainstorming learning activity. Open Learner Models (OLMs) were core to the reflection task, to scaffold student's self-assessment of *egalitarian contribution*; and group interaction from *ideas sparked* from each other. We present multiple OLMs to the group generated from logs automatically captured from the collaborative activity. Our work advances the understanding of OLMs for brainstorm reflection, and the benefit of multiple OLM representations.

Keywords: OLMs, Visualisations, Brainstorming, F2F Collaboration.

1 Overview

Analyzing alternative views of captured student data can be used to provide effective support to both students and teachers [2]. This is particularly crucial for developing collaborative skills for idea generation, and for students to reflect on how well they contributed to the group [3] and their interaction with others. Reflection involves actively monitoring, evaluating and modifying one's thinking and comparing it to peers. Reflection-on-action is when one evaluates their own process, "thinking *back on what [they] have done in order to discover [how] knowing-in-action [their actions] may have contributed to an unexpected outcome"* [4]. Open Learner Models (OLMs) have long been used as a method to support student reflection on their development of knowledge, skills, performance and understanding [1]. We support reflection with OLM visualisations immediately after the brainstorm (Figure 1—top).

We created models and their visualisations for two key aspects of group brainstorming: *contribution equality* in terms of the number of ideas created by each student, and group effect in terms of the number of *ideas sparked*. We scaffold each area differently. For *contribution equality,* we analyse the effect of two group OLMs on students' self-awareness, by presenting them in sequence, incrementing the detail of the student information shown (Figure 1—2,3). For *idea sparking*, we compare the inspection of the final product to a replay of the whole brainstorm process, and a hands-on reflection task with the presentation of a group summary OLM to students (Figure 1—5,6). We analyse the effect of the scaffolded reflection activity by measuring changes in self-awareness, from Likert data and students' written responses, after presenting each new piece of information. We examine whether *students gain greater insights from studying each of the different OLMs.*

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 615–616, 2014.

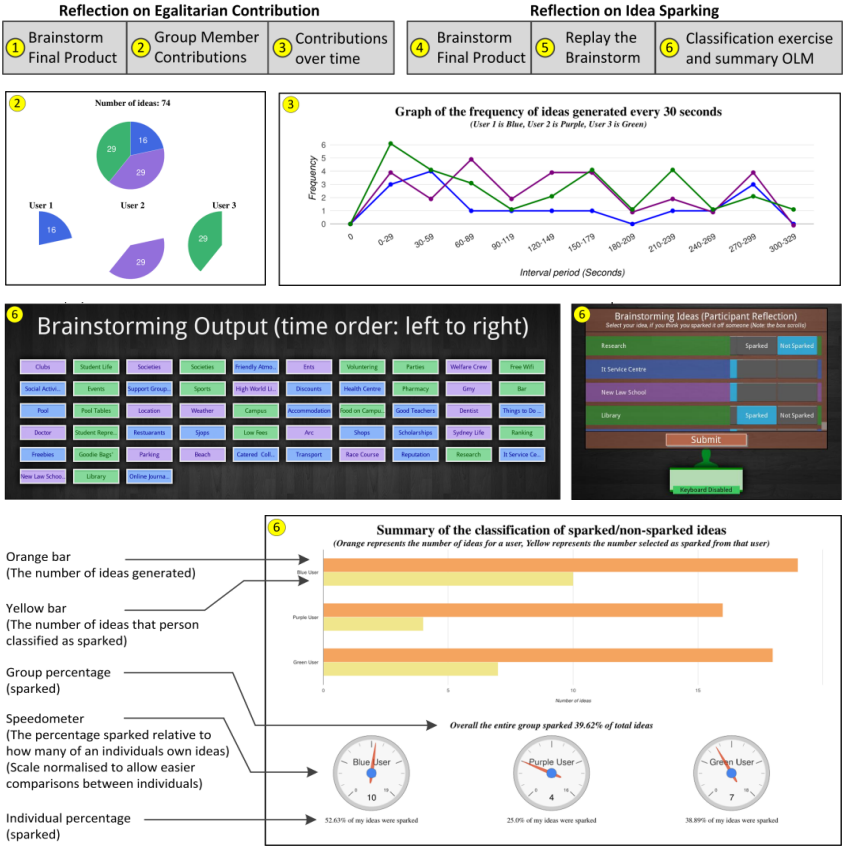


Fig. 1. Egalitarian Contribution and Idea Sparking OLMs

Our set of carefully designed OLMs, offered students the benefit of reflection on what they did, how they did it, and what they learnt. Our work enabled learners to step back and critically reflect on their actions. Multiple representations for both egalitarian participation and idea sparking led to insights for the majority of students. This work moves towards demonstrating OLM effectiveness for gaining insights into the collaborative process. Moving forward, we will examine the integration of these OLMs into an authentic classroom setting and explore their long-term use over multiple brainstorms.

References

1. Bull, S., Kay, J.: Student Models that Invite the Learner. The SMILI Open Learner Modelling Framework, IJAIED 17(2), 89–120 (2007)
2. Martinez Maldonado, R., Kay, J., Yacef, K., Schwendimann, B.: An Interactive Teacher's Dashboard for Monitoring Groups in a Multi-tabletop Learning Environment. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 482–492. Springer, Heidelberg (2012)
3. Osborn, A.F.: Applied Imagination. Charles Scribener's Sons, New York (1953)
4. Schon, D.: The reflective practioner. Temple Smith, London (1983)

Question Asking During Collaborative Problem Solving in an Online Game Environment

Haiying Li[1], Ying Duan[1], Danielle N. Clewley[1], Brent Morgan[1], Arthur C. Graesser[1], David Williamson Shaffer[2], and Jenny Saucerman[2]

[1] University of Memphis, Institute for Intelligent Systems, Memphis, USA
{hli5,yduan1,dnclwley,brent.morgan,graesser}@memphis.edu
[2] University of Wisconsin-Madison, Departments of Educational Psychology and Curriculum and Instruction, Madison, USA
{dws,jsaucerman}@education.wisc.edu

Abstract. This paper investigated frequency of questions and depth of questions in terms of both task difficulty and game phase when players collaboratively solve problems in an online game environment, Land Science. The results showed frequency of questions increased with both the task difficulty and unfamiliar tasks in the game phases. We also found players asked much more shallow questions than intermediate and deep questions, but more deep questions than intermediate questions.

Keywords: question asking, collaborative problem solving, online game environment.

Question Asking. Questions that students ask reflect their specific knowledge deficits, uncertainty about information, and apparent contradictions [1]. Student question asking reveals active learning, construction of knowledge, curiosity and the extent of the depth of the learning process [2]. Previous research on question asking focused on the classroom [3] and one-on-one tutoring [4] environments. Student questions in the classroom were infrequent and unsophisticated as compared with one-on-one tutoring environments, because one-on-one tutoring environments could tailor activities to the student's knowledge deficit and removing social barriers [1]. Recently, multiparty educational games have allowed groups of students to interact with computer-mediated communication on tasks that require collaborative learning and problem solving [5]. However, there are few empirical studies on question asking in this multiparty environment. This study investigated the question asking during collaborative problem solving in an online game environment, Land Science.

Land Science is an interactive urban-planning simulation with collaborative problem solving in an online game environment [6]. Players are assigned an in-game internship in which they act as land planners in a virtual city with the guidance of a mentor. They communicate with others through text chats for inquiries.

This paper examines the frequency of questions as a function of the task difficulty, game phase, and question depth in Land Science. Three hypotheses are proposed: the frequency of questions increases as a function of increasing (1) task difficulty, (2) the task unfamiliarity, and (3) question depth.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 617–618, 2014.

Method. 100 middle and high school students participated in 7 Land Science games. Two student researchers manually identified 1,936 (13.32%) questions from students' chats, and then coded them into 18 question categories according to the *Graesser-Pearson Taxonomy* [7], and the Other category (the average Kappas above .76). Then the questions were scaled into shallow, intermediate, versus deep level (see 7 for detail). The 14 stages of the game were scaled into easy, medium and difficult by a member of the Land Science development team based on the task familiarity and complexity. In addition, four phases were coded as introduction, new task, repeated task and closing.

Results and Discussion. Relative frequency of questions was operationally defined in the unit of per 100 words. Jonckheere-Terpstra trend tests were performed on 3 task difficult levels and 4 game phases separately. Results showed that the frequency of questions increased with task difficulty (p=.023), and with task unfamiliarity (p=.071). A nonparametric Kendall's tau-b test confirmed the trend (r=.458) in task difficulty and task unfamiliarity (r=.331). Therefore, players did ask more questions as task difficulty and task unfamiliarity increased. General Linear Model showed there was a significant effect for depth of question, $F(2,37)=401.27$, $p<.001$, $\eta^2=.956$. Post-hoc Bonferroni tests indicated that shallow questions (M=.80, SD=.097) were significantly more than deep (M=.15, SD=.078) and intermediate (M=.05, SD=.032) questions, and deep questions were significantly more than intermediate questions.

These findings confirmed that question asking during collaborative problem solving in multiparty educational game environment was similar to classroom environment: players asked more shallow questions [5]. Therefore, the mentor should demonstrate how to ask deep question in order to facilitate deep learning.

Acknowledgement. This work was supported by the National Science Foundation (0918409) for the project of AutoMentor: Virtual mentoring and assessment in computer games for STEM learning. Any opinions are those of the authors.

References

1. Otero, J., Graesser, A.C.: PREG: Elements of a Model of Question Asking. Cognition & Instruction 19, 143–175 (2001)
2. Graesser, A.C., Ozuru, Y., Sullins, J.: What Is a Good Question? In: McKeown, M.G., Kucan, L. (eds.) Threads of Coherence in Research on the Development of Reading Ability, pp. 112–141. Guilford, New York (2009)
3. Dillon, J.: Questioning and Teaching: A Manual Practice. Teachers College Press, New York (1988)
4. Graesser, A.C., Person, N.K.: Question asking during tutoring. American Educational Research Journal 31, 104–137 (1994)
5. Kumar, R., Rosé, C.P.: Architecture for Building Conversational Agents that Support Collaborative Learning. IEEE Transactions on Learning Technologies 4(1), 21–34 (2011)
6. Shaffer, D.W., Gee, J.P.: Epistemic Games as Education for Innovation. BJEP Monograph Series II, Number 5-Learning through Digital Technologies 1(1), 71–82 (2007)

Aligning Ontologies to Bring Semantics to Learning Object Search

João Carlos Gluz[1], Luis Rodrigo Jardim Da Silva[1], and Rosa Vicari[2]

[1] Post-Graduation Program in Applied Computer Science (PIPCA) – UNISINOS – Brazil
jcgluz@unisinos.br, rodjle@gmail.com
[2] Interdisciplinary Center for Educational Technologies (CINTED) – UFRGS – Brazil
rosa@inf.ufrgs.br

Abstract. Within the educational context, researchers have focused on applying agent and ontology-based technologies to improve the processes of localization, retrieval, cataloging, and reuse of learning objects. This scenario highlights semantic heterogeneity issues, creating an excellent opportunity to evaluate, and explore ontology alignment techniques able to provide semantic integration between different ontologies. This work presents the *MSSearch* service, which combines state of the art agent and ontology-based technologies, with advanced alignment techniques to provide a semantic search service for a learning object repository. *MSSearch* was tested with a base of more than 11.000 learning object, answering queries in real-time. The quality of the answers were checked by educational experts and considered very satisfactory, when compared against similar queries made with the standard search engine of a public repository of learning objects, containing a similar set of learning objects.

Keywords: Ontology Alignment, Learning Objects, Multiagent Systems, Metadata.

1 Introduction

The *MSSearch* system presented in this work uses advanced ontology alignment techniques to create a semantic search engine, and a native OWL Learning Objects (LO) repository. The OBAA metadata ontology [3] was chosen to represent, and store LO metadata. The most important problem addressed by *MSSearch* is how to correlate LO metadata stored in the repository to educational ontologies, which represent, for instance, the learning domains, teaching strategies, and other educational topics. The establishment of relations among metadata and educational ontologies, or among distinct, but generally heterogeneous educational ontologies could be very complex. Fortunately, there are some techniques that can make this process easier, allowing the automatic, or semi-automatic establishment of the relations among the ontologies. Ontology alignment [1,2] is currently regarded as an important mechanism for the integration of semantically heterogeneous databases, and as an enabling technology to provide semantic searches on these databases.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 619–620, 2014.

2 The MSSearch System

The architecture of MSSearch system was designed according to the guidelines, and principles presented in [4]. The ontology layer is formed by a set of educational ontologies aligned to the metadata ontology. The interface layer contains the web interface with common users (*WebQueryInterface*) and administrators (*WebAdminInterface*), the web services interface (*RESTfulInterface*), and the interface with LO repositories through the OAI-PMH harvesting protocol (*OAIPMHInterface*).

The agents layer is composed by the following agents: *MetaQuery*: agent responsible for executing the queries in semantic repository; *MetaUpdate*: agent that updates metadata stored in the repository; *MetaLoad*: agent, which is charged with the task of to populate the database with learning object metadata; *OntoAlign*: agent that perform the alignment of ontologies; *SemanticSearch*: agent that implements the semantic search mechanism. This agent also implements the relevancy-based ordering of query results; *RDFBaseManager*: this agent encapsulates the storage facility of native RDF triples storage, which currently is the graph storage system provided by JENA TDB; *OWLReasoner*: agent that encapsulates the OWL inference engine used in MSSearch. Currently this agent is integrated with the Pellet reasoner. Agents *MetaQuery*, *MetaUpdate*, *MetaLoad*, *RDFBaseManager*, and *OWLReasoner* form the core subsystem of *MSSearch*, which combines the JENA TDB RDF database, with the Pellet reasoner, to provide a native OWL repository able to store, locate, and retrieve LO metadata. The remaining agents implement the semantic search, and alignment functionality.

A performance evaluation experiment was conducted to measure the execution time of operations to load, and query. The load operation was tested from 99 till 11088 LO, which were obtained from BIOE repository (*http://objetoseducacionais2.mec.gov.br/?locale=en*). Based on test results, it was possible to estimate that the load time remained linearly proportional to the number of objects, indicating a possible maximum complexity of order $O(n)$. In another test a complex SPARQL query was performed aiming to recover all LO in the semantic repository, ordered by their title. According to the test results, the performance of query operation appears to be logarithmically proportional when the number of LO stored in the repository ranges from 99 to 4200, passing to a more linear performance after 4200. A user perception experiment was conducted with four teachers. The results show that, from the point of view of its users, *MSSearch* consistently returned best query results than BIOE.

References

1. Ehrig, M.: Ontology alignment: bridging the semantic gap. Springer (2007)
2. Euzenat, J., Shvaiko, P.: Ontology matching. Springer (2007)
3. Gluz, J.C., Vicari, R.M.: An OWL Ontology for IEEE-LOM and OBAA Metadata. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 691–693. Springer, Heidelberg (2012)
4. Vicari, R.M., Gluz, J.C.: An Intelligent Tutoring System (ITS) View on AOSE. Int. J. of Agent-Oriented Software Engineering 1, 295–333 (2007)

Social Network Signatures of Effective Online Communication

Xiaoxi Xu, Tom Murray, Beverly Park Woolf, and David A. Smith

University of Massachusetts, Amherst, MA

Abstract. In this paper, we study effective communication skills by analyzing the structure properties (e.g., degree, hub) of participants' interactions in an online classroom discussion context. We perform a regularized canonical correlation analysis to explore the social network signatures of effective communication skills (e.g., *perspective taking*). Experiments on computer-mediated communication among college students have shown that a statistically significant correlation exists between effective communication skills and social network profiles, measured on the same participant, with an effect size of 0.81. We discover that people showing more *perspective taking* behaviors are more popular and influential than others in their communication network. Such people also tend to reach out to people who behave similarly, which implies a like-attracts-like social phenomenon that complies with the Law of Attraction.

In this paper, we address an important and yet unexplored research question about effective communication: *what are the network signatures of effective communication skills?* We create a regularized canonical correlation analysis model to study the associations between an array of ten effective communication skills and a group of 17 social network metrics, all measured on the same participant. We answer intriguing questions, such as, are people who show perspective taking behaviors more popular than others in their communication network? This research is a part of a larger research endeavor to understand an emerging social communication phenomenon in online interactions, which we call *communication intelligence.* The constructs of communication intelligence, or *intelligence-embodied communication skills*, are the ten effective communication skills that we study. These skills include connection, proof, restraint, agreement, appreciation, self-reflection, perspective taking, monitoring, balance, and plan. The 17 social network measures used in this study are: in degree, out degree, degree, weighted in degree, weighted out degree, weighed degree, eccentricity, closeness centrality, betweenness centrality, authority, hub, modularity class, page rank, component ID, strongly connected ID, clustering coefficient, and eigenvector centrality.

Experimental Data: Our experiment data were from computer-mediated communication among college students. We had a total of 44 students with females (55%) and males (45%) relatively evenly distributed. Most of the students were juniors (34%) and seniors (45%). Students from two disciplines (i.e., pre-law and communication studies) discussed ill-defined topics, such as "right to die," and

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 621–622, 2014.

"internet free speech." They were randomly assigned to small groups of 7 or 8 people with the goal of encouraging more focused discussion. In our experiments, we collected two set of quantitative measures: (1) *scores of the use of intelligence-embodied skills* and (2) *measures of social network profiles*. For each student, the scores associated with each intelligence-embodied skill used in the overall discussion were computed by averaging over the number of posts associated with that student. The social network profiles were generated through Gephi and then were normalized by group size.

Research Method – Regularized Canonical Correlation Analysis: Canonical correlation analysis (CCA) is a method for exploring the relationship between two sets of variables, all measured on the same experimental unit. CCA is not only a regression method, but also a dimension reduction method, in that *it determines the relationship between two sets of variables and computes how many dimensions are necessary to understand the association between these two sets of variables*. Regularized canonical correlation analysis (RCCA) imposes a ridge penalty on CCA to address the issue that multicolliearity is present within either or both sets of variables, or the number of experimental units is less than the number of measuring variables. In this research, we use RCCA to identify associative patterns between participants' use of intelligence-embodied skills and their network metrics, because intelligence-embodied communication skills appear to be interrelated.

Experimental Results: With regularized canonical correlation analysis, ($\lambda_1 = 0.0001$ for communication skill variables and $\lambda_2 = 0.00001$ for network metric variables), we found one statistically significant (α=0.1) canonical dimension. This significant canonical dimension has a canonical correlation 0.90 (p-value=0.08) with a large effect size of 0.81. For the communication skill variables, the first canonical dimension is most strongly influenced by *perspective taking* (0.63). This result provides an exciting way to study *perspective taking* through the lens of social network metrics, as shown below.

- *Popular* – people showing more *perspective taking* behaviors are more popular (i.e., positive correlations with hub, degree) than others in the communication network.
- *Influential* – people showing more *perspective taking* behaviors are more influential (i.e., a positive correlation with authority). Their neighborhoods do not interact much themselves (i.e., a negative correlation with clustering coefficient). They contribute to a large local community (i.e., a positive correlation with eccentricity) that has more communication (i.e., a correlation with strongly connected).
- *Like-attracts-like* – people showing more *perspective taking* behaviors are more likely to communicate with people who behave similarly. This is based on a correlation found between perspective taking and network component – people tend to communicate with others who demonstrate similar level of perspective taking. In other words, their communication network demonstrates propinquity.

Future Work: For further validation of our results, we will replicate the above experiments with a larger sample of populations and possibly from diverse cultures.

A Multi-level Complex Adaptive System Approach for Modeling of Schools

Ted Carmichael[1], Mirsad Hadzikadic[2], Mary Jean Blink[1], and John C. Stamper[1,3]

[1] TutorGen, Inc., Wexford, PA, USA
{tcarmichael,mjblink}@tutorgen.com
[2] University of North Carolina at Charlotte, Charlotte, NC, USA
mirsad@uncc.edu
[3] Carnegie Mellon University, Pittsburgh, PA, USA
jstamper@cs.cmu.edu

Abstract. The amount of data available to build simulation models of schools is immense, but using these data effectively is difficult. Traditional methods of computer modeling of educational systems often either lack transparency in their implementation, are complex, and often do not natively simulate non-linear systems. In response, we advocate a Complex Adaptive Systems approach towards modeling and data mining. By simulating agent-level attributes rather than system-level attributes, the modeling is inherently transparent, easily adjustable, and facilitates analysis of the system due to the analogous nature of the simulated agents to real-world entities. We explore the design a CAS model of schools using multiple levels of data from varied data streams.

Keywords: Complex Adaptive Systems, Agents, Educational Data Mining.

1 Multi-level CAS Design of an Educational System

As schools become increasingly wired, the ability to collect data at multiple levels has grown exponentially to the point of becoming overwhelming. We classify the multiple data streams into four levels: Individual, Classroom, School, and District. This work is centered on finding the complementary links between these levels and using them together to bring a much clearer picture of the overall educational system.

At the highest levels, most of the academic work in the fields of learning analytics, educational data mining, and intelligent tutoring systems focus specifically at the classroom level or the individual student level using data from learning management systems or finer grain data from logs created from educational technologies[3]. Some work has brought together log data and correlated it with student grades, but little has been done to harness all of these data streams into a robust model. We propose a CAS (Complex Adaptive System) model to do this, for two reasons: the inherent transparency of using agent-based analogues, and the ease with which a CAS model can represent non-linearities. CAS is a method developed in physics, mathematics, and other sciences [1,2] to deal with the issues of complexity, and has been redefined by a growing number of applications in many domains. The most striking feature of a CAS is that even simple agents – with only a few attributes and rules– can produce complex, dynamic behavior at many different scales of interest.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 623–624, 2014.

Agent-Based Modeling (ABM) is a technique for creating a computer-based simulation of a CAS. Crucially, ABM relies on modeling *agent*-level behavior, rather than *system*-level behavior. The agents of such a system can represent schools, classrooms, or even individual students and teachers. The agents, then, are analogues for real-world entities, and are thus endowed with the same properties as their physical counterparts. In this way, non-linear behavior can emerge from the simulated system in the same way that it emerges in the classroom, school, or school district. Further, an ABM is inherently transparent, as the simulated agents have properties that are directly analogous to those of the real agents in the system of interest. For these reasons we believe ABM is a fruitful method for simulating all the complex interactions and non-linearities found in a school system.

Educational systems currently collect many characteristic-, performance- and outcome-level data, including grades, test results, economic status, gender, age, race, etc. However, such data, while useful, still leave many aspects of classroom performance unreported. For example, none of them include the nature and frequency of interactions among students, teachers and students, students and principals, teachers and principals, or principals and superintendents. In addition, there are no correlations between the availability of resources, the nature of such interactions, and the overall performance of students and schools/school districts. CAS methodology can offer a way to simulate and model such interactions at multiple levels, including classrooms, schools, districts, and states. Due to the interactive nature of the classroom there is a great potential for threshold "tipping point" effects to exist, and it is intuitively true that some students or student clusters can have an outsize effect on the rest of the class. One of the goals of this research will be to discover and understand the underlying dynamics of such threshold effects, within the classroom, the school, and the district-wide school system, so that a smarter approach in resource allocation can produce a more effective educational system. This work identifies the links between multiple streams of data and the development of CAS model to represent an entire school ecosystem, from the individual student to the district level. This model allows for predictive analysis at each level by simulating interactions at the other levels. The end result of this effort produces a robust model of an educational system at multiple scales, one that can not only help determine the causal factors of desirable outcomes, but also allow for multiple "what if" scenarios to be run in simulation, so that these outcomes can be improved and resources are expended in the most efficient manner.

References

1. Carmichael, T., Hadzikadic, M., Dréau, D., Whitmeyer, J.: Towards a General Tool for Studying Threshold Effects Across Diverse Domains. In: Ras, Z.W., Ribarsky, W. (eds.) Advances in Information and Intelligent Systems. SCI, vol. 251, pp. 41–62. Springer, Heidelberg (2009)
2. Gell-Mann, M.: Complex Adaptive Systems, pp. 17–45. Addison-Wesley (1994)
3. Stamper, J., Carmichael, T.: A Complex Adaptive System Approach to Predictive Data Insertion for Missing Student Data. In: Proceedings of the 3rd Int. Conference on Computer Blended Learning (ICBL 2007), Florinopolis, Brazil. Kassel Press (May 2007)

Assessing Science Inquiry Skills Using Trialogues

Diego Zapata-Rivera, Tanner Jackson, Lei Liu, Maria Bertling, Margaret Vezzu, and Irvin R. Katz

Educational Testing Service, Princeton, NJ 08541 USA
{dzapata,gtjackson,lliu001,mbertling,mvezzu,ikatz}@ets.org

Abstract. Trialogue-based tasks can be used to gather evidence that may be difficult to obtain using traditional assessment approaches, such as embedded questions. However, more research needs to be done in order to create valid, fair, and reliable conversation tasks that can be used for assessment purposes. This paper describes ongoing efforts at developing and evaluating trialogues for assessing students' science inquiry skills.

Keywords: Adaptive technologies, conversation systems, assessment.

1 Introduction

Natural language conversations between students and pedagogical agents (e.g., a virtual peer or teacher) have been used successfully as part of intelligent tutoring systems and formative assessment systems. A trialogue is a particular type of conversational task in which there are typically two virtual agents (e.g., a tutor and a peer) and one human student. Compared to traditional testing formats, trialogues may elicit more evidence of students' skills and their conceptual knowledge, and may allow for interactive, adaptive assessments [1]. This paper describes aspects of our development process and presents current work implementing and evaluating trialogues for assessing science inquiry skills.

2 Trialogue-Based Tasks

The current work on conversation-based assessments leverages many advances made through previous research on natural language intelligent tutoring systems. More specifically, these trialogues are based on the research and architecture of AutoTutor [2].

The development process of these trialogue-based tasks also follows the principles of Evidence-Centered Design [ECD; 3]. This iterative process starts from a clear definition of the constructs, followed by the scene design process that involves authoring, implementing, and testing of scripts. Characters and other graphical components are designed, storyboards are produced and revised, and a trialogue-based task prototype is developed and evaluated with the intended audience.

Conversation diagrams have been designed to facilitate authoring of these tasks. These diagrams serve as communication tools to facilitate communication about task design among an interdisciplinary group of experts that includes assessment developers, dialogue engineers, and scientists. Utterances including sample responses are connected, forming conversation paths. These paths may involve several turns between the student and the virtual agents depending on the student's input. Several

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 625–626, 2014.

types of responses are usually handled including: correct response, partially correct response, irrelevant response, metacognitive and metacommunicative questions, and silence. The scoring process has two components: (1) path-based scoring and (2) revised scores based on additional evidence (from human or automatic).

3 The Volcano Scenario

The current trialogue scenario introduces students to factors related to volcanic eruptions and allows them opportunities to converse with virtual agents, place seismometers to collect data, analyze data, take notes, and make data-based predictions. These activities were designed to evaluate students' science inquiry skills such as data collection and evidence-based reasoning. Several trialogue-based tasks gather information about decisions students make during data collection, conceptual misconceptions, and alert level predictions based on data collected. For example, after making notes about the data collected by seismometers, the student interacts with two virtual agents (Dr. Garcia and a student agent named Art) to review and compare one of his/her own notes with one of Art's notes.

4 Preliminary Results

A small study with 10 students (50% female) in 6th to 8th grade was conducted using the volcano scenario that included 2 trialogue-based tasks. Participants completed a background questionnaire, interacted with the volcano scenario, and completed usability and engagement questionnaires. Each session lasted approximately 90 minutes. Results from the usability and engagement questionnaire showed that, in general, students enjoyed the activity. Participants were able to complete the activity with minimal instruction. Students generated mindful conversations and reflected various levels of the target constructs.

5 Future Work

New trialogue-tasks targeting a variety of constructs are being developed, as well as tools to facilitate the development of these tasks. Existing trialogues are being refined based on new data. Future work also includes conducting large-scale validity studies, as well as improving testing and scoring approaches and investigating the psychometric properties of these tasks.

References

1. Zapata-Rivera, D.: Exploring the use of Trialogues in Assessment. Cognition and Assessment SIG Symposium. Annual meeting of the American Educational Research Association (AERA), San Francisco (2013)
2. Graesser, A.C., Person, N.K., Harter, D.: The Tutoring Research Group: Teaching tactics and dialogue in AutoTutor. Int. J. of Artificial Intelligence in Education 12, 257–279 (2001)
3. Mislevy, R.J., Steinberg, L.S., Almond, R.G.: On the structure of educational assessments. Measurement: Interdisciplinary Research and Perspectives 1, 3–62 (2003)

Attitudinal Gains from Engagement with Metacognitive Tutors in an Exploratory Learning Environment

David A. Joyner and Ashok K. Goel

Design & Intelligence Laboratory, School of Interactive Computing
Georgia Institute of Technology, Atlanta, Georgia, USA
david.joyner@gatech.edu, ashok.goel@cc.gatech.edu

Abstract. MILA–T (MILA–Tutoring) is constructed to give students explicit instruction on scientific modeling and inquiry, intending in part to help cultivate positive attitudes toward science. The results of a two-week controlled experiment using MILA–T in middle school classroom show a significant effect of MILA–T on students' attitudes towards science.

Keywords: Attitudes toward science; metacognitive tutoring; middle school education; scientific inquiry; scientific modeling.

1 Introduction

Middle school science education carries metacognitive learning goals: students learn about scientific inquiry and modeling to reflect on and regulate their knowledge of science [6]. Attitudinal learning goals on curiosity, skepticism, and positive argumentation are addressed [2]. Here, we examine whether metacognitive tutoring helps improve middle school students' attitudes toward science, scientific inquiry, and careers in science. Our hypothesis is that metacognitive tutors improve students' attitudes towards these topics. We present a controlled experiment with an exploratory learning environment called MILA (**M**odeling & **I**nquiry **L**earning **A**pplication, [3]) that evolves from the ACT system [4], and a metacognitive tutoring extension called MILA–T (MILA-**T**utoring), in which access to MILA–T was varied. Attitudinal surveys were given before and after engagement, and we report changes to these scores.

2 Experimental Design and Results

Students participated in the intervention for approximately 50 minutes per day for nine days. Students completed attitudinal surveys on the first and last days and participated in a seven-day curriculum in between. The survey measured five constructs: Attitude Toward Scientific Inquiry [1], Career Interest in Science [1], Anxiety toward Science [5], Perception of the Science Teacher [5], and Desire to Do Science [5]. This study is a controlled experiment. In the control condition, students received MILA without MILA–T enabled during the seven-day curriculum. In the experimental condition, MILA–T is available to the students, providing individualized, situated

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 627–628, 2014.

feedback on the model construction process. 237 students participated in the intervention, with 99 in the control condition and 138 in the experimental.

The primary question of this research is whether changes in students' scores on these metrics changed over the course of the intervention based on exposure to MILA–T. To answer this question, we conducted a multivariate analysis of variance. First, we examined whether students were roughly equivalent on the given metric prior to the intervention. Then, we examined whether students' scores on that metric changed, and whether those changes were connected to the experimental condition.

Prior to the intervention, no significant relationship between attitude toward scientific inquiry and condition existed. Analysis of the overall change to attitude toward science inquiry revealed no significant change overall. However, breaking the groups down by condition revealed a statistically significant ($p < .05$) difference between the two groups. Students in the experimental group experienced an average increase of 1.46 points on their attitude toward scientific inquiry score ($\sigma = 7.16$). Students in the control group, on the other hand, experienced an average *decrease* of 1.16 points on their attitude toward scientific inquiry score ($\sigma = 6.33$). Thus, students interacting with MILA–T experienced a statistically significant increase in their attitudes toward scientific inquiry compared to students without MILA–T. Students in the experimental condition concluded the study with a higher ($p < .05$) attitude toward scientific inquiry ($\mu = 22.14$, $\sigma = 7.73$) than those in the control condition ($\mu = 20.02$, $\sigma = 7.44$).

Prior to the intervention, no statistically significant relationship existed between career interest in science and condition. Analysis of changes within groups revealed that students in the experimental group experienced a statistically significant increase in their career interest in science ($p < .05$) of 2.03 points ($\sigma = 6.01$). Students in the control group, on the other hand, no significant increase. These results indicate that participation with MILA–T led to an increase in career interest in science, while participation without MILA–T did not.

References

1. Fraser, B.J.: Test of science related attitudes. Australian Council for Educational Research. The Australian Council for Educational Research Limited, Hawthorn (1981)
2. Georgia Department of Education. Seventh Grade Science Curriculum (2006), retrieved from `https://www.georgiastandards.org` (retrieved January 19, 2014)
3. Joyner, D.A., Majerich, D.M., Goel, A.K.: Facilitating Authentic Reasoning About Complex Systems in Middle School Science Education. In: Proc. of the 11th Conference on Systems Engineering Research, Atlanta, GA (2013)
4. Vattam, S., Goel, A., Rugaber, S., Hmelo-Silver, C., Jordan, R., Gray, S., Sinha, S.: Understanding Complex Natural Systems by Articulating Structure-Behavior-Function Models. Educational Technology and Society 14(1), 66–81 (2011)
5. Weinburgh, M.E., Steele, D.: The modified attitudes toward science inventory: Developing an instrument to be used with fifth grade urban students. Journal of Women and Minorities in Science and Engineering 6, 87–94 (2000)
6. White, B., Frederiksen, J.: Inquiry, modeling, and metacognition: Making science accessible to all students. Cognition and Instruction 16(1), 3–118 (1998)

Understanding Students' Emotions during Interactions with Agent-Based Learning Environments: A Selective Review

Jason M. Harley[1] and Roger Azevedo[2]

[1] McGill University, Montreal, Quebec
jason.harley@mail.mcgill.ca
[2] North Carolina State University, Raleigh, North Carolina
razeved@ncsu.edu

Abstract. This selective review discusses the emotions that learners report experiencing while interacting with agent-based learning environments.

Keywords: Emotions, affect, pedagogical agents, intelligent tutoring systems.

1 Toward a Deeper Understanding of Emotions and ABLEs

How do students feel about interacting with specific types of computer-based learning environments (CBLEs)? Does the incidence of discrete emotions vary between similar types of these environments? What features support or hinder learners' experience of different emotions? This selective review addresses these questions as they relate to a type of CBLE: agent-based learning environments (ABLEs). ABLEs are unique from other CBLEs because of their use of pedagogical agents (PAs). PAs are animated characters designed to provide several functions such as immediate and tailored prompts and feedback to support student learning [1-7]. The primary objective of this review is to compare the emotions elicited by six different ABLEs. This selective review differs from other reviews, in several ways: (1) by focusing only on CBLEs with PAs; (2) examining any study that measured emotions using one or more methods so long as they met the criteria; (3) comparing and contrasting learners' incidence of each of the discrete emotions reported for all six of the ABLEs. Seven studies were selected on the basis of the following inclusion and exclusion criteria: (1) studies had to measure more than one discrete emotional state using a forced-choice measure[1]; (2) they had to report the incidence of emotions as either proportions or frequencies; and (3) in the case of multiple published articles based on the same or part of a common data set, the study with the larger sample size was taken.

Table 1 was created to eliminate the redundancy of the large number of emotional labels used by the seven studies by organizing them into sets that could: (1) be operationalized as different emotional states and (2) that reduced the number of emotional labels, but maintained as much meaningful variation in learners' emotions as possible. This synthesis was guided by research and operationalization of emotions by Pekrun

[1] Emotions in Table 1 could add up to more than 100% if they possessed different object-focuses (e.g., PA [admiration/reproach] vs. event outcome [joy/distress]) [1].

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 629–631, 2014.

[8] and D'Mello, Graesser, and colleagues [2]. Emotions were therefore associated within the dimensions of valence and activation. Positively-valenced, activating emotions that were specifically related to learning and characterized as cognitive-affective states (e.g., curiosity, engagement) were grouped together because they represent ideal emotional states where the learner is not just feeling 'good and energized' (e.g., happy), but in an emotional state where they are prepared to learn effectively.

2 Results

Table 1. Proportions of grouped discrete emotions experienced with ABLEs

			ABLE						
			AutoTutor [2]	Operation ARIES! [3]	Crystal Island [6]	[7]	MetaTutor [5]	Prime Climb [1]	Wayang Outpost [4]
Val.	Act.	Emotion %							
+	Act.	Happy/Joy/ Delight /Excitement	.06	.02	.25	.14	.09	.92**	.34
+	Act.	Eng./Flow / Focus/ Curiosity	.24	.24	.42	.41			
+	Act.[2]	Admiration						.82*	
+	De-Act.	Concentrated/ Satisfied							.58
-	Act.	Anger/ Frustration	.13	.06	.07	.16	.03		.06
-	Act.	Fear/ Anxiety/ Distress		.01	.09	.05	.00	.08**	
-	Act.	Disgust/ Contempt/ Reproach					.00	.18*	
-	De-Act.	Boredom/ Tired	.18	.33	.03	.09			.02
-	De-Act.	Sadness			.02		.03		
+/-	Act.[2]	Confusion	.17	.09	.13	.16			
+/-	Act.	Surprise	.03	.01			.03		
NA	Baseline	Neutral	.19	.26			.77		
+	Act.	-	.30	.26	.67	.55	.09	.92/.82	.34
+	De-Act.	-							.58
-	Act.	-	.13	.07	.16	.21	.03	.08/.18	.06
-	De-Act.	-	.18	.33	.05	.09	.03	-	.02
+/-	Act.?	-	.20	.10	.13	.16	-	-	
NA	Baseline	-	.19	.26	-	-	.77	-	

3 Discussion

A number of preliminary conclusions can be drawn from this review: First, game-like elements, when implemented in a sufficient quantity (e.g., more than a narrative context) and with sufficient quality to make the environment truly game-like are related to learners' experience of positive, activating emotions [1, 6, 7]. Similarly, the relevance of content to students' academics and the affordance of choice in an ABLE is also related to learners' experience of positive emotions [1, 4, 6, 7]. This review

illustrates that there is a range in the incidence of desired (positively-valenced, activating) emotions that learners experience while interacting with ABLEs. Few negatively-valenced, activating emotions are elicited, however, which is good news. Instead, the greatest challenge for researchers to target in emotional interventions is boredom. Neutral was found to be one of the most commonly appearing states in those environments that measured it [2, 3, 5]. Future research should include neutral because it is important to capture the range of students' emotional states, including those that may be considered to be a non or baseline emotional state. More studies with forced-choice emotional labels and their incidence are needed to validate and expand upon the number of ABLEs presently reviewed and the samples they drew upon.

Acknowledgements. This research was supported by a Joseph-Armand Bombardier Canada Graduate Scholarship for Doctoral research from the Social Science and Humanities Research Council (SSHRC) of Canada awarded to the first author.

References

1. Conati, C., Maclaren, H.: Empirically building and evaluating a probabilistic model of user affect. User Modeling and User-Adapted Interaction 19, 267–303 (2009)
2. D'Mello, S.K., Graesser, A.C.: AutoTutor and affective AutoTutor. ACM Transactions on Interactive Intelligent Systems 2 (2013)
3. D'Mello, S., Lehman, B., Pekrun, R., Graesser, A.: Confusion can be beneficial for learning. Learning and Instruction 29, 153–170 (2014)
4. Dragon, T., Arroyo, I., Woolf, B.P., Burleson, W., el Kaliouby, R., Eydgahi, H.: Viewing student affect and learning through classroom observation and physical sensors. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 29–39. Springer, Heidelberg (2008)
5. Harley, J.M., Bouchet, F., Azevedo, R.: Aligning and comparing data on learners' emotions experienced with MetaTutor. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 61–70. Springer, Heidelberg (2013)
6. McQuiggan, S.W., Robison, J.L., Lester, J.C.: Affective transitions in narrative-centered learning environments. Journal of Ed. Technology & Society 13(1), 40–53 (2010)
7. Sabourin, J., Mott, B., Lester, J.C.: Modeling learner affect with theoretically grounded dynamic Bayesian networks. In: D'Mello, S., Graesser, A., Schuller, B., Martin, J.-C. (eds.) ACII 2011, Part I. LNCS, vol. 6974, pp. 286–295. Springer, Heidelberg (2011)
8. Pekrun, R.: Emotions as drivers of learning and cog. dev. In: Calvo, R.A., D'Mello, S. (eds.) New Perspectives on Affect and Learning Tech., pp. 23–39. Springer, NY (2011)

Authoring System to Design Pedagogical Devices: The SAPRISTI System

Dominique Leclet-Groux and Ismail Hassan Djilal

Laboratoire Modélisation, Information et Systèmes (MIS)
Université de Picardie Jules Verne
33 rue Saint-Leu – 80037 Amiens Cedex 1, France
dominique.leclet@u-picardie.fr

Abstract. This paper presents an Authoring System for the design of pedagogical devices dedicated to the teachers. This system helps a teacher to create his pedagogical scenario. It generates also the teacher's pedagogical device (based on Web 2.0 tools) and a dashboard, which allows controlling the students' activities.

Keywords: Authoring System, Instructional Design.

1 Research Context

The research context concerns the project-based pedagogy, which can be classified in the active pedagogy [1]. This pedagogy is supported by a pedagogical method named MAETIC, which is represented by a book [2]. This method targets students of professional trainings. It permits the acquisition of know-how corresponding to the educational objectives fixed by the teacher. For that, the students develop a "product" by the implementation of project management techniques.

MAETIC recommends that students follow a five steps cycle, to realize their activities. This five-step process represents the pedagogical scenario. On the students' side, every group of students holds a logbook to describe the progress of the project and realize deliverables. On the teacher's side, a pedagogical device (named *e-suitcase MAETIC*) allows to inform the students about the progress of the Teaching Unit. The teacher writes, on his e-suitcase, the students' activities. To control his teaching, he has also a dashboard, which allows controlling the students' activities.

To structure the design process of the *e-suitcases MAETIC*, we follow a design method created by D. Leclet-Groux [3] and called MAUI (French acronym, instructional design method based on cognition). This iterative method follows a succession of stages, by gradually refining the specifications. It is also an incremental method. The designer develops a core of the system and adds supplementary features. MAUI is located in the Instructional Design Domain and based on the ADDIE model [4]. Evaluations of the MAUI design method have been realized, on the ground [5].

2 The SAPRISTI System

MAETIC has been used in Maroco (Faculty of Science and Technology, University Sidi Mohamed Ben Abdellah, Fès) and in Djibouti (University of Djibouti). Initially,

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 632–633, 2014.

15 teachers were trained to the concept of collective project-based pedagogy and the use of MAETIC was explained. But, we found that this training was not sufficient because, we encountered the two following problems.

1. When teachers wish to apply MAETIC in their teachings, they have only the MAETIC book, which outlines the activities to be performed by students. They have problems in the scenario design and they don't have help with the design of their pedagogical scenarios (order of sessions, duration, skills preferred, …). To solve this problem, we aim to provide *assistance to the scenarios design*, through a computer tool. This assistance helps the definition of the teachers' pedagogical needs and customizes the MAETIC activities.
2. Teachers can have difficulties in supervising students, when they are novice at the use of the method. They do not have a screen, where all the students' activities are grouped to control the best of workflow. They don't have help to the establishment of their pedagogical device. To solve this problem, we propose to develop a tool that *automatically generates the E-Suitcase MAETIC and the students' logbooks*. This tool creates also a dashboard that allows seeing the students' activities. These activities are grouped on the one screen to control the students' workflow.

This tool calls the SAPRISTI System (French acronym: Système Auteur pour la concePtion et la généRation de dIspositifs pédagogiqueS support de maeTIc). It can be considered as a Authoring Tool [6]. It supports the MAUI design method. SAPRISTI is composed by two components: **1- the Assistance Component**. First, it collects information about the description of the Teaching Unit. Second, it generates the pedagogical scenario. For this, it relies on model of knowledge (activities and skills) and rules. **2- the Generator Component**, which generates the technological environment. These devices are represented in the form of weblogs.

References

1. Barr, R.B., Tagg, J.: From Teaching to Learning - A New Paradigm for Undergraduate Education, pp. 13–25 (November/December 1995)
2. Leclet, D., Talon, B.: La methode MAETIC. In: LV, p. 61 (2008) ISBN: 978-2-35209-161-5
3. Leclet, D.: Environnement interactifs d'apprentissage dans des contextes professionnels, des tuteurs intelligents aux systèmes supports d'apprentissage à distance, HDR de l'Université de Picardie Jules Verne, 227 p. (2004)
4. Strickland, A.W.: ADDIE. Idaho State University College of Education Science, Math. & Technology Education (retrieved June 29, 2006)
5. Leclet, D., Talon, B.: Assessment of a Method for Designing E-Learning Devices. In: Proceedings of World Conference on Educational Multimedia, ED-MEDIA 2008, June 30-July, pp. 1–8. AACE/ Springer, Vienna (2008)
6. Murray, T.: Authoring Intelligent Tutoring Systems: An analysis of the state of the art. International Journal of Artificial Intelligence in Education 10, 98–129 (1999)

Fostering Teacher-Student Interaction and Learner Autonomy by the I-TUTOR Maps

Vincenzo Cannella[1], Laura Fedeli[2], Arianna Pipitone[1], Roberto Pirrone[1], and Pier Giuseppe Rossi[2]

[1] Dep.t of Chemical, Management, Computer, Mechanical Eng., Univ. of Palermo
Viale delle Scienze, Bdg 6, 90128 Palermo, Italy
[2] Department of Education, Cultural Heritage and Tourism, University of Macerata
P.le Bertelli, 62100, Macerata, Italy
{vincenzo.cannella26,arianna.pipitone,roberto.pirrone}@unipa.it,
{pg.rossi,laura.fedeli}@unimc.it

Abstract. The paper analyses the use of an automatically generated map as a mediator; that map visually represents the study domain of a university course and fosters the co-activity between teachers and students. In our approach the role of the teacher is meant as a mediator between the student and knowledge. The mediation (and not the transmission) highlights a process in which theres no deterministic relation between teaching and learning. Learning is affected by the students previous experiences, their own modalities of acquisition and by the inputs coming from the environment. The learning path develops when the teachers and the students visions approach and, partly, overlap. In this case we have co-activity. The teacher uses artifacts-mediators in such a process (Bruner). The automatically generated map can be considered a mediator. The paper describes the experimentation of the artifact to check if its use fosters: (1) the elicitation of the different subjects perspectives (different students and the teachers), and (2) the structural coupling that is the creation of an empathic process between the perspectives of the teacher and the student as the way to enable co-activity processes between teaching and learning.

Keywords: Co-actvity, Structural Coupling, Mediation, Latent Semantic Analysis, Self Organizing Map, Zoomable User Interfaces.

1 Introduction

The artifact described in the paper was created within the project I-TUTOR (Intelligent Tutoring for Lifelong Learning - http://www.intelligent-tutor.eu/) approved by the European Community. The research group composed by researchers of the information science field and education field has been working for the last years in the development of AI enabled artifacts for e-learning [1][2][3]. The system aims at helping teachers and students foster a professional and enactive approach. In such a direction some plug-ins for the Moodle Learning Management System have been developed. The I-MAP plug-in is a concept map,

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 634–637, 2014.

that represents the course domain in terms of all its relevant topics as they're described by the teacher. It relies on the creation of two semantic spaces aimed at modeling both the course topics and the students interaction with the VLE: the *conceptual space* and the *activity space* (one for each student). Latent Semantic Analysis (LSA) has been adopted to compute the semantic space generated from a document corpus on the basis of the occurrence frequencies of a set of meaningful terms in each document. A self-Organizing Map (SOM) neural network is used for both learning the topology of data in the space itself and clustering input vectors. Finally, the map represents the 2D projection of the SOM lattice after the training as a grid of cells. The map has been developed as a Zooming User Interface representing graphically documents and topics. The reader is referred to [4] for a detailed description. In the experimentation the map was used to activate a co-activity/empathy [5]. Co-activity and empathy are meant as a progressive approach between the teachers perspective and the students one. At the beginning the savy knowledge (of the teacher) and knowledge that comes from the common sense (of the student) can show many elements of discontinuity. Thanks to the didactical transposition [6] and to the listening the teaching and learning process lets the actors reach, through continuous adaptations, a level of consistency between the two perspectives. The result of such a process is the structural coupling [7][8][9] In such a direction its necessary that the teacher can have multiple and flexible mediators, that is, artifacts and processes (active, iconic, analogic and simboli; [10]) that let the teacher represent reality. The path is organized in teaching and learning activities [11] that let the student experience in an active way the learning paths (open-ended activities). The use of the map is set in this direction: from the teachers viewpoint the maps nodes are the key words of the course and, then, the map represents the savy knowledge related to the course domain; since the maps nodes are not connected the student can freely build a net among the concepts selecting both the starting node and the path to be created.

2 Experimentation and Findings

The objective of the experimentation is aimed at verifying (1) if the teaching process activates an empathic attitude (co-activity) between the teacher and the student e (2) if the map can let students create a personalized path. The overall research design is framed within a qualitative approach implying a phenomenological method of inquiry. The experimentation involved a small sample of participants: one group of 10 students enrolled in the face-to-face graduate course in *General Didactics*, a second group composed by 5 students studying in e-learning modality in the same graduate course and a third group of 10 students who followed the course in the past years with a different syllabus.. The participants of each group were interviewed by the teacher through a semi-structured open conversation [12]. Students were asked to examine the map generated by I-MAP (that has just nodes and no hedges) and to interpret it describing in a narrative way the path that could connect all or a part of the map nodes according to their own logic. The task implied also to make it explicit the meaning

of the connection among the nodes and to discuss it with the teacher. Gathered data were: the net built by each student and the linguistic analysis of the description made by students themselves and focused also on the presence of the deixis phenomenon. The use of the I-MAP tool demonstrated (Table 1) its efficacy in group 1 and 2 both in the impact in the processes of co-activity teacher-students, since even if the maps created are different and with personalized elements, they showed to be consistent with the global approach of the course. Also the linguistic aspects highlight the students attitude towards the map according to a co-activity logic. Besides the use of the pronoun we" and the informal register preferred by students show the presence of a dialogic process between the teacher and the student. Groups 1 and 2 had a similar behavior compared to group 3 where a wider dispersion emerges in the occurrences of the node used. The map lets every student express his/her own perspective. As highlighted in Table 1 students created different paths starting from different nodes. There were two main options: some students (starting from the node action) developed a path more focussed on key concepts, others (starting from the node design) reported the teachers habitus from the design step to the assessment.

Table 1. Results of the experimentation

Node name	**Visited**			**Start**		
	Group 1	**Group 2**	**Group 3**	**Group 1**	**Group 2**	**Group 3**
Apprendimento	12%	7 %	13%	13%	0%	17%
Azione	18%	19%	9%	38%	55%	0%
Contesto	14%	8%	14%	13%	0%	17%
Costruttivismo	4%	5%	8%	0%	0%	17%
Dispositivo	14%	14%	7%	0%	0%	0%
Formazione	6%	8%	13%	0%	0%	0%
Progettazione	12%	13%	9%	38%	0%	33%
Progetto	6%	7%	10%	0%	45%	17%
Valutazione	12%	14%	14%	0%	0%	0%
std. dev.	0.05	0.05	0.03	0.16	0.22	0.12

In the future we plan to let the user have the chance to choose the starting node not only in the interpretation of the map, but also in its construction, enhancing the awareness of the importance of this choice and its effects.

References

1. Rossi, P.G., Carletti, S., Bonura, D.: A platform-independent tracking and monitoring toolkit. In: Proc. of the AAAI 2009 Fall Symposium in Cognitive and Metacognitive Educational Systems (FS-09-02), Arlington VA, USA, pp. 76–80. AAAI Press (2009)
2. Pirrone, R., Azevedo, R., Biswas, G.: Why metacognition in modern educational systems? In: AAAI Fall Symposium - Technical Report, vol. FS-09-02, pp. vii–viii (2009)

3. Bentivoglio, C.A., Bonura, D., Cannella, V., Carletti, S., Pipitone, A., Pirrone, R., Rossi, P.G., Russo, G.: Intelligent agents supporting user interactions within self regulated learning processes. Journal of e-Learning and Knowledge Society-English Version 6, 27–36 (2010)
4. Pipitone, A., Cannella, V., Pirrone, R.: Automatic concept maps generation in support of educational processes. Journal of e-Learning and Knowledge Society 10, 85–103 (2014)
5. Berthoz, A., Jorland, G. (eds.): L'Empathie. Odile Jacob, Paris (2004)
6. Chevallard, Y.: La transposition didactique: du savoir savant au savoir enseigné. La Pensée Sauvage, Grenoble, France (1991)
7. Maturana, H., Varela, F.: Autopoiesis: The organisation of the living. In: Maturana, H., Varela, F. (eds.) Autopoiesis and Cognition; The Realization of the Living, Reidel, Boston (1980)
8. Proulx, J., Simmt, E.: Enactivism in mathematics education: moving toward a re-conceptualization of learning and knowledge. Enactivism in Mathematics Education: Moving Toward a Re-conceptualization of Learning and Knowledge 4, 59–79 (2013)
9. Rossi, P.E.A.: Enactivism and didactics. some research lines. Education Science and Society 4, 37–57 (2013)
10. Bruner, J.: Toward a theory of instruction. Belkapp Press, Cambridge (1966)
11. Laurillard, D.: Teaching as a design science. Building pedagogical patterns for learning and teaching, New York (2012)
12. Corbetta, P.: La ricerca sociale: metodologia e tecniche. III Le tecniche qualitative, Il Mulino, Bologna, Italy (2003)

It Takes Two: Momentary Co-occurrence of Affective States during Computerized Learning

Nigel Bosch[1] and Sidney D'Mello[1,2]

[1] Departments of Computer Science, University of Notre Dame
[2] Psychology, University of Notre Dame
{pbosch1,sdmello}@nd.edu

Abstract. We investigated the incidence of momentary co-occurrence of affective states in a computerized learning environment. Novice students (N = 99) used a learning environment designed to teach the basics of computer programming. Only 46 of these students reported a sufficient number of co-occurring affective states for statistical modeling. Two co-occurring pairs of affective states occurred at rates higher than chance: Confusion/Uncertainty + Frustration and Curiosity + Flow/Engagement. We found that the co-occurrence of Curiosity + Flow/Engagement was related to success and fewer errors when testing code as well as the use of available hints and overall performance.

1 Introduction

Most research into affective states in ITSs and computerized learning systems has assumed that a student experiences one affective state at a time (see meta-analysis [1]). We expand this topic by examining co-occurring affective states, or instances when multiple affective states are experienced at the same time. Determining what affective states co-occur and how those co-occurrence patterns are related to learning is important for more effective design of intelligent tutoring systems (ITSs) that sense and respond to student affect. For example, if confusion and frustration co-occur, it is unclear whether an affect-sensitive ITS should respond to confusion, frustration, or both. We contrast previous research of co-occurring affective states (such as [2]) by focusing on affective states that are learning-centered and arguably more likely to be relavent to ITSs [3]. In particular, we investigated what pairs of affective states co-occurred and how co-occurrence related to interaction events and performance.

2 Method

Ninety nine students completed 35 minutes of problem-solving exercises with the Python computer programming language. Students were retrospectively shown synchronized videos of their face and on-screen activity and were asked to make judgments about what affective states (13 choices including Neutral) they were experiencing at various points in the learning session. With each judgment, students could also voluntarily provide a secondary, co-occurring affective state they experienced. Of 99 novice computer programming students, 46 students had at least 10 secondary affect ratings and provided usable distributions to analyze co-occurring affective states.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 638–639, 2014.

3 Results and Discussion

The most commonly occurring affective states (Anxiety, Boredom, Confusion, Curiosity, Flow/Engagement, and Frustration) were examined for co-occurrence using *Lift*, an association rule learning metric. Lift accounts for the prior probability of each affective state when calculating co-occurrence likelihood, and was computed for each student to ensure independence of data points. We performed one-sample t-tests comparing the Lift values of each co-occurring pair with a test value of 1 (chance). Confusion + Frustration (*Mean Lift* = 1.138, $N = 46$, $p = .123$) and Curiosity + Flow/Engagement (*Mean Lift* = 1.335, $N = 40$, $p = .038$) were the pairs that occurred above chance, through non-significantly for the Confusion-Frustration pair.

We then correlated the Lift of the two co-occurring affective state pairs with key events from the learning session. Due to the small sample size, we focused on the size rather than significance of the correlations and found that Confusion + Frustration did not appear to exhibit any meaningful trends. However, Curiosity + Flow/Engagement was associated in the expected direction with a higher proportion of Key Press events ($r = .208$), less hint usage ($r = -.203$), more error-free code ($r = .314$), and overall better performance ($r = .226$).

Though a seemingly infrequent phenomenon, co-occurring affect states do exist and have some connections to the learning process. Understanding more about the complex nature of affective states in learning environments can lead to better affect detection and thus better affective awareness in intelligent tutoring systems. Affective awareness can in turn can improve the efficacy of teaching in a world where computers play the role of teacher more and more frequently.

Acknowledgment. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958) and the Bill & Melinda Gates Foundation. Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

References

1. D'Mello, S.: A selective meta-analysis on the relative incidence of discrete affective states during learning with technology. Journal of Educational Psychology 105, 1082–1099 (2013)
2. Barrett, L.F.: Discrete Emotions or Dimensions? The Role of Valence Focus and Arousal Focus. Cognition & Emotion 12, 579–599 (1998)
3. Pekrun, R., Stephens, E.J.: Academic emotions. In: Harris, K.R., Graham, S., Urdan, T., Graham, S., Royer, J.M., Zeidner, M. (eds.) APA Educational Psychology Handbook. Individual Differences and Cultural and Contextual Factors, vol. 2, pp. 3–31. American Psychological Association, Washington, DC (2012)

Development of a Learning Environment for Human Body Drawing by Giving Error Awareness for Bones and Contours

Masato Soga, Suguru Yamada, and Hirokazu Taki

Faculty of Systems Engineering, Wakayama University
930 Sakaedani, Wakayama, 640-8510 Japan
soga@sys.wakayama-u.ac.jp

Abstract. We developed a new learning environment that targets beginners and aims at producing mastery of capabilities for observing body proportions and drawing the human body precisely. The learning environment has functions that create awareness of bones in a motif, and that then evaluate bones and contours shown in the human body sketches.

Keywords: Skill, Sketch, Drawing, Learning environment, Recognition.

1 Introduction

Various tools and software have been produced to support drawing of pictures and dia-grams on a virtual plane in computers. For example, Bill Baxter et al. have developed a system to draw pictures on a virtual canvas by operating a paintbrush in virtual space[1)]. It uses a force feedback display device, a Phantom, as the interface and operates a stylus pen as the paintbrush[1)]. However, the system provides only tools sufficient for drawing pictures within a virtual space and does not support learning for drawing skill.

We have built several learning support systems for sketching to date[2-3)]. However, the motif of the systems was still objects such as a glass and a plate. In the system, circumscribed rectangles of the motif were drawn first to catch the rough shape of the motif. In the precedent research[4)], however, objects to be drawn were changed from still objects to moving human beings. The present study succeeds the precedent research and rebuilds the learning support environment of drawing the human body. This study is intended to overcome the shortcomings of earlier studies and to develop a new learning support environment for high-precision human body drawing. The learning support environment for human body drawing to be built in this study is intended for beginners who never learned human body drawing. Our performance target is intended to induce learners to ascertain the proportions of a human body model and draw a human body based on those proportions. In addition, this study devotes no attention to shadow. Therefore, our system is not a learning support system for drawing shadows.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 640–643, 2014.

2 Proposed Method

We have built a new system including the following elements to alleviate the shortcomings of our earlier system. The new system supported function to change composition, bone diagnosis of entire body drawing, and diagnosis of contour drawing

2.1 Flow chart

We might separate our system into three phases of actions, as shown in Figure 1: first is the composition decision phase; second is the bone drawing phase; and third is the contour drawing phase.

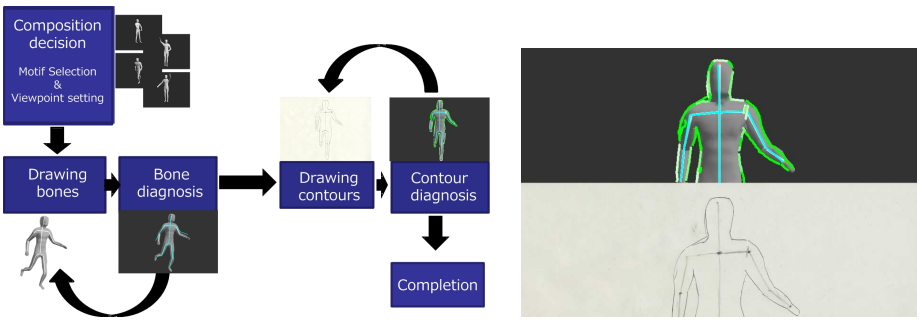


Fig. 1. Left: Flow chart, Right: Contours produced using a handwriting input system

2.2 System Configuration

Our system is so-called application software and uses one set of PCs. The set includes a pen tablet for learner's information input for handwriting. It is an Intuos4 system from WACOM.

Information Input for Handwriting.
A learner of our system performs information input for handwriting, making use of a pen tablet. They might simply draw a human body on the paper set on the tablet with the attached pen with a pencil lead (graphite). This remodeling enables learners to send handwriting information input to a computer through the tablet when they draw a human body using a pen. The right picture of Figure 1 shows the contours displayed on the system screen produced from handwriting information of human body drawing.

3 Learning Support System for Sketching Human Body

Learners exercise drawing guided by the display on the system screen. The system screen resembles the left picture of Figure 2. Learners work in separate three phases of actions in our system. Details of the three phases are described below.

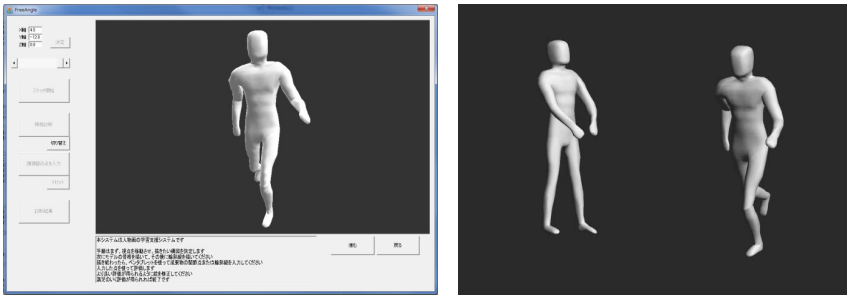


Fig. 2. Left: System screen, Right: Examples of the human body model used

3.1 Phase of Drawing Composition

The composition is determined by the direction and posture of the object to be drawn. We prepare several 3DCG models of naked human bodies which have different postures, which might be directed freely (Right picture of Fig. 2).

Determining the composition fixes the bones and contours of a human body model. Correct information is generated at this point of phases for use in later phases.

3.2 Phase of Drawing Bones

We inserted the phase of drawing bones in sketching human body in our research based on a textbook[5]. The system diagnoses bones of the entire body drawn. The left picture of figure 3 portrays bones of the diagnosed human body.

Learners draw bones on the paper while referring to the example displayed on the system screen. They might switch between display and non-display of the example on screen as they wish. To diagnose bones of the drawn human body, the learner inputs positional information related to joints and ends of bone by pointing at them (totally 15 places) on the paper on the tablet using the attached pen. The system shows bones of human body superimposed on bones of the example (Right picture of Fig. 3).

Fig. 3. Left: Bones of a model human body, Right: Screen displaying bones of the drawn human body

Fig. 4. Left: Characteristic points to generate contours of the human body model, Right: Screen shot displaying human body sketching

3.3 Phase of Drawing Contours

After learners' master bone drawing, they proceed to the phase of contour drawing to practice those associated skills.

Although the bone drawing is diagnosed after one set of drawings, the diagnosis of the contour drawing is done in real time. On completion of a contour drawing on the paper, it is displayed on the system screen. Then the system judges whether that contour is drawn based on a corresponding contour of the human body model. A learner's contour drawing is shown on the screen in four colors. Learners can judge their own performance from the colors (Right picture of Fig. 4).

4 Concluding Remarks

For this study, we built a learning support system for sketching a human body. Our system diagnoses bones and contours of human body drawn by the learner. We wish to add an advising function to our system in our future development.

References

1. Baxter, W., Scheib, V., Lin, C.M., Manocha, D.: DAB: Interactive Haptic Painting with 3D Virtual Brushes. In: Proc. of the 28th Annual Conference on Computer Graphics and Interactive Techniques, pp. 461–468 (2001)
2. Soga, M., Kuriyama, S., Taki, H.: Sketch Learning Environment with Diagnosis and Drawing Guidance from Rough Form to Detailed Contour Form. In: Chang, M., Kuo, R., Kinshuk, Chen, G.-D., Hirose, M. (eds.) Edutainment 2009. LNCS, vol. 5670, p. 109. Springer, Heidelberg (2009)
3. Shirouchi, K., Soga, M., Taki, H.: AR-supported sketch learning environment by drawing from learner-selectable viewpoint. In: ICCE 2010, pp. 533–542 (2010)
4. Soga, M., Fukuda, T., Taki, H.: Sketch Learning Environment for Human Body Figure by Imitative Drawing. In: Velásquez, J.D., Ríos, S.A., Howlett, R.J., Jain, L.C. (eds.) KES 2009, Part II. LNCS, vol. 5712, pp. 599–606. Springer, Heidelberg (2009)
5. Loomis, A.: Figure Drawing for All It's Worth. Titan Books; Facsimile edition (2011)

An Exploratory Study of Learners' Brain States

Ramla Ghali and Claude Frasson

Département d'informatique et de recherche opérationnelle
Université de Montréal
2920 Chemin de la Tour, Montréal
Québec, Canada, H3C 3J7
{ghaliram,frasson}@iro.umontreal.ca

Abstract. In Intelligent Tutoring Systems, continuous analysis of learner's brain states is essential. Several studies have proposed different methods to evaluate learner's mental states in cognitive tasks. However, these studies do not take into account the nature of the cognitive task. In this paper, we have developed various categories of brain games in order to study the variation of some specific brain states (engagement, workload and distraction) depending on the type and difficulty of the game. The preliminary results showed a close relationship between the category of game, the workload mental state and learner's performance.

Keywords: Brain games, Engagement, Workload, Distraction, EEG.

1 Introduction

In Intelligent Tutoring Systems (ITS) recognition of user brain states and cognitive status remains of great importance. To detect and assess users' alertness several studies have been undertaken in the field of artificial intelligence, human computer interaction, cognition and neuroscience [2, 3, 5]. The major part of these systems was based on two fundamental mental metrics, namely, mental workload and mental engagement. *Mental workload* can be seen as the mental effort and energy invested in terms of human information processing during a particular task. *Mental engagement* is related to the level of mental vigilance and alertness during the task. The loss or diminution of engagement is considered as a *distraction* [4].

In this paper we aim to study the behavior and the evolution of these brain states through their EEG signals [1] depending on the category and difficulty of task presented to the learner. Thus, we developed three categories of brain games (*memorization, concentration and reasoning*) and assessed their variation among learners.

2 Experiment and Results

In order to study the behavior of the brain in different cognitive games and track the impact of games' category on different player's brain states, we conducted an experiment with 20 participants (mean age=28, SD=4.67) from Montreal University. This study consists of 3 steps: (1) Initially, we installed the B-Alert X10 headset on the

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 644–646, 2014.

participant, (2) The participant is invited to do three tasks of baseline to establish a classification of brain state, and (3) finally, the participant is invited to play some brain games which are grouped into three main categories (*memorization*, *concentration* and *reasoning*). During all the experiment, EEG was recorded from 9 sensors integrated into a Wi-Fi cap, with a linked-mastoid reference. The sensors are placed according the 10-20 system. The EEG was sampled at a rate of 256 Hz, converted to PSD and processed by the B-Alert software. Thus, three brain states (Workload, Engagement and Distraction) were extracted and analyzed for this study.

An important result obtained from statistical analysis showed that workload and engagement states depend on the game category. This result is assumed after conducting three one way ANOVA tests (see table 1).

Table 1. Relationship between brain states and game category

Brain States	**Results of ANOVA**	
	F	p
Workload	3.32	0.04*
Engagement	18.33	0.000*
Distraction	0.56	0.57

This result is very consistent since learner's concentration and mental activity increase according to the nature of proposed task; more the nature or category of the task is interesting, more he is engaged on the task and he reasons.

Another result confirmed that the workload only depends on the game difficulty (One Way ANOVA: $F(2,224)=0.64$, $p=0.04$*). However, no significant results were found for the engagement and distraction states.

3 Conclusion

In this paper, we have assessed the variation of three brain states (Engagement, Workload and distraction) obtained from EEG signal processing depending on the category and difficulty of game. We have successfully shown that Workload and Engagement states depend significantly on the category of game unlike distraction. Moreover, only the Workload state depends on the difficulty of game.

These results are important from an educative perspective because we should think more in terms of learner cerebral abilities according to the category of task in a game design. More precisely, we think to adapt the tutor's module, in intelligent tutoring systems, according to learner's cerebral states evolution and category of task presented to the learner.

Acknowledgments. We acknowledge the Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT) and McGill University (LEADS) for funding this work.

References

1. Berka, C., Levendowski, D.J., Cvetinovic, M.M., et al.: Real-Time Analysis of EEG Indexes of Alertness, Cognition, and Memory Acquired With a Wireless EEG Headset. International Journal of Human-Computer Interaction 17, 151–170 (2004)
2. Pope, A.T., Bogart, E.H., Bartolome, D.S.: Biocybernetic system evaluates indices of operator engagement in automated task. Biological Psychology 40, 187–195 (1995)
3. Prinzel, L.J., Freeman, F.G., Scerbo, M.W.: A Closed-Loop System for Examining Psychophysiological Measures for Adaptive Task Allocation. IJAP Journal 10, 393–410 (2000)
4. Stevens, R.H., Galloway, T., Berka, C.: EEG-Related Changes in Cognitive Workload, Engagement and Distraction as Students Acquire Problem Solving Skills. In: Conati, C., McCoy, K., Paliouras, G. (eds.) UM 2007. LNCS (LNAI), vol. 4511, pp. 187–196. Springer, Heidelberg (2007)
5. Wilson, G.F.: An analysis of mental workload in pilots during flight using multiple sychophysiological measures. Int. J. Aviat Psychol. 12, 3–18 (2004)

Personalizing Knowledge Tracing: Should We Individualize Slip, Guess, Prior or Learn Rate?

Junjie Gu, Yutao Wang, and Neil T. Heffernan

Department of Computer Science, Worcester Polytechnic Institute
Worcester, MA, USA
{jgu2,yutaowang,nth}@wpi.edu

Abstract. The intelligent tutoring system field is concerned with ways of personalizing to the student. Wang and Heffernan introduced the Student Skill model and showed that it was reliably better than the Knowledge Tracing (KT) model in predictive accuracies. One limitation of their work is that they only investigated one particular way of personalizing, which individualizes all four KT parameters simultaneously. But it may be better if we just use some of the parameters to personalize the model. More generally, we want to address the research question: What are the most important features to personalize? In this work, we systematically explored all 16 possible ways of incorporating student features into the model. We found that prior and slip are the two most important features to individualize, and the best model is the one with all four parameters individualized. Additionally, the one parameter that can be dropped without any hurt to performance is guess.

Keywords: Knowledge Tracing, Bayesian Networks, prediction, personalization, Intelligent Tutoring System.

1 Introduction

The traditional way of modeling student knowledge is Corbett and Anderson's Knowledge Tracing (KT) model [1]. Wang and Heffernan introduced the Student Skill (SS) model [5] and showed that it was reliably better than the KT model in predictive accuracies. The goal of our experiment is to search for the best structures of the SS model by trying all 16 possible ways of incorporating student features. The dataset we used came from the 2009-2010 school year of ASSISTments, containing 1775 distinct students, 123 distinct skills and 695,732 data points. The code and data used in the experiments are available online [6].

2 Methodology and Discussion

In this paper, we investigated the research question: which of the four features: slip, guess, prior, or learn rate of student are most important to individualize in a Bayesian Knowledge Tracing framework. We extended Wang and Heffernan's work by exploring more structures of the SS model and searched for the best combination of individualization features.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 647–648, 2014.

Two major observations were made from the experiments. First, the results showed that if we individualize only one feature for student, the most valuable feature would be slip or prior. It is not surprising that prior is an important feature to individualize since students' prior knowledge differs greatly. Since slip represents the probability of a wrong answer given the student knows the skill, the fact that individualizing slip makes the greatest difference suggests teachers or tutoring systems may need to pay attention to the students with large slip rates to check if they lose interest after mastering a skill or if they are still confused with some aspects of the skill while already mastered the major part of it, and take different actions accordingly.

Second, the single best model is the one with all four parameters personalized for student, but is not reliably different than the one without student guess. This result indicates that if we don't want to individualize all four parameters due to efficiency or data amount, guess rate could be the first feature to consider removing.

This paper investigated a new research question. No one in the ITS field has looked at what parameters to best individualize but this opens up a whole new idea. Our finding that prior and slip are more important to individualize is a novel contribution. But we did not answer the question, why is this so? What is it about prior and slip that gives this extra boost in precision? This raises a new question about what might be better ways to individualize.

Acknowledgements. We acknowledge funding from NSF (#1316736, 1252297, 1109483, 1031398, 0742503), ONR's 'STEM Grand Challenges' and IES (# R305A120125 & R305C100024).

References

1. Corbett, A., Anderson, J.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction 4, 253–278 (1995)
2. Gong, Y., Beck, J.E., Heffernan, N.T.: Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting. In: Aleven, V., Kay, J., Mostow, J. (eds.) ITS 2010, Part I. LNCS, vol. 6094, pp. 35–44. Springer, Heidelberg (2010)
3. Murphy, K.P.: The Bayes Net Toolbox for Matlab, Computing Science and Statistics (2007), http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html
4. Pardos, Z.A., Heffernan, N.T.: Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 255–266. Springer, Heidelberg (2010)
5. Wang, Y., Heffernan, N.T.: The Student Skill Model. In: Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 399–404. Springer, Heidelberg (2012)
6. Personalizing Knowledge Tracing, http://tinyurl.com/ohofyrg

Towards a Learning Ecology Using Modest Computing to Address the 'Banking Model of Education'

Roberto Martinez-Maldonado[1], Ana Pinto[2], and Mario Moreno-Sabido[3]

[1] School of Information Technologies
[2] School of Education, University of Sydney,
NSW 2006, Australia
[3] Department of Systems and Computing,
Instituto Tecnológico de Mérida,
Av. Tecnológico km. 4.5, 97118, México
roberto@it.usyd.edu.au, apin8882@uni.sydney.edu.au,
mario@itmerida.mx

Abstract. It is suggested that most learning technologies used in higher education reinforce what is known as the *banking concept of education*. Teachers and designers often give too much importance to results and content delivery. We explore the role of learning technologies to promote students' meaningful learning, critical thinking and collaboration, as well as teacher's awareness and orchestration. Our approach aims to bridge the gap between principles of pedagogy, student modelling, modest computing and usability. We will show the applicability of our approach as a learning ecology including in three scenarios: face-to-face, remote, and mobile learning environments.

Keywords: Design ·Modest computing ·Learning ecology ·Banking education.

1 Introduction

It has been posed that most learning technologies used in higher education courses reinforce what is known as the *banking model of education* [2]. This term was first used by Freire [3] to describe the type of teacher-student relationship where the former attempts to *deposit* content into the latter. Students are receivers of information rather than critical thinkers [3]. Teachers and designers of learning technologies, often inadvertently, give more importance to the results and the content rather than the process of meaningful learning [6]. We propose the development of a learning ecology to promote students' meaningful learning, critical thinking and collaboration, as well as to enhance teacher's awareness and *orchestration* [1]. We refer to learning ecology as the series of technologies, practices and other contextual factors underpinning student's learning opportunities distributed across multiple spaces. Our research aims to show how the particular affordances of learning technologies can be exploited by teachers and designers to define and enact learning tasks that address the banking model of education by promoting collaboration, dialogue and problem-solving skills.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 649–651, 2014.

2 Proposed Approach and Work in Progress

Figure 1 shows the main elements of our approach. The first element is the *Theoretical Layer*. This includes the pedagogy and learning theories we ground upon. For example, we ground on Freire's ideas [3] that propose ways to tackle banking education through teacher-student dialogue and problem-posing collaborative activities. Various tools that afford these activities have been presented in the ITS/AIED community. This approach is closely related to other well established principles such as the promotion of meaningful learning (e.g. Novak's concept maps) and constructivism [6]. Our second layer aims to bring those theories to practice, into real learning settings. This includes the metaphor of *orchestration* [1]. This is an usability approach that highlights the role of the teacher and technology in terms of coordination and awareness. In addition, we aim to align to the idea of *modest computing* [1] which proposes practical ways to exploit the affordances of technology to make them useful for teachers and learners, even if complexity of the technical approach is minimal.

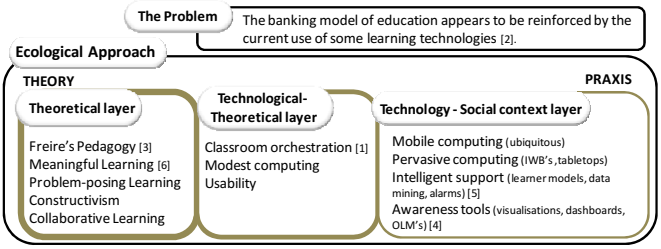


Fig. 1. Our three layered approach: educational theory, orchestration and technologies

Finally, we aim to integrate the *Technology-Social context layer* with the theories, including the use of *intelligent* tools (e.g. as suggested by McCalla [5]; data mining, automated alarms and learner models), or even simpler approaches such as student's data visualisations and teacher's dashboards (modest computing) [4] that can provide enough support to help teachers and students enhance their dialogical relationship. These tools complement a number of emerging technologies that are currently being used for teaching and learning in ubiquitous (e.g. mobile computing), pervasive (e.g. tabletops and interactive whiteboards) and remote (internet-based) environments.

The work will explore the applicability of our approach as a learning ecology in, but not limited to, three potential scenarios: a face-to-face pervasive setting for small-group problem-posing activities [4]; an open learning system for remote collaboration and a mobile ubiquitous environment. From the teacher's perspective our work seeks out to provide them with dashboards that help them *orchestrate* the technology, *monitor* student's progress and receive automated alarms of student's inactivity.

References

1. Dillenbourg, P., Zufferey, G., Alavi, H., Jermann, P., Do-Lenh, S., Bonnard, Q.: Classroom orchestration: The third circle of usability. Proc. CSCL 2011, 510–517 (2011)
2. Freie, J.F., Behuniak., S.M.: Paulo Freire and ICTs: Liberatory Education Theory in a Digital Age. The International Journal of Technology, Knowledge and Society 3(4), 53–62 (2007)
3. Freire, P.: Pedagogy of the oppressed. Continuum, New York (1970)
4. Martinez Maldonado, R., Kay, J., Yacef, K., Schwendimann, B.: An interactive teacher's dashboard for monitoring multiple groups in a multi-tabletop learning environment. In: |Cerri, S.A., Clancey, W.J., Papadourakis, G., Panourgia, K. (eds.) ITS 2012. LNCS, vol. 7315, pp. 482–492. Springer, Heidelberg (2012)
5. Mccalla, G.: The ecological approach to the design of e-learning environments: Purpose-based capture and use of information about learners. Interactive Media in Education (1) (2004)
6. Novak, J.: Meaningful Learning for Empowerment. In: Learning, Creating, and Using knowledge: Concept Maps as Facilitative Tools in Schools and Corporations, pp. 23–40 (2010)

Negotiation Driven Learning

Raja M. Suleman, Riichiro Mizoguchi, and Mitsuru Ikeda

School of Knowledge Science,
Japan Advanced Institute of Science & Technology, Nomi, Ishikawa, Japan
{suleman,mizo,ikeda}@jaist.ac.jp
http://www.jaist.ac.jp

Abstract. Negotiation mechanism used in the current implementations of Open Learner Models is mostly positional based and provides minimal support for learners to understand why their beliefs contradict with that of the system. This study aims at proposing a new paradigm of learning that uses negotiation coupled with targeted responses to motivate a learner and enhance their metacognitive skills along with their cognitive skills.

Keywords: Negotiation, Metacognition, negotiation-driven learning, inter-est-based negotiation, learner motivation.

1 Introduction

In recent years much research has been done in the field of Intelligent Tutoring Systems (ITS) to support and promote independent, self-regulated learning. Open Learner Models (OLMs) aim at enhancing both cognitive and metacognitive skills of a learner through guided content, externalization, scaffolding and negotiation. However, negotiation has been underutilized in the current implementations of OLMs. Negotiating or debating with others allows us to identify alternative perspectives [1]. According to the Constructivist Learning Theory "learning is a process of construction of knowledge through dialogues"[1]. Therefore in this study we propose the paradigm of Negotiation-Driven Learning (NDL) with the aspiration to enhance the role of negotiation as a problem-understanding technique and use it to promote metacognitive activity and enhance learning.

2 Background

The negotiation aspect of the current implementations of OLMs is aimed at solving the problem of the conflict between the learner's beliefs and that of the system [2]. OLMs rely upon the externalization of a learner's knowledge to promote metacognitive skills, while negotiation is generally related with the occurrence and resolution of conflict. Position-Based Negotiation (PBN) is employed to resolve these conflicts, however this approach confines the scope of negotiation as more of a "problem solving" technique rather than a "problem understanding" technique [3].

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 652–653, 2014.

3 Negotiation Driven Learning

NDL aims at exploiting the benefits of Interest-Based Negotiation (IBN) [1], which aims at exploring underlying interests of the parties rather than their negotiating positions. IBN plays a vital role in NDL, since in NDL we are concerned with motivating the learner by trying to understand their reason for holding a particular belief, which in turn can help identify why such beliefs are held and how can a learner be persuaded to change them. The proposed system would generate a Behavioral Model (BM) of the learner as they interact with the system. The BM will include information about the interactions of the learner with the system; their interest in their respective LM, their enthusiasm in discussing their LM, their help-seeking pattern and their confidence in their abilities. The behavioral model will be continuously updated through the Session Manager (SM) which would record interactions of the learner with the system in real-time. Once the baseline BM of the learner is generated it will be used by the Automated Negotiation Agent (ANA) to understand the motivational state the learner is in and use this information to select the best suited negotiation strategy from the Plan Base (PB) to maximize learning.

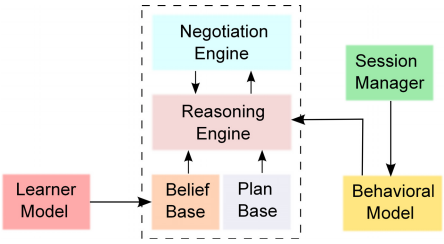


Fig. 1. Proposed Interest-Based Automated Negotiation Agent

4 Conclusion

Negotiation provides an excellent opportunity to challenge the learners and promote metacognitive skills by motivating them to think more objectively about their learning. Although the research on NDL is in its early stages, we believe that the paradigm of NDL holds great potential as it opens up new perspectives of learning by using automated IBN to challenge and intrinsically motivate learners through discussions.

References

1. Miao, Y.: An intelligent tutoring system using interest based negotiation. Control, Automation, Robotics and Vision (2008)
2. Bull, S., Vatrapu, R.: Negotiated Learner Models for Today. In: ICCE 2012: C1 Artificial Intelligence in Education (2012)
3. Pasquier, P., Hollands, R., Dignum, F., Rahwan, I., Sonenberg, L.: An empirical study of interest-based negotiation. In: Proceedings of the Ninth International Conference on Electronic Commerce, Minneapolis, MN, USA, August 19-22 (2007)

SCALE: Student Centered Adaptive Learning Engine

Mary Jean Blink[1], John C. Stamper[1,2], and Ted Carmichael[1]

[1] TutorGen, Inc., Wexford, PA, USA
{mjblink,tcarmichael}@tutorgen.com
[2] Carnegie Mellon University, Pittsburgh, PA, USA
jstamper@cs.cmu.edu

Abstract. We present a new ITS system called SCALE (Student Centered Adaptive Learning Engine), which is focused on improving learning outcomes by using data collected from existing and emerging educational technology systems combined with machine learning techniques to automatically generate adaptive capabilities. This allows for the creation of intelligent tutoring systems in a less costly fashion in terms of time and effort. SCALE uses data logs collected from an existing educational technology system to create the initial adaptivity and then improves over time as additional data is added or with the help of human input. This paper describes two main adaptive capabilities of problem selection and hint generation.

In this research, we present a system called SCALE (Student Centered Adaptive Learning Engine), which has been designed to greatly reduce the high cost of adaptive learning by implementing methods of deriving intelligent tutoring capabilities from collected student data. A key differentiator of SCALE from existing intelligent tutoring systems is that it improves over time with additional data and/or with the help of human input. SCALE employs a 'human-centered, data driven' approach to discover or improve the underlying models that drive learning. Unlike a pure machine learning solution, SCALE is able to report to the developers exactly why the system behaves as it does and allows for human input to maximize improvements through refinement over time. By using existing large datasets previously collected from existing educational technologies, we have tested and validated the techniques used in the system.

While intelligent tutoring systems have delivered significantly better results compared to non-adaptive software, their use has been limited due to the difficulty and cost of creating the adaptive content. Most tutors rely on "student models" that are time consuming to create and require experts to understand the subject material and comprehend the underlying processes used to provide help and feedback. We streamline this work by building initial models using data collected from students solving problems with the intent to enhance the development of ITSs. Previous work in the automatic discovery of student models [4] and automated hint generation [1,5] lay the foundation of the system. SCALE features functionality that includes generating student models that build and organize themselves and improve over time as more data is collected, and dynamically selecting the students' next problems to maximize student learning and minimize time needed to master a set of skills. SCALE also provides

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 654–655, 2014.

hints and feedback on multi-step problems, and utilizes a "feedback loop" to provide continuous improvement of the features over time as more data is collected.

The Knowledge Tracing and problem selection mechanisms use past research on knowledge component (KC) modeling like that used in DataShop [3]. The hint and feedback mechanism utilize past research with the Hint Factory [1], which is a novel method of automatically generating context specific, just-in-time (JIT) hints for students solving multi-step problems [1]. The method is designed to be as specific as possible, derived on-demand, and directed to the student's problem-solving goal, to provide the right type of help at the right time.

We have demonstrated the ability to use data collected from educational technologies to automatically generate adaptive capabilities. The main contribution of this work is to demonstrate the design of the SCALE system to provide problem selection and knowledge tracing, as well as providing just in time hints and feedback. While previous efforts have demonstrated these abilities individually, SCALE represents the first complete commercial viable solution for a complete ITS generated with data.

In the future, SCALE will provide tools that let instructors and developers explore the data using meaningful visualizations that will provide insights into student learning that builds off additional previous research in improving student models [2]. Often this means identifying areas where the existing models seem to contradict the data collected. Built around the concept of curating data, these tools can also prompt developers, educators, and users of the educational software for more human input in order to improve the underlying models that the system generates.

Acknowledgement: This work was supported by NSF Grant IIP-1346448.

References

1. Barnes, T., Stamper, J.: Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 373–382. Springer, Heidelberg (2008)
2. Koedinger, K., McLaughlin, E., Stamper, J.: Automated Student Model Improvement. In: Proceedings of the 5th International Conference on Educational Data Mining (EDM 2012), Chania, Greece, June 19-21, pp. 17–24 (2012)
3. Stamper, J.C., Koedinger, K.R., Baker, R.S.J.d., Skogsholm, A., Leber, B., Demi, S., Yu, S., Spencer, D.: Managing the Educational Dataset Lifecycle with DataShop. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS (LNAI), vol. 6738, pp. 557–559. Springer, Heidelberg (2011)
4. Stamper, J.C., Koedinger, K.R.: Human-machine Student Model Discovery and Improvement Using DataShop. In: Biswas, G., Bull, S., Kay, J., Mitrovic, A. (eds.) AIED 2011. LNCS (LNAI), vol. 6738, pp. 353–360. Springer, Heidelberg (2011)
5. Stamper, J.: Automating the Generation of Production Rules for Intelligent Tutoring Systems. In: Proceedings of the 9th Intl. Conference on Interactive Computer Aided Learning (ICL 2006). Kassel University Press (2006)

Relationship between Student Writing Complexity and Physics Learning in a Text-Based ITS

Reva Freedman and Douglas Krieghbaum

Northern Illinois University, Department of Computer Science, DeKalb, IL, USA
rfreedman@niu.edu, dkrieghbaum@aol.com

Abstract. In this paper we study 2217 essays written during ITS-based physics tutoring. Using output from the Stanford parser, we calculate various simple and more complex linguistic features, including average sentence length, tree height and number of subordinate clauses. Using the WEKA J48 implementation of the C4.5 algorithm and other statistics, we attempt to find relationships between linguistic features, the complexity of the students' text, students' scores on a physics posttest and their learning gain from the tutoring sessions.

Keywords: intelligent tutoring systems, text complexity, linguistic features, parsing, learning gain.

1 Introduction

This paper describes an initial attempt to identify relationships between the linguistic complexity of students' writing about physics word problems and their physics skill.

We examined the following questions:

Question 1: Since students wrote multiple versions of each essay with tutoring in between, was there a significant difference in essay complexity between initial and final essays?

Question 2: How does essay locale (first/medial/last) affect essay complexity?

Question 3: How does linguistic complexity affect learning?

Question 4: Do other features affect essay complexity?

Our study uses a set of 2217 student essays, along with the pretest and posttest scores for these students. We used the Stanford Parser[1] to parse the files. We then used the C4.5 algorithm [1], implemented in WEKA[2] as J48, to test our hypotheses.

2 Methods

The data used in this study were originally collected for testing ITSPOKE [2], a spoken dialogue ITS that uses the facilities of the text-based Why2-Atlas physics ITS [3].

[1] http://nlp.stanford.edu/software/lex-parser.shtml
[2] http://www.cs.waikato.ac.nz/ml/weka/

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 656–659, 2014.

In the ITSPOKE system, a student is given a qualitative problem in elementary college physics. The student responds with an essay answer, then is coached using tutorial dialogue to improve the answer until it is judged acceptable. In general, students revised their essays by adding a missing concept or revising an incorrect one.

Each student did about five problems from a set of 11, with a pretest before the first problem and a posttest after the last. There were 91 students who did a total of 495 problems. The students wrote a total of 2217 essays, or about 4.5 essays per problem. There were a total of 14524 sentences, or about 6.5 sentences per essay.

To reduce the frequency of erroneous parses, we engaged in several forms of data cleaning. The most important was spelling correction, which reduced the unique word count from the 2217 essays (247192 words) from about 2000 words to 1471. In one extreme case, there were 27 wrong spellings for *acceleration*, totaling 130 instances.

This study involved 16 features, including three measures of essay complexity.

1. Experiment type. Sessions could have a human tutor, a synthesized voice, or a response built from prerecorded snippets of human voices.

2. Essay locale. Students wrote between one and 16 essays per problem. We coded essays as the student's first, middle or last attempt.

3–6. Part of speech counts. We counted nouns, verbs, adverbs and prepositions per essay. All counts were normalized by dividing by the number of words in the essay.

7–10. Constituent counts. We also counted the number of noun phrases, adjective phrases, adverb phrases, and prepositional phrases, normalized by essay length.

11–13. Measures of linguistic complexity. We used four measures of linguistic complexity. As in the Flesch readability formula [4], we used the number of words per sentence as a simple measure of writing complexity. This number was calculated at the essay level, i.e., total words in the essay divided by the number of sentences. Since the height of the parse tree is a rough measure of the amount of subordination in a sentence, we used the average height of the parse trees in a student essay as a second measure. The third measure was the average number of subordinate clauses per essay, implemented as the number of SBARs generated by the Stanford parser.

14–16. Student educational data – pretest score, posttest score and learning gain. Pretest and posttest scores were available at the student level, i.e., students took the pretest before their first problem and the posttest after their last. Pretest and posttest scores are expressed as the percent of correct answers. Per convention, the normalized learning gain was defined as the student's improvement with respect to questions missed on the pretest, i.e., (posttest - pretest)/(1 - pretest).

3 Results

We used the two-tailed paired *t*-test to determine whether final student essays were significantly longer than the corresponding initial essays. After deleting problems where students only wrote one essay, there were 482 essays. The average lengths were significantly different, averaging 53 words for the initial essays and 129 for the final essays. The value $t = -22.38$ ($df = 481$) is significant at the $p < .001$ level.

More importantly, we used the two-tailed paired *t*-test to determine whether final student essays contained a larger percentage of SBARs than the corresponding initial essays. We obtained $t = 2.97$ ($df = 481$), which is significant at the $p < .01$ level. Thus students did write more complex essays after being tutored. The other measures of complexity performed equivalently.

Table 1 shows the results for questions 2–4. For question 2, we used J48 to inquire to what extent essay locale (initial, medial or final) predicted whether the given measures of essay complexity were greater or less than the median.

Question 3 is at the student level rather than at the essay level. To compare against student-level measures of learning, we rolled up essay-level linguistic measures to the student level. Student average sentence length for initial (resp. final) essays equals the total number of words in all of the student's initial essays divided by the total number of sentences in those essays. Similarly, student average SBAR percent equals the total SBARs in any of the student's initial (resp. final) essays divided by their total words.

Question 4 asks whether we can identify any of the causes of complexity. We tested all 2^{16} combinations of the 16 basic features. The last two lines of Table 1 show two typical results, i.e., whether experiment type or essay locale can predict whether the percent of SBARs (compared to total words) is greater or less than the median. As the reader can see, the accuracy is similar to the previous experiments.

Table 1. Results for Questions 2–4

Ques.	Input	Output	Accuracy
2	Essay locale	Avg tree height	61.89%
	Essay locale	Avg sentence length	50.07%
	Essay locale	Avg SBARs/sent.	50.29%
3	Student average sentence length for initial essays	Pretest score	50.40%
		Posttest score	51.01%
		Learning gain	50.20%
	Student average SBAR % for initial essays	Pretest score	50.40%
		Posttest score	53.44%
		Learning gain	50.20%
	Student average sentence length for final essays	Posttest score	52.41%
		Learning gain	56.29%
	Student average SBAR % for final essays	Posttest score	53.43%
		Learning gain	50.05%
4	Experiment type	% of SBAR words	57.56 %
	Essay locale	% of SBAR words	52.50 %

4 Conclusions and Related Work

In this paper we used several measures of linguistic complexity to compare the complexity of student essays in a physics ITS with experimental measures, such as the location of the essay in a series, and learning measures, such as the students' posttest

scores and learning gains. Although Student's *t* showed a significant relationship between essay locale (first or last essay of a series) and essay complexity, as measured by the percent of subordinate clauses, most relationships between features were not significant.

In addition to average sentence length, Flesch's Reading Ease formula [4] uses average syllable count. Litman et al. [5] uses statistics derived from counting student and tutor words, including total words, words per turn, and the ratio between students and tutor words. Lipschultz et al. [6] uses the percent of domain-related words in the student's utterance. Coh-Metrix [7] calculates a large number of features, including parts of speech and word frequency statistics, in order to measure cohesion. Connectives and logical operators are two simple linguistic categories that are significant for measuring cohesion. To the best of our knowledge, none of these authors uses complexity metrics derived from syntactic parse trees. We are currently adding some of these measures to our study to see if they improve the accuracy level.

Acknowledgements. We thank Diane Litman for the use of the ITSPOKE data. The first author also wishes to acknowledge her kindness as a sabbatical host.

References

1. Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufman, San Mateo (1992)
2. Litman, D., Silliman, S.: ITSPOKE: An Intelligent Tutoring Spoken Dialogue System. In: Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics Demonstration Papers, pp. 5–8 (2004)
3. VanLehn, K., et al.: The Architecture of Why2-Atlas: A Coach for Qualitative Physics Essay Writing. In: Cerri, S.A., Gouardères, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 158–167. Springer, Heidelberg (2002)
4. Flesch, R.: A new readability yardstick. Journal of Applied Psychology 32, 221–233 (1948)
5. Litman, D., Rosé, C., Forbes-Riley, K., VanLehn, K., Bhembe, D., Silliman, S.: Spoken vs. Typed Human and Computer Dialogue Tutoring. International Journal of Artificial Intelligence in Education 16(2), 145–170 (2005)
6. Lipschultz, M., Litman, D., Jordan, P., Katz, S.: Predicting Changes in Level of Abstraction in Tutor Responses to Students. In: Proceedings of the 24th International FLAIRS Conference, pp. 525–530 (2011)
7. Graesser, A., McNamara, D., Louwerse, M., Cai, Z.: Coh-Metrix: Analysis of Text on Cohesion and Language. Behavior Research Methods, Instruments and Computers 36, 193–202 (2004)

The Impact of Epistemological Beliefs on Student Interactions with an Intelligent Tutoring System

Scotty D. Craig[1], Jun Xie[2], Xudong Huang[2], Author C. Graesser, and Xiangen Hu[2]

[1] Arizona State University, Mesa, AZ USA
[2] University of Memphis, Memphis, TN, USA
Scotty.Craig@asu.edu, {jxie2,xhuang3,graesser,xhu}@memphis.edu

Keywords: Intelligent Tutoring Systems, Epistemological beliefs, ALEKS, Knowledge Space.

Computer technologies may present some potential advantages not present in human instructors. There are currently many effective tutoring systems that have been created. The program chosen for the current evaluation was the Assessment and LEarning in Knowledge Spaces (ALEKS). It uses adaptive programming to best serve the learner's needs in learning mathematics. This program has been shown to be as effective as other mathematics tutoring systems in direct tests [1]. On average, It has been observed that students improved their performance on researcher-conducted tests, standardized state tests, and national tests. Teacher feedback has indicated that ALEKS is a successful training program, noting that students have shown increased math skill, confidence, and retention. In a randomized trial, students were assigned to a technology guided condition using ALEKS or a teacher guided condition received traditional style instruction from human teachers. It was found that the program overall outperformed non program performance on standardized tests the two conditions did not differ from each other [2].

The ALEKS system is an open system in which the student has some control over the learning process. This freedom can be seen as a benefit increasing students control and persistence in the learning process. However, a student's implicit beliefs about learning could provide an obstacle to learning with this type of technology [3]. Dweck and her colleagues [4] studied young children's beliefs about the flexibility of their intelligence can have a direct impact on their learning strategies.

We investigated the impact of student's beliefs on learning within the ALEKS tutoring system to see if different behaviors and outcomes from the system were observed. Because the ALEKS system always provides students with problems that are a challenge to them, Dweck's findings would predict that students that have fixed learning beliefs would attempt fewer problems than students with a flexible view of learning. Similarly, students with a flexible view would be more persistence and show better performance within ALEKS.

The current study was run as an after school program meeting two days per week for 25 week duration. Sixth graders that volunteer for the program interacted with the intelligent tutoring system, ALEKS, in three 20 minute blocks each day and receive two 20- minute breaks (a snack break and a game break) between the learning sessions. The current paper focuses on the data from students in year three of the program that completed the Epistemological belief scale. This sample consisted of $n = 69$ students.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 660–661, 2014.

The metrics used in this study were topics attempted, topics mastered, TCAP pretest and the Epistemological belief scale. The topics attempted and topics mastered metrics were collected by the ALEKS system during the course of the 25 week program. The scores of student's 5th grade Tennessee Comprehensive Assessment Program (TCAP) were used to assess pre-program mathematics knowledge. Students also completed the subsection of the Epistemological belief scale associated with views on fixed learning. This subscale consisted of 10 likert-scaled questions from strongly agree to strongly disagree. This scale was validated for use with middle school students in mathematics [3]. From this test, students were categorized as either having a fixed learning point of view (n=9) or a flexible learning viewpoint (n=60).

A t-test performed on the students 5th grade TCAP indicated not a significant difference between groups on pretest knowledge. However, students with a more flexible view of learning (M=45.38) did perform slightly better than those a fixed view of learning (M=37.00).

A t-test performed on the number of topics attempted by students during the program indicated that there was a significant difference between groups ($t(67) = 2.10$, $p < .05$) with students with a more flexible view of learning attempting more problems (M = 207.70(130.92) versus M=112.44(89.81)).

A t-test performed on the number of topics mastered by students during the program found a significant difference between groups ($t(67) = 2.46$, $p < .05$) with students with a more flexible view of learning showing mastery of more problems (M = 87.17(60.99) versus M=52.89(34.57)).

Our current study found evidence that a student's beliefs about learning can have an impact on both how much a student will try and now much the student will learning when working with educational technology. In that, students that view learning as fixed, will not try as hard within the systems. However, flexible learners show more persistence by attempting more problems than those with a fixed view. It would appear that this persistence also enables students to master more of the material.

Acknowledgments. This research was supported by the Institute for Education Sciences (IES) Grant R305A090528.

References

1. Sabo, K.E., Atkinson, R.K., Barrus, A., Joseph, S., Perez, R.S.: Searching for the two sigma advantage: Evaluating algebra intelligent tutors. Computers in Human Behavior 29, 1833–1840 (2013)
2. Craig, S.D., Hu, X., Graesser, A.C., Bargagliotti, A.E., Sterbinsky, A., Cheney, K.R., Okwumabua, T.: The impact of a technology-based mathematics after-school program using ALEKS on student's knowledge and behaviors. Computers & Education 68, 495–504 (2013)
3. Schommer-Aikins, M., Duell, O.K., Hutter, R.: Epistemological beliefs, mathematical problem-solving beliefs, and academic performance of middle school students. The Elementary School Journal 105(3), 289–303 (2005)
4. Dweck, C.S., Leggett, E.L.: A social-cognitive approach to motivation and personality. Psychological Review 95, 256–272 (1988)

Analyzing Learning Gains in a Competition Intelligent Tutoring System

Pedro J. Muñoz-Merino, Carlos Delgado Kloos, and Manuel Fernández Molina

Department of Telematic Engineering, Universidad Carlos III de Madrid, Spain
{pedmume,cdk}@it.uc3m.es, manuferna@gmail.com

Abstract. We designed and implemented the ISCARE tutor which enables competition one against one solving a collection of exercises in a limited amount of time, with a double adaptation: adaptation of matches so that students with similar knowledge levels are paired; and adaptation of exercises. This study proves that a competition system with the characteristics of ISCARE can be an effective tool for learning, producing important learning gains during the learning process.

Keywords: assessment, competition, learning gains, motivation.

1 Introduction

We designed and implemented a new competition system. More details about this design are in [1] and [2]. This new tutoring system was called ISCARE (Information System for Competition based on pRoblem Solving in Education) which is based on exercise solving. ISCARE incorporates features to try to motivate students (e.g. matches one against one, adaptive challenges in different rounds with classmates who are close in knowledge, real time visualization of the opponent's exercise progress, or leaderboards) and to reduce negative emotions (e.g. adaptation of matches or a reduction of the score difference between winners and losers). In [3], we showed that the ISCARE system can motivate students without generating negative emotions. This paper aims at analyzing if the ISCARE competition system can bring learning.

2 The Experiment

During two different course editions, students interacted with the ISCARE system during a class session at a Computer Architecture Laboratory course. A total of 25 students were considered in the 2013 edition (with adaptation of exercises enabled in ISCARE), and 32 in the 2012 edition (without adaptation). The total number of rounds was set to 4, so each student competed against 4 different classmates. In addition, there were 12 exercises per round with a limited time of 10 minutes. Before the interaction with the competition tutor, students did a pre-test, next interacted for 60 minutes with ISCARE, and next did the post-test. The pre-test and post-test lasted for about 10 minutes each one.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 662–663, 2014.

3 Results

There are different metrics for learning gains in the literature. We followed the one proposed in [4], i.e. *LG1=(post_test – pre_test)/(1 – pre_test)*, for students who got a post-test grade greater than his/her pre-test grade. For students who had a pre-test score greater than the post-test one, we applied *LG2=(post_test – pre_test)/pre_test*. Therefore, the learning gain of any students will range within the interval [-1, 1]. Considering all students in both course editions (N= 57), learning gains were with mean 0.53 and std. deviation 0.46, being the confidence interval at 95% [0.40, 065]. Applying a dependent t-test between the post-test and pre-test, there is a statistically significant difference in favor of the post-test (t=5.626, p=0.000).

If the metric applied for calculating the learning gains were the same for all students and being *LG=(post_test – pre_test)/pre_test* , then learning gains would be with mean 0.29 and std. deviation 0.38, being the confidence interval at 95% [0.19, 0.39].

Learning gains were impressively high. This result proves that interactions with a competition intelligent tutoring system such as ISCARE were effective and that this system improved learning in the presented experience as the pre-test and post-test were designed of a similar level of difficulty. Therefore, an ITS such as ISCARE that implements competition and other educational features can bring learning.

Nevertheless, the interpretation of these results should take into account the specific context (e.g. difference of about 60 minutes between the pre-test and the post-test, or the specific topics covered in the experiment).

Acknowledgments. This work was supported in part by the EEE project TIN 2011-28308-CO3-1 within the Spanish "Plan Nacional de I+D+I," and by the Madrid regional community project eMadrid S2009/TIC-1650

References

1. Fernández Molina, M., Muñoz-Merino, P.J., Muñoz-Organero, M., Delgado Kloos, C.: Educational Justifications for the Design of the ISCARE Computer Based Competition Assessment Tool. In: Leung, H., Popescu, E., Cao, Y., Lau, R.W.H., Nejdl, W. (eds.) ICWL 2011. LNCS, vol. 7048, pp. 289–294. Springer, Heidelberg (2011)
2. Muñoz-Merino, P.J., Fernández Molina, M., Muñoz-Organero, M., Delgado Kloos, C.: An adaptive and innovative question-driven competition-based intelligent tutoring system for learning. Expert Systems With Applications 39(8), 6932–6948 (2012)
3. Muñoz-Merino, P.J., Fernández Molina, M., Muñoz-Organero, M., Delgado Kloos, C.: Motivation and Emotions in Competition Systems for Education: An Empirical Study. IEEE Transactions on Education (in press, 2014), http://dx.doi.org/10.1109/TE.2013.2297318
4. Aleven, V., McLaren, B.M., Roll, I., Koedinger, K.R.: Toward Tutoring Help Seeking: Applying Cognitive Modeling to Meta-Cognitive Skills. In: Lester, J.C., Vicari, R.M., Paraguaçu, F. (eds.) ITS 2004. LNCS, vol. 3220, pp. 227–239. Springer, Heidelberg (2004)

Leveraging Semi-Supervised Learning to Predict Student Problem-Solving Performance in Narrative-Centered Learning Environments

Wookhee Min, Bradford W. Mott, Jonathan P. Rowe, and James C. Lester

North Carolina State University, Raleigh, North Carolina, USA
{wmin,bwmott,jprowe,lester}@ncsu.edu

Abstract. This paper presents a semi-supervised machine-learning approach to predicting whether students will be successful in solving problem-solving tasks within narrative-centered learning environments. Results suggest the approach often outperforms standard supervised learning methods.

Keywords: Narrative-centered learning environments, game-based learning environments, semi-supervised learning.

1 Introduction

Recent years have witnessed growing interest in narrative-centered learning environments, which tightly integrate interactive narratives, digital games, and the adaptive pedagogy of intelligent tutoring systems to generate highly engaging interactive story experiences for personalized learning [1]. Because students have considerable autonomy in these open-ended environments, it is possible for students to unintentionally spend time on problem-solving tasks for which they already have mastery, and inadvertently skip problem-solving tasks where they have gaps in knowledge. This paper introduces a data-driven method for predicting whether students will successfully complete problem-solving tasks based on their prior performance. We leverage self-training semi-supervised learning as a framework for predicting problem-solving task success [2]. We compare this framework to naïve Bayes (NB) and support vector machine-based (SVM) classifiers. Results suggest that self-training often provides the most accurate predictions. The resulting models show significant promise for supporting pedagogical planning in narrative-centered learning environments.

2 Results and Discussion

To evaluate the self-training semi-supervised learning approach to predicting problem-solving performance, we analyze student data from a classroom deployment of CRYSTAL ISLAND [1]. During game play, students progressed in solving the problem scenario by completing concept matrices based on informational texts they encountered in the game. In this work, we analyze student data from 10 frequently attempted concept matrices (on average, 565 students attempted them).

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 664–665, 2014.

Prediction accuracy rates are compared across self-training semi-supervised learning, supervised learning, and a baseline using the majority label. The results fall into two major categories: (1) the self-training method outperforms the corresponding supervised learning technique and baseline, and (2) the baseline performs better than both self-training and supervised learning. In the first category, 3 out of the 10 classifications show that self-training using NBs outperforms both the other two approaches, and 4 out of the 10 classifications show that self-training using SVMs outperforms the other approaches. Table 1 describes pairwise comparisons using one-way repeated measures ANOVA for NBs ($F(1.13, 32.64) = 74.91$, $p < 0.001$) and SVMs ($F(1.01, 39.55) = 34.98$, $p < 0.001$) for the classifications in this category. Statistical significance is measured using least significant difference post-hoc tests. The second category of observations in which the baseline (M=87.39) performs better than both supervised learning (M=84.12) and self-training (M=84.38) consists of relatively easy problem-solving tasks in which 87.39% of students successfully solved the tasks.

Table 1. Average Model Accuracy on Predicting Success of Problem-Solving Tasks in First Category. (statistical significance over * baseline and § supervised learning)

Approach	Naïve Bayes	Support Vector Machine
Baseline	68.107	74.880
Supervised Learning	71.724*	83.338*
Self-Training	73.182*§	83.788*§

We have proposed an approach to predicting problem-solving performance leveraging semi-supervised learning. Results suggest that the self-training semi-supervised learning method can improve predictive models' accuracy over standard supervised learning techniques, and thus support adaptive pedagogical planning in narrative-centered learning environments.

Acknowledgements. This research was supported by the National Science Foundation under Grant IIS-1138497. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors.

References

1. Min, W., Rowe, J.P., Mott, B.W., Lester, J.C.: Personalizing Embedded Assessment Sequences in Narrative-Centered Learning Environments: A Collaborative Filtering Approach. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 369–378. Springer, Heidelberg (2013)
2. Zhu, X., Goldberg, A.: Introduction to semi-supervised learning. Synthesis Lectures on Artificial Intelligence and Machine learning 3(1), 1–130 (2009)

Opening the Door to Philosophy for Teachers with GYM-Author

Valery Psyché[1], Jacqueline Bourdeau[2], Jules Mozes[1], Alexandre Kalemjian[3], Pierre Poirier[1], Roger Nkambou[1], Alexie Miquelon[1], and Céline Maurice[1]

[1] Université du Québec à Montréal, P.O. Box 8888, Station Centre-ville, Montréal, QC H3C 3P8 Canada
{psyche.valery,mozes.jules,poirier.pierre,nkambou.roger, miquelon.alexie,maurice.celine}@uqam.ca

[2] Télé-Université, 5800, rue Saint-Denis, bureau 1105, Montréal, QC H2S 3L5, Canada
bourdeau.jacqueline@teluq.ca

[3] Collège Montmorency, 475 bd. de l'Avenir, Laval, QC H7N 5H9, Canada
AKalemjian@cmontmorency.qc.ca

Abstract. Can a system have the ability to dynamically generate, on demand, a large number of self-learning and self-assessment exercises in order to supplement a learning environment in philosophy? We addressed this issue with our Phi-GYM project with its integrated authoring tool for tutoring systems in philosophy. Our motivation in designing the authoring tool was to: (1) Find an effective way to semi-automatically generate a wide range of exercises, and; (2) Provide philosophy teachers with an easy, autonomous, and collective way to create exercises related to classical philosophical texts without worrying about any technology.

Keywords: Authoring system; Tutoring system; Exercise generation; Philosophy; Text reading and comprehension.

1 Introduction

One goal of massive online education is to provide learning for thousands of students. Rapid and easy design of material that respects proven educational paradigms in a given field is thus essential to ensure the quality of such courses. The Quebec government thus undertook to fund technologies that facilitate the easy production of open, online, self-learning and self-assessment material. Thus was born the metaphor that inspired the *Philosophical Gymnasium*[1] (Phi-GYM), which aims to allow practice the intellectual gymnastics needed by Quebec college[2] students, who all have to read and write philosophical texts. The first issue we addressed and which led to the development of an authoring tool was: How can a system have the ability to dynamically generate, on demand, a multitude

[1] Known in French as "*Le Gymnase Philosophique*".

[2] The College level in Quebec stands between the high school (grades 7-12) and university levels.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 666–669, 2014.

of pedagogically relevant self-learning and self-assessment exercises in order to supplement philosophys planned learning environment? Our motivations in designing the authoring system were: (1) To find an effective way to continuously update the learning environment in order to provide a wide range of exercises; (2) To provide philosophy teachers with an easy, autonomous, and collective way to pen exercises, allowing them to focus on the exercises philosophical content and not worry about technology.

There is currently no such an authoring system. The projects originality also resides in the fact that both the learning (GYM-Tutor) and authoring (GYM-Author) environment are designed as an integrated one and developed in parallel. After a review of related work, we focus on the architecture of the GYM-Author with an emphasis on its semi-automatic generation function and then explain how the two systems are integrated.

2 Related Work

The beginning of the 2000s saw $AIED^i$ and ITS research communities interested in authoring systems, and their classification [1-5], as they addressed the problem of the generation of learning material. Authoring systems do not support the learning itself; their purpose is to support the creation and generation of learning materials for the ITS environment, and, often provide means to generate this learning semi-automatically and even automatically. As highlighted by [6], semi-automatic generators of exercises combine the advantages of [the automatic and manual] classes of generators, which is why we chose to design GYM-Author as semi-automatic exercise generator. We believe these to be the most adaptive, and they fit one of our primary goals to help teachers in a more efficient way.

3 Design and Architecture Overview

GYM-Author is a web based pedagogy-oriented authoring tool in philosophy, a domain where much of the learning is made through the reading of classical texts, often written hundreds and even thousands years ago, and by producing written texts about these classical texts (either about the texts themselves or about their content).

GYM-Author provides many roles to teachers; the *learning designer role* and the *knowledge expert role* (Fig. 1). The learning designer is responsible for the edition of various pedagogical scenarios according to the objectives and the contexts of learning. The knowledge expert is responsible for creating contents and exercises in philosophy following a predefined pedagogical scenario constrained by the types of questions offered: Multiple choice questions, Tagging, Cloze test and brief answer. The knowledge expert can edit the domain knowledge in order to categorize the didactic material. For instance, the philosophical notions can be organized according to the author of the philosophical text, the period of text, etc. GYM-Author, provides a predefined scenario, which is a structure of learning modules, activities, and exercises based on underlying reading and

writing strategies. This scenario is only editable by the learning designer while it is implicit for the knowledge expert. Thus, Gym-author lets the knowledge expert deal with cognitive aspects (organization of domain knowledge), leaving the pedagogical aspects (organization of learning objectives and contexts) to the learning designer. GYM-Authors architecture owns the main tools (exercise generation, scenario authoring, collective edition and preview) that support the authoring process, the domain model and the pedagogical model (Fig. 1). GYM-Author and GYM-Tutors architectures share some components (Fig. 2). Both share: (1) the same Domain Model populated by philosophical texts, quotes, notions and related contents; (2) the same Pedagogical Model (pedagogical scenario, hints) but they use different parts of it when needed. While GYM-Author owns an Author Model (profile, scenario building rights, collaboration rights), GYM-Tutor has a Learner Model (profile, progression, performance, philosophical text annotated). The core of GYM-Author is an Exercise generator tool and Pedagogical scenario-authoring tool. The core of GYM-Tutor is to be done; its the inference engine that proceeds to the cognitive diagnostic task based on the Learner Model.

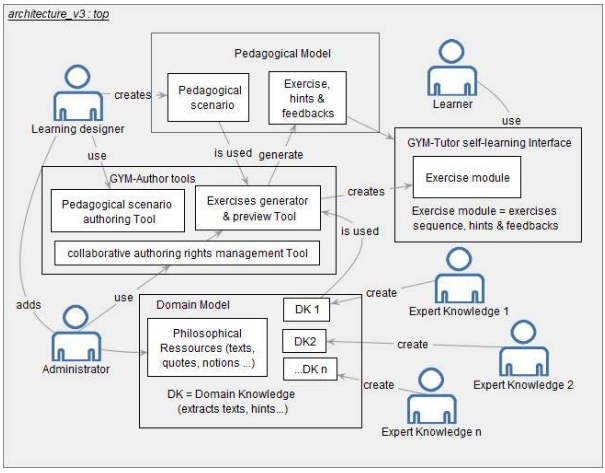


Fig. 1. GYM-Author architecture

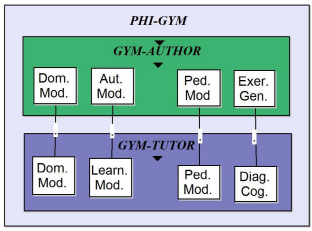


Fig. 2. Phi-GYM system

4 Usability Evaluation

To improve and validate the design of the Phi-GYM system during its production, we have adopted a participatory design approach, based on short iterative cycles of design and development with evaluations. The expert in ergonomics intervened for evaluation by inspection then we conducted scenario-based qualitative tests with eight potential end-users. The data analysis has shown that several aspects which seem obvious to us were not for users on their first try

(I dont know where to go, I cant find any instruction that help me, This text doesnt seem important to me so I skip it) but the system appears to be well-mastered, memorized and pleasant after a full exploration (Its fun to do this kind of exercises, Finally I find this tool convenient). The user quickly becomes effective with the system and manages to generate perfectly usable exercises. These results are already considered in the next iteration and further tests are planned including with GymTutor.

Acknowledgment. Ministre de lducation du Qubec (funding); Institute of Cognitive Science at UQAM; GDAC Laboratory; LICEF Research Centre et Laboratoire LORIT Laboratory; Sasha Vorobyova, copyediter.

References

1. Murray, T.: Authoring intelligent tutoring systems: An analysis of the state of the art. International Journal of Artificial Intelligence in Education (IJAIED) 10, 98–129 (1999)
2. Murray, T.: An Overview of Intelligent Tutoring System Authoring Tools: Updated analysis of the state of the art. In: Authoring Tools for Advanced Technology learning environments, pp. 491–544. Springer (2003)
3. Mitrovic, A., Koedinger, K.: Special Issue on Authoring Intelligent Tutoring Systems. International Journal of Artificial Intelligence in Education 19(2) (2009)
4. Woolf, B.P.: Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning. Morgan Kaufmann (2010)
5. Nkambou, R., Bourdeau, J., Psyché, V.: Building intelligent tutoring systems: An overview. In: Nkambou, R., Bourdeau, J., Mizoguchi, R. (eds.) Advances in Intelligent Tutoring Systems. SCI, vol. 308, pp. 361–375. Springer, Heidelberg (2010)
6. Cablé, B., Guin, N., Lefevre, M.: An Authoring Tool for Semi-automatic Generation of Self-assessment Exercises. In: Lane, H.C., Yacef, K., Mostow, J., Pavlik, P. (eds.) AIED 2013. LNCS (LNAI), vol. 7926, pp. 679–682. Springer, Heidelberg (2013)

Using Log Data to Predict Response Behaviors in Classroom Discussions

Ruth Wylie[1], Brandon Helding[2], Robert Talbot[3], Michelene T.H. Chi[1], Susan Trickett[2], and Rodney D. Nielsen[4]

[1] Arizona State University
{ruth.wylie,michelene.chi}@asu.edu
[2] Boulder Language Technologies
{b.a.helding,sbtrickett}@gmail.com
[3] University of Colorado Denver
robert.talbot@ucdenver.edu
[4] University of North Texas
rodney.nielsen@unt.edu

Abstract. The Comprehension SEEDING system allows students to respond to an open-ended question using tablet computers; the system provides formative feedback to teachers to facilitate discussion and encourage students to engage in reflective behaviors. Data from a semester-long intervention suggested that few students engaged in this reflective process, leading us to question under what conditions the reflection process does or will occur. Using logistic regression, we investigated different ways the system was used, and what types of usage lead to desired, reflective behavior.

Keywords: data mining, classroom response technology, classroom discussion.

1 Introduction

Comprehension SEEDING is a new type of classroom response technology in which a teacher poses a discussion question and students reply by typing an answer on a tablet computer. After a sufficient number of responses have been received, the system automatically clusters the responses (for a description of the system and clustering algorithm and other system components, see [1]). Clustering allows teachers to quickly evaluate the class's understanding and use that information to lead a productive discussion, which, in turn, should encourage students to reflect on and revise their original answer. To facilitate the discussion process, the system includes a number of features (e.g., teachers can "pause" students from entering or editing responses, teachers can display individual or cluster representatives, etc.)

Preliminary data from a semester-long pilot study showed that students only revised or changed their answers 32% of the time. To raise this rate, we examined the log data associated with how students used the system, identifying types of system use that lead to the revisions/reflective behavior. We used the Comprehension SEEDING system log data to ask: What student and teacher behaviors predicted a statistically significant change in likelihood that a student would revise his/her original response?

Data were collected from 8 sixth grade science teachers (416 students) who used the system to ask 414 questions, which generated 8,751 question-response pairs. We

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 670–671, 2014.

first coded answers to determine the type of revision students made to their initial response (no change, substantive change, or non-substantive change.) Two raters coded 20% of the data and achieved Kappa = 0.81 ($p<0.01$), 95% CI (0.75, 0.86) (characterized as almost perfect agreement [2]). The remainder of the data was coded by a single coder. In order to predict whether or not students would revise their answer, the following predictors were used:

- Student variables: teacher, class, pretest score,
- Experience variables: # of days into the school year, # of times class period had previously used SEEDING, # of times teacher had previously used SEEDING with sixth grade science classes
- Question variables: seconds to first response, whether or not the teacher displayed a response, and whether or not the teacher paused the question.

2 Results

The outcome (dependent variable) focused on three types of revisions: no change (68.1% of the responses), non-substantive change (e.g., grammar and spelling changes) (5.3% of the responses) and substantive change (26.6% of the responses). Because our outcome was categorical, we examined potential predictors using multinomial logistic regression. Teacher, number of days into school year, and the pretest score, did not predict the likelihood of a student changing his or her answer with statistical significance. On the other hand, class, seconds to first response, number of times teachers used SEEDING, number of times a class used SEEDING, and whether or not the teacher displayed student responses did change the likelihood of a student changing his/her response. Among those predictors, seconds to first response mattered statistically, but did not correspond with a meaningful change in student behavior. Whether or not a teacher displayed student responses increased the odds of a non-substantive (grammar/spelling) change by 61%; however, displaying student answers did not predict a change in the likelihood of a substantive revision. The number of times the class used SEEDING positively predicted likelihoods to substantially change answers; using SEEDING more increased the odds of changing a response 13%. Surprisingly, the data show that the number of times teachers used SEEDING with their sixth grade science classes actually decreased the likelihood that a student would substantially change his/her answer by 9%. This result is complicated by the fact that teachers may have used the system for classes that were not part of the study (i.e., non-sixth grade classes) and is an interesting area for future research.

Acknowledgments. The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120808 to the University of North Texas.

References

1. Paiva, F., Glenn, J., Mazidi, K., Talbot, B., Wylie, R., Chi, M., Dutilly, E., Helding, B., Lin, M., Trickett, S., Nielsen, R.: Comprehension SEEDING: Comprehension through Self Explanation, Enhanced Discussion, and INquiry Generation. In: Trausan-Matu, S., Boyer, K.E., Crosby, M., Panourgia, K. (eds.) ITS 2014. LNCS, vol. 8474, pp. 283–293. Springer, Heidelberg (2014)
2. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics 33(1), 159–174 (1977)

A Rule-Based Recommender System to Suggest Learning Tasks*

Hazra Imran, Mohammad Belghis-Zadeh, Ting-Wen Chang, Kinshuk, and Sabine Graf

Athabasca University, Edmonton, Canada
{hazraimran,tingwenchang,kinshuk,sabineg}@athabascau.ca,
mobelghis@yahoo.ca

Abstract. Learner-centered learning can be defined as an approach to learning in which learners choose the topic to study and learning tasks. Because of available choices, learners can find it difficult to make a decision about which of the topics/tasks would be more appropriate for them. Identifying other learners with similar characteristics and then considering the tasks that worked well, makes it possible to suggest appropriate tasks to a learner. Based on this concept, we introduce a rule-based recommender system that supports learner-centered learning and helps learners to select learning tasks that are most suitable for them, with the focus on maximizing their learning.

Keywords: Personalization, Learning Management Systems, Recommender System.

1 Introduction

Nowadays, recommender systems in e-learning are gaining a lot of attention as these systems support learners by providing relevant and personalized information that is likely of interest to them. In learner-centered learning, learners can choose the topics to learn and tasks to complete. The benefit of such approach is that it motivates learners and can enhance their performance [1]. When learners make choices about what to learn then typically they decide based on their interest and knowledge. However, sometimes, learners can find it difficult to choose what to learn. In such cases, recommender systems can help making appropriate selections.

In this paper, we propose a rule-based recommender system that supports learners in learner-centered settings where courses have assignments that allow learners to select from a number of available learning tasks with different difficulty levels (e.g., easy, moderate and challenging). In such assignments, learners can select different combinations of tasks (e.g., many easy tasks, a few challenging tasks, etc.). Our proposed system provides recommendations of which learning tasks are most suitable for learners, considering the different difficulty levels of the learning tasks. The provided

* The authors acknowledge the support of Mitacs, Athabasca University, NSERC, iCORE, Xerox, and the research related gift funding by Mr. A. Markin.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 672–673, 2014.

recommendations are based on the learner's previous performance, the tasks that the learner planned to do and the performance of learners with similar characteristics. The system has been designed to be integrated in any learning management system. The architecture of the proposed recommender system is presented in Fig 1.

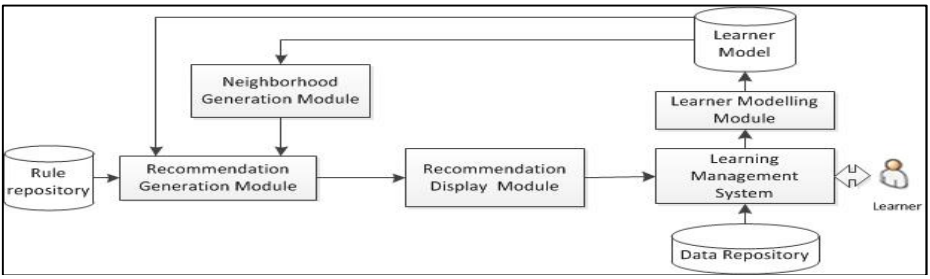


Fig. 1. Architecture of Rule-Based Recommender System

As shown in Fig. 1, the proposed recommender system consists of four modules:

- Learner Modelling Module aims at gathering information about the learners' learning styles based on the Felder-Silverman learning style model [2], prior knowledge, expertise level and performance.
- Neighborhood Generation Module aims to find learners who share the same characteristics as a target learner (a learner for whom a recommendation is calculated).
- Recommendation Generation Module aims to generate suitable recommendations for a target learner. These recommendations are generated based on a set of rules which are used to rank the appropriateness of difficulty levels and then select the appropriate learning tasks.
- Recommendation Display Module displays recommendations to the learner.

The main benefit of the proposed system is that instead of ratings from learners (which are often used in recommender systems), the proposed system uses actual performance of other similar learners to identify which tasks worked well for those similar learners. In addition, the proposed recommender system uses an advanced neighborhood approach to find similar learners. This enables our system to generate more suitable recommendations that support learners more effectively, leading to a better selection of learning tasks from which learners can benefit most.

References

1. Rich, J.D., Colon, A.N., Mines, D., Council, C.: Learner-centered assessment strategies for greater student retention. Universal Journal of Education and General Studies 2(6), 196–199 (2013)
2. Felder, R.M., Silverman, L.K.: Learning and Teaching Styles in Engineering Education. Engineering Education 78(7), 674–681 (1988) Proceeded by a preface in 2002, http://www4.ncsu.edu/unity/lockers/users/f/felder/public/Papers/LS-1988.pdf

Reducing Student Hint Use by Creating Buggy Messages from Machine Learned Incorrect Processes

Douglas Selent and Neil Heffernan, USA

Worcester Polytechnic Institute

Abstract. The goal of this research is to improve existing forms of help in tutoring systems by using "Buggy" messages, which are a simple text message specific to the incorrect answer. Buggy messages are created from machine learned incorrect processes based on the student's incorrect answer. A randomized control trial is run in ASSISTments to determine if the buggy messages were effective.

Keywords: Buggy Messages, Randomized Control Trial, Machine Learning.

1 Introduction

In many tutoring systems students are often given the option to ask for help. In the ASSISTments tutoring system hints are the most common form of help. "Buggy" messages are when a message appears on the screen after a student enters a wrong answer. They were first introduced in [4] and a history of related research is summarized in [5]. Messages will only appear if the answer given by the student matches a predicted wrong answer. Buggy messages are supported but rarely used in ASSISTments because it takes too much time to predict and enter all the incorrect messages for all possible common wrong answers. The machine learning algorithm in this paper can identify most wrong answers and exactly how a student derived them by taking advantage of the existing infrastructure and data in ASSISTments. Buggy messages have several advantages over hints such as not giving the student the answer and provide more personalized tutoring which address issues mentioned in [1-3].

2 Machine Learning Process

The entire machine learning process can be broken down into five main parts. All possible solution paths to an incorrect answer are derived, reconstructed, stored, generalized, and finally chosen. Basically all incorrect solution paths are derived for each incorrect answer in a template Since there can be several ways to reach a single answer the machine learning algorithm uses the template-instance infrastructure in ASSISTments to remove ambiguity on which incorrect process a student used and generalize the incorrect process across all instances.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 674–675, 2014.

3 Randomized Control Trial

The purpose of this study is to see if buggy messages will cause an increase in learning rate compared to hints. A randomized control trial was run on the ASSISTments intelligent tutoring system. The control group received hints and the experiment group received hints and buggy messages. Out of all the statistics examined the only significance difference was the amount of hints used, with the experimental group using fewer hints. Hints per problem had a significant reduction ($p=0.03$), with an effect size of 0.24, where the group with hints ($n=172$) used 0.48 hints her problem and the group with buggy messages ($n=184$) used 0.35 hints per problem.

4 Contributions, Conclusions and Future Work

The contributions this paper makes are that we propose an algorithm to search and discover functions that can predict common wrong answers across problems that are generated from the same template. We think that this algorithm can help ASSISTments and other system that create problems with similar algorithms. To test to see if the bug messages were effective we reported on a study where we showed that although we did not get an increase in student learning, we did show a positive effect in decreasing the number of hints that students use. As future work there needs to be a way to create bug messages that are known to be good bug messages, since the ones used in the randomized control trial may, or may not, have been good messages.

Acknowledgement. We acknowledge funding from NSF (#1316736, 1252297, 1109483, 1031398, and 0742503), ONR's 'STEM Grand Challenges' and IES (# R305A120125 & R305C100024).

References

1. Aleven, V., Koedinger, K.R.: Limitations of student control: Do students know when they need help? In: Gauthier, G., VanLehn, K., Frasson, C. (eds.) ITS 2000. LNCS, vol. 1839, pp. 292–303. Springer, Heidelberg (2000)
2. Baker, R., Walonoski, J., Heffernan, N., Roll, I., Corbett, A., Koedinger, K.: Why students engage in "gaming the system" behavior in interactive learning environments. Journal of Interactive Learning Research 19(2), 185–224 (2008)
3. Beck, J.E., Chang, K.-m., Mostow, J., Corbett, A.T.: Does help help? Introducing the Bayesian Evaluation and Assessment methodology. In: Woolf, B.P., Aïmeur, E., Nkambou, R., Lajoie, S. (eds.) ITS 2008. LNCS, vol. 5091, pp. 383–394. Springer, Heidelberg (2008)
4. Brown, J.S., Burton, R.R.: Diagnostic models for procedural bugs in basic mathematical skills. Cognitive Science 2(2), 155–192 (1978)
5. Sison, R., Shimura, M.: Student modeling and machine learning. International Journal of Artificial Intelligence in Education (IJAIED) 9, 128–158 (1998)

Modeling Student Dropout in Tutoring Systems

Michael Eagle and Tiffany Barnes

North Carolina State University, Department of Computer Science,
890 Oval Drive, Campus Box 8206 Raleigh, NC 27695-8206
{mjeagle,tmbarnes}@ncsu.edu

Abstract. Intelligent tutors have been shown to be almost as effective as human tutors in supporting learning in many domains. However, the construction of intelligent tutors can be costly. One way to address this problem is to use previously collected data to generate models to provide intelligent feedback to otherwise non-personalized tutors. In this work, we explore how we can use previously collected data to build models of student dropout over time; we define dropout as ceasing to interact with the tutor before the completion of all required tasks. We use survival analysis, a statistical method of measuring time to event data, to model how long we can expect students to interact with a tutor. Future work will explore ways to use these models to to provide personalized feedback, with the goal of preventing students from dropping out.

1 Introduction

Student dropout, which we are defining as students who quit interacting with the tutor before completion of all required tasks, can confound study results. Dropout can be a form of selection bias in the form of attrition bias [6]. The effects of attrition bias can threaten the internal validity of a study, as well as harm the studies generalizability if the attrition causes the sample to become different than the target population [5].

Anderson, Corbett, Koedinger, and Pelletier used mean duration data to compare differences between groups of students with and without intelligent feedback in the LISP tutor [1]. The authors state that the mean times (for the control group) are underestimates, as many students in the control (no-feedback group) did not complete all assignments. In other words, if the control group persisted, the time they took to complete tasks would have been longer than the observed durations. Differences in posttest scores could also be partially explained by the fact that students in control group did not complete the tutor.

In our previous work, we modeled the total time-in-tutor using survival analysis and found that introduction of automatically generated hints reduced the time needed to complete the tutor by almost half [3]. For this paper, rather than focusing on the *total time until tutor completion* we will focus on the *total time until tutor dropout*. We want to know, how long will a student interact with a tutoring system before dropping out. We also want to explore how we can use these models of dropout to create new interventions aimed at keeping students engaged with the tutor until completion.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 676–678, 2014.

2 Preliminary Results

Survival analysis is a series of statistical techniques that deal with the modeling of time to event data [4]. It derived its name from its start within medical literature. Survival analysis is also known as reliability analysis or duration analysis. It has been used to model the reliability of parts, measure bird disturbance [2], as well as to model the activity of Wikipedia editors [9].

We start by first plotting the Kaplan-Meier survival estimator, see figure 1, which is represented as a series of declining steps which is intended to approach the true survival function. This plot is useful as a visualization for the overall differences between the two groups. We perform our preliminary analysis on the Spring and Fall 2009 Deep Thought logic tutor dataset as analyzed by Stamper, Eagle, and Barnes [7]. We look specifically at 151 students who stopped using the tutor before completing all of the questions required for the homework assignment. Next, we generate an Accelerated Failure Time (AFT) model which assumes that the effect of any covariates, is to accelerate the time until dropout by some constant factor [8]. Application of the AFT model provides us with coefficients of the model, the intercept (mean) was $\mu = 4.20$ and the SD (scale) was $\sigma = 1.44$. The median of the survival function, the location where 50% of people have dropped out of the tutor, is found by $e^{\mu} = e^{4.20} = 66.89$, meaning that half of the students had dropped out after about an hour of tutor interactions. We have plotted the resulting survival curve in figure 1.

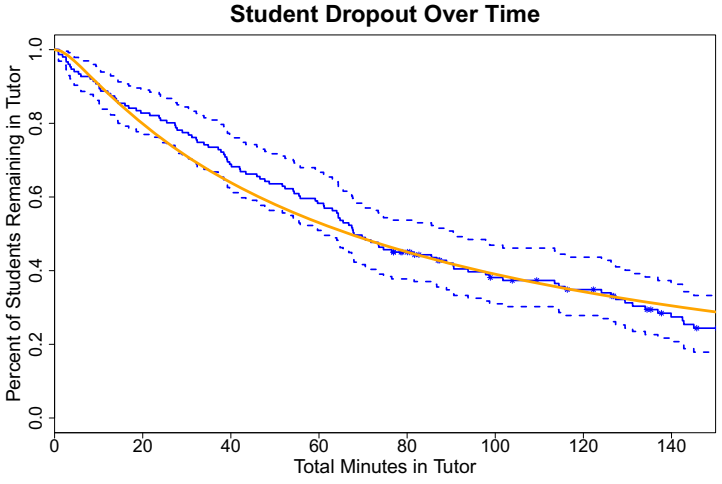


Fig. 1. The Kaplan-Meier survival estimation and corresponding 95% confidence intervals show the percent of students remaining in tutor over time. The lighter (orange) line is the AFT model produced from the same data.

3 Discussion and Future Work

The results of this analysis provide us with a model of how long we can expect students to interact with a tutor. In future work we can explore potential differences in duration with covariates (such as pretest scores,) to provide more detailed models. We are also interested in exploring ways to use this information to inform dropout preventing interventions. We hypothesize that we can mitigate dropout by providing feedback when students reach certain thresholds of time within specific problems as well as total tutor time. To test our hypothesis, we will build survival models based on past student data, and using these models we will provide feedback in the form of a pop-up window that will encourage the student, as well as provide them with resources if they are struggling. Although, for this to be effective we need to have a better understanding of the reasons why students dropout. For a pilot study, we might want to flag these students and have the course instructor or teaching assistants reach out. We can augment our dropout models with information about the students current tutor performance, to get an idea of how likely the student is to complete the tutor. Overall, the use of survival modeling will provide us with more accurate representations of student time-in-tutor, and we can use this information to create interventions that will reduce the number of students who quit the tutor without finishing.

References

1. Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R.: Cognitive tutors: Lessons learned. The Journal of the Learning Sciences 4(2), 167–207 (1995)
2. Chatwin, T.A., Joy, R., Burger, A.E.: Set-back distances to protect nesting and roosting seabirds off vancouver island from boat disturbance. Waterbirds 36(1), 43–52 (2013)
3. Eagle, M., Barnes, T.: Survival analysis on duration data in intelligent tutors. In: Intelligent Tutoring Systems, Honolulu, Hawaii (2014)
4. Hosmer, D.W., Lemeshow, S., May, S.: Applied Survival Analysis: Regression Modeling of Time to Event Data, 2nd edn. Wiley-Interscience, New York (2008)
5. McGuigan, K.A., Ellickson, P.L., Hays, R.D., Bell, R.M.: Adjusting for attrition in school-based samples bias, precision, and cost trade-offs of three methods. Evaluation Review 21(5), 554–567 (1997)
6. Miller, R.B., Hollist, C.S.: Attrition bias (2007)
7. Stamper, J., Eagle, M., Barnes, T., Croy, M.: Experimental evaluation of automatic hint generation for a logic tutor. International Journal of Artificial Intelligence in Education (IJAIED) 22(1), 3–18 (2012)
8. Therneau, T.M., Grambsch, P.M.: Modeling Survival Data: Extending the Cox Model. Springer, New York (2000)
9. Zhang, D., Prior, K., Levene, M., Mao, R., van Liere, D.: Leave or stay: the departure dynamics of wikipedia editors. In: Zhou, S., Zhang, S., Karypis, G. (eds.) ADMA 2012. LNCS (LNAI), vol. 7713, pp. 1–14. Springer, Heidelberg (2012)

A Tool for Summarizing Students' Changes across Drafts

Homa B. Hashemi and Christian D. Schunn

University of Pittsburgh, Pittsburgh, PA 15260, USA
{hob10,schunn}@pitt.edu

Abstract. In the context of popular peer review educational approaches, teachers wish to know whether the students are benefiting from peer reviews and applying the changes in their second drafts. This paper presents a tool for teachers that compares information about students first and second drafts of papers focusing on the extent and type of changes in the papers.

Keywords: Compare drafts, Students' changes, Peer review, Learning analytics.

1 Introduction

Computer-supported peer review systems are popular educational systems that help students to improve their learning and also teachers to make better decisions. Generally in a peer review system, instructors assign students a writing task and then students submit their first drafts to the system. In turn, the system assigns each paper to several peers in the class to review according to criteria provided by the instructor. Finally, the system distributes the peer reviews to the authors, who read the reviews and, ideally, use them to improve their second drafts.

In the context of a peer review system, many teachers use the first draft as a peer review exercise and the second draft as an instructor-graded object; other teachers have multiple rounds of peer review. In either case, teachers wish to know whether students are playing the system (or otherwise not benefiting from peer review) and just turning in the same paper for the second draft. Teachers also wish to know whether changes tend to be relatively minor ones or whether more substantive revisions are taking place.

Occasionally instructors use tools like Microsoft Office Word to compare drafts of documents one by one. There are also some nicer tools specifically designed for text version comparison (such as [2,3]), but they all need to be fed by one pair of drafts at each time and it would a daunting task for a teacher to compare all the drafts of a class manually. Therefore, an automatic tool to extract comparison information and summarize it for teachers would be beneficial for them to make sense of the students' changes.

In this paper, we address this issue by designing a tool for teachers to summarize students' changes across drafts. Our goal is to create a prototype interface

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 679–682, 2014.

that draws attention to problematic cases of each student assignment and also make general class trends salient, even if the class has multiple hundreds of students. For this study, we employ a questionnaire which taps the general attitudes of teachers towards our tool. The results show that this tool would be useful for teachers but it needs additional improvements.

2 Methods

Given two versions of a text, we first split the texts into sentences and then use an off-the-shelf text version comparison tool to find and highlight the changes between two drafts. In our experiments, we used Compare Suite [1] which provides a command line version and also a graphic representation of the differences. The generated report of this tool counts changes and shows the track-changes view. However, this information is overwhelming and we did a simple post-processing over the output of these reports to aggregate the adjacent modified errors. Figure 1 shows an example of detailed report of the draft comparison.

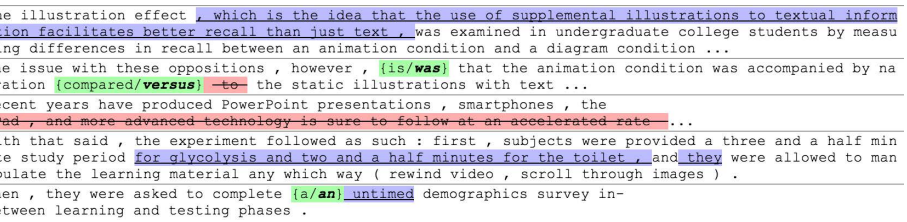


Fig. 1. A screenshot of the detailed drafts comparison. Deleted words are red (strikethrough), added words are purple (underlined) and modified words are green ({old word/***new word***}).

Moreover, to provide a sense of amount of change between drafts, we define a measure to calculate the change based on number of added sentences, deleted sentences and modified words in each sentence:

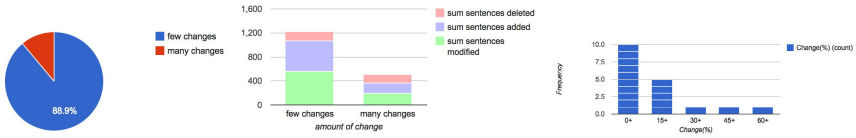
$$Change(\%) = \frac{A + D + \sum_i^M \frac{m_i}{|L_i|}}{A + D + M + U} \times 100$$

where A, D and M are total number of added, deleted and modified sentences respectively. U is number of unmodified shared sentences between two drafts. Furthermore, L_i is the total number of words in sentence i and m_i is the number of modified words in sentence i. Intuitively, when number of added or deleted sentences increases, the percentile of change also increases. Also, if the number of modified words in a sentence increases, the change in that sentence increases.

Figure 2 shows a part of summarized comparison reports for each student in a class. In this set of experiments, the *change* threshold is set 50%, i.e. *change* more than 50% is categorized as "many changes" and less than 50% as "few changes". As can be seen the higher number of added, deleted and modified

Pseudo name	Change (%)	Change category	Total sentences draft 1	Total sentences draft 2	sentences modified	sentences added	sentences deleted
9f190d15	62%	many changes	245	338	97	123	30
Tor	57%	many changes	352	277	100	40	115
yuyuac	25%	few changes	417	497	65	85	5
snowflakes	16%	few changes	219	202	14	6	23
Lord Jenkins	11%	few changes	314	313	17	14	15
H0pe1234	7%	few changes	474	489	27	20	5
happyfeet	2%	few changes	565	569	26	5	1

Fig. 2. A screenshot of a sample class comparison report



(a) Pie chart (b) Stacked column chart (c) Histogram

Fig. 3. Graphs of comparison reports for a class based on changes in sentences

sentences results in a higher *change* score. We also depict the statistics of changes in a class as graphs to help instructors better understand whether students are benefiting from peer review assignments for their courses. Figure 3 shows the graphs that are provided to teachers.

3 Pilot Study

In order to evaluate the utility of our tool for instructors, we ran a pilot study for 4 classes (two psychology, one computer science and one physics). We asked the instructors' feedback focusing specially on these questions:

1. What patterns did you easily notice about how much your students revised their documents?
2. What was hard to read or confusing in the currently provided information?
3. How, if at all, would this kind of data influence your teaching?
4. What additional information would you like about your students' revision behaviors?

Generally, instructors agreed that "the reports are good summaries of the changes in the drafts and the color coding makes them intuitive to glance through". On the other hand, they all mentioned that "the detailed reports of draft comparisons are also overwhelming and not so exciting on their own" (such as Figure 1). So, in order to "make for easier reading of patterns", they suggested to give more control to the teachers over showing the changes. For example, by turning on and off displaying different change types or by displaying only the modified text and "hovering over it pops-up the old text". Two instructors were not sure that the reports will influence their teaching, since the changes were too much. But one instructor mentioned that "it can help with grading - facilitate quicker recognition of how students are responding to feedback." One instructor was also positive about the charts and thought that "all three charts

will be very useful for getting a sense of what the students are doing with the revisions." Finally, they all made some suggestions to improve the reports, for instance one of the shared suggestions was to "tie localized feedback/revision suggestions to implemented revisions". In future, we are planning to apply these suggestions and deploy this tool as a component of a real peer review system.

Acknowledgement. This research is supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A120370 to the University of Pittsburgh. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute or the U.S. Department of Education.

References

1. Compare Suite, `http://www.comparesuite.com/` (accessed January 20, 2014)
2. Bourdaillet, J., Ganascia, J.-G.: Medite: A unilingual textual aligner. In: Salakoski, T., Ginter, F., Pyysalo, S., Pahikkala, T. (eds.) FinTAL 2006. LNCS (LNAI), vol. 4139, pp. 458–469. Springer, Heidelberg (2006)
3. Lardilleux, A., Fleury, S., Cislaru, G.: Allongos: Longitudinal alignment for the genetic study of writers drafts. In: Gelbukh, A. (ed.) CICLing 2013, Part II. LNCS, vol. 7817, pp. 537–548. Springer, Heidelberg (2013)

Example-Based Problem Solving Support Using Concept Analysis of Programming Content

Roya Hosseini and Peter Brusilovsky

University of Pittsburgh - Pittsburgh, PA
{roh38,peterb}@pitt.edu

Abstract. This paper presents two different approaches to example-based problem solving support in the domain of programming based on concept analysis of the learning content. The goal of these approaches is to offer students a set of most relevant remedial examples when they have trouble solving a problem. The paper reviews earlier work and introduces a global and a local approach for selecting examples that are similar to the problem in terms of concept coverage and structure of the content, respectively. It also reports results of a lab study conducted to explore the effectiveness of each approach.

Keywords: concept-based similarity, problem solving support, remediation.

1 Introduction

Example-based problem solving is one of the efficient approaches used by Intelligent Tutoring Systems (ITSs) in the programming domain [1]. In this approach, when the student has trouble solving a problem, the system tries to find the relevant examples which might be helpful to solve the problem. The approach has been used, for example, in ELM-ART ITS for LISP programing [2]. While known to be efficient, this approach remains one of the least explored since the original LISP research was based on advanced episodic learner modeling which is difficult to build for other programming domains.

Our goal is to create a different version of the example-based problem solving support for Java programming which is generalizable in multiple different programming languages without too many effort that is required for advance analysis of content in a system like ELM-ART. The main innovation is in analyzing domain concepts related to programming problems and examples and using the underlying concept structure to find similarity between examples and problems. The similarity can be obtained using number of complicated approaches, but our first challenge was to choose between global and local similarity approaches. A global concept-based similarity considers whether two sets of concepts are more or less similar as a whole and in its simple form could be identified by cosine similarity of vectors of concepts. A local concept-based similarity considers similarity on the structure level where detailed level of similarity can be identified by structure of blocks and adjacent concepts.

To explore and compare these two approaches, we developed a specialized concept analysis tool, JavaParser, which can extract not only the list of concepts but also concept structure [3]. The parser provides a fine-grained level of indexing per line of code which helps identifying blocks of code that have sets of adjacent concepts.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 683–685, 2014.

The parser helped us index a considerable volume of Java programming problems and examples and we could then start a study comparing the approaches. In the present work, we introduce global and local concept-based approaches for finding similar examples for Java problems and present preliminary results of a lab study that aims to compare the effectiveness of these approaches.

2 Method

We propose global and local concept-based approaches for example-based problem solving. The conventional global concept-based similarity approach is based on cosine similarity with TF-IDF weighting for vectors of concepts. The local concept-based similarity approach is based on selecting sets of examples that have the closest blocks of code to the question that the user failed in. The main idea of this approach is to build subtrees of concepts that have appeared together as blocks in each of the contents. As a result, each subset of concepts that are either in the same line or in the same block, will be merged together to create a subtree for the content. Having created the subtrees, we can find the similarity of a question and example by comparing their corresponding subtrees. Several methods have been suggested to compare trees, among which Tree Edit Distance (TED) is quite well known and has been widely used in other studies for similar purposes [4]. The distance between the question and the example is obtained using Eq. (1):

$$WD_{Q,E} = \sum_{q=1}^{N} Arg\min_{e \in M}(TED(s_q, s_e) \times \frac{1}{w_{Qe}}) + \sum_{e=1}^{M} Arg\min_{q \in N}(TED(s_e, s_q) \times \frac{1}{w_{Eq}}) \quad (1)$$

where WD_{QE} is the weighted distance between question Q and example E; N and M represent the total number of the subtrees in Q and E, respectively; TED is the Tree Edit Distance between the given subtrees s_q and s_e; W_{Qe} is the sum of Term Frequency–Inverse Document Frequency (TF-IDF) values of the concepts in the subtree s_e for the question Q; Similarly, W_{Eq} is the sum of TF-IDF values of the concepts in the subtree s_q for the example E. Finally, the weighted distance WD_{QE} is used in Eq. (2) to determine the similarity between example E and question Q:

$$Sim = \frac{1}{e^{\alpha WD_{Q,E}}} \quad (2)$$

where Sim is the local concept-based similarity between the example E and question Q; and α is a coefficient for the exponential function which is set to 0.01 in the present study. Finally, the top five examples with the highest similarity value can be selected for presenting to the student as remedial support.

3 Preliminary Evaluation

We conducted a lab study to investigate the effectiveness of local and global concept-based similarity approaches. We hypothesized that the local approach might outperform

the global one since it considers the closeness of chunks in determining the similarity in a somewhat similar way to ELM-PE that was known as a very successful example recommendation approach. The study started in January 2014 and is ongoing. To date, we have had 12 subjects. The learning materials are organized into 6 Java topics, 83 annotated Java examples, and 24 parametric Java questions. Subjects were assigned to three topics based on their pretest scores and had to solve 4 questions in each of those topics. We selected relevant examples for the questions and asked the student to rate their helpfulness in two different conditions: one at the time that student's answer to question was incorrect and one after the student finished solving a question. The rating ranges from 0 representing 'Not helpful at all' to 3 representing 'Very helpful'. An analysis of the results showed that the average rating of students for the examples selected by the global concept-based approach (1.95) was higher than the local concept-based approach (1.49). Contrary to what we expected, closeness of the concepts did not help select more relevant examples. However, for a much more detailed discussion of results, we need to look into different conditions under which each approach performs better.

4 Conclusion

We proposed global and local concept-based approaches that provide remedial recommendations of examples for Java programming problem solving. These approaches can be leveraged to address the problem solving support in different programming languages with little effort. The result of the ongoing lab study showed that global similarity is almost always helpful for students while local similarity is not. The collected rating data can be useful for cross validation of multiple numbers of similarity approaches and this data is open for other researchers as well. For future work, we would like to perform detailed analysis on the results and also explore whether the approaches can be improved by taking into account the user knowledge.

References

1. Brusilovsky, P., Peylo, C.: Adaptive and intelligent Web-based educational systems. International Journal of Artificial Intelligence in Education 13(2-4), 159–172 (2003)
2. Weber, G.: Individual selection of examples in an intelligent learning environment. Journal of Artificial Intelligence in Education 7(1), 3–31 (1996)
3. Hosseini, R., Brusilovsky, P.: JavaParser: A Fine-Grain Concept Indexing Tool for Java Problems. In: The First Workshop on AI-supported Education for Computer Science (AIEDCS 2013), pp. 60–63 (2013)
4. Zhang, K., Shasha, D.: Simple fast algorithms for the editing distance between trees and related problems. SIAM Journal on Computing 18(6), 1245–1262 (1989)

Clustering Constructed Responses for Formative Assessment in Comprehension SEEDING

Frank Paiva and Rodney D. Nielsen

University of North Texas
frankpaiva@my.unt.edu, rodney.nielsen@unt.edu

Abstract. In this paper we discuss the Comprehension SEEDING system and how it enhances the classroom discussion. SEEDING attempts to accomplish this by providing real-time formative assessment of student responses. Teachers can pose questions using the SEEDING system, students submit typed responses, and the system automatically groups the responses according to their semantic similarity. These components equip teachers with the necessary tools to engage the classroom in a broad and enhanced discussion.

1 Introduction

In a typical classroom, teachers pose questions both to assess student understanding and also to facilitate learning. Students learn as a result of engaging with the material through shared discourse (Larson, 2000). However, this is difficult to achieve as teachers typically can only involve a few students in a discussion. To address this problem, classroom response technologies, such as clickers, provide a means to allow all students to respond to multiple choice questions.

Clickers are a classroom response system in which students respond to questions that are projected onto a screen in the classroom. Previous work on clickers has shown that they can be beneficial for enhancing student learning (Duncan, 2006; Keller et al., 2007; Penuel, Boscardin, Masyn, & Crawford, 2006; Siau, Nah, Siau, Sheng, & Nah, 2006). However, there are limitations that could explain why small-scale efficacy tests for the use of the technology have seen mixed results (Bunce et al., 2006; Carnaghan & Webb, 2007; Duggan et al., 2007). In order for teachers to take advantage of clickers, they are limited to multiple choice questions. This can make it difficult to assess deep knowledge in a closed-response question format (Campbell, 1999). Even with meaningful distractors, multiple-choice questions only require students to recognize the correct response. According to the ICAP (Interactive, Constructive, Active, Passive) framework (Chi, 2009), constructing answers to free-response questions is a more cognitively engaging task and leads to larger learning gains.

2 Comprehension SEEDING

The Comprehension SEEDING system allows teachers to pose free-response questions. Students can answer these questions via tablets. While students compose their responses, the system provides a real-time analysis of the student

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 686–688, 2014.

responses. Once SEEDING receives a few student responses, it automatically groups the student responses into four clusters. Teachers have the option to view and share each student response with the class. However, this can be time consuming and might only address a few possible misconceptions about a question. Using the four clusters, teachers can quickly determine the current overall status of the classroom′s understanding of the question.

SEEDING is grounded in results from three key areas of cognitive and learning sciences research: (1) student self explanation, (2) formative assessment with classroom engagement and discourse, and (3) educational question-asking practices. The Comprehension SEEDING system is divided into three distinct but related components that work together to create an enhanced learning environment for both teachers and students. The enhanced discussion (ED) component and its theoretical advantages are described in the following section.

3 Enhanced Discussion

As students respond to a question, SEEDING performs analysis and provides teachers real-time feedback on the students' understanding. To cluster student responses, we need an understanding of each student′s response and its entailment relationship to the question′s reference answer. We dont simply want to label responses as correct or incorrect. Instead, if a response is not correct, we want to understand and identify specifically where the error or misconception exists. To achieve this level of semantic analysis, we break down the reference answer and all responses into their fine-grained semantic facets following (Nielsen et. al, 2008). An analysis of all of these semantic facets is used to generate the feature vectors used by the clustering algorithm, as discussed below.

Feature vectors are comprised of four sets of features, each of which is assigned a total weighting or importance. The sum of the weights over the four sets of features is 1.0. The first set of features is based on the subset of semantic facets found in the reference answer that are not also found in the question. These features were given a weight of 0.45. The second set of features, which has a weight of 0.225, is based on the remaining facets found in the reference answer. The third set of features, with a weight of 0.1, is based on the facets found only in the question. The final set of features, comprising the remaining weight of 0.225, is based on any additional facets that occur in multiple student responses. In future work, the weights of each set of features will be learned based on training data. Student responses are converted into feature vectors according to which facets in these four groups are entailed by the response.

SEEDING clusters responses automatically as soon as the percent of the class that has submitted responses passes a threshold. Each time, the system decomposes all the current responses into facet-based representations. This will allow the system to compare the student facets with the reference answer facets when clustering the student responses. For example, if just three students submit a response that addresses all of the facets in the reference answer, those students form a cluster. Similarly, this holds true for responses that fail to address certain facets

from the reference answer. At the time of clustering, all responses are turned into vectors. Four randomly selected response vectors are assigned as the initial cluster centroids. We iterate over each student response vector, calculate its distance from each cluster centroid, and assign the response to the cluster whose centroid is closest. After each iteration, the cluster's centroid is recalculated by averaging the response vectors assigned to it. These two steps, assigning responses to the closest cluster and recomputing the cluster centroids, are repeated for 10 iterations or until convergence, when the clusters stop changing.

4 Conclusion

In this paper, we presented the Comprehension SEEDING system, new classroom engagement technology designed to improve classroom discussion by increasing student engagement and providing immediate formative feedback to teachers. We focused on SEEDING′s enhanced discussion component and in particular, on the novel use of clustering and clustering algorithm, which provide a key foundation to facilitate formative assessment and enhanced classroom discussion.

Acknowledgements. This research was supported by the Institute of Education Sciences, US Dept. of Ed; Grant R305A120808 to UNT. The opinions expressed are those of the authors.

References

1. Bunce, D.M., VandenPlas, J.R., Havanki, K.: Comparing the effectiveness of student achievement of a student response system versus online WebCT quizzes. Journal of Chemisty Education 83(3), 488–493 (2006)
2. Carnaghan, C., Webb, A.: Investigating the Effects of Group Response Systems on Student Satisfaction, Learning, and Engagement in Accounting Education. Issues in Accounting Education 22(3), 391–409 (2007)
3. Chi, M.T.H.: Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities. Topics in Cognitive Science 1(1), 73–105 (2009)
4. Duggan, P.M., Palmer, E., Devitt, P.: Electronic voting to encourage interactive lectures: A randomised trial. BMC Medical Education 7(25) (2007)
5. Duncan, D.: Clickers: A New Teaching Aid with Exceptional Promise. Astronomy Education Review 5(1), 70 (2006)
6. Keller, C., Finkelstein, N., Perkins, K., Pollock, S., Turpen, C., Dubson, M., Hsu, L., et al.: Research-based Practices For Effective Clicker Use. In: AIP Conference Proceedings, pp. 128–131 (2007)
7. Larson, B.: Classroom discussion: a method of instruction and a curriculum outcome. Teaching and Teacher Education 16(5-6), 661–677 (2000)
8. Nielsen, R., Ward, W., Martin, J., Palmer, M.: Extracting a Representation from Text for Semantic Analysis. In: Proc. ACL-HLT (2008)
9. Penuel, W.R., Boscardin, C.K., Masyn, K., Crawford, V.M.: Teaching with student response systems in elementary and secondary education settings: A survey study. Educational Technology Research and Development 55(4), 315–346 (2006)
10. Siau, K., Nah, F.F., Siau, K., Sheng, H., Nah, F.F.: Use of a Classroom Response System to Enhance Classroom Interactivity Use of a Classroom Response System to Enhance Classroom Interactivity (2006)

Negotiation Driven Learning: A New Perspective of Learning Using Negotiation

Raja M. Suleman, Riichiro Mizoguchi, and Mitsuru Ikeda

School of Knowledge Science,
Japan Advanced Institute of Science & Technology, Nomi, Ishikawa, Japan
{suleman,mizo,ikeda}@jaist.ac.jp
http://www.jaist.ac.jp

Abstract. Negotiation mechanism used in the current implementations of Open Learner Models is mostly positional based and provides minimal support for learners to understand why their beliefs contradict with that of the system. This study aims at proposing a new paradigm of learning that uses negotiation coupled with targeted responses to motivate a learner and enhance their metacognitive skills along with their cognitive skills.

Keywords: Negotiation, Metacognition, negotiation-driven learning, inter-est-based negotiation, learner motivation.

1 Introduction

In recent years much research has been done in the field of Intelligent Tutoring Systems (ITS) to support and promote independent, self-regulated learning. Much of this research has primarily focused on generation and visualization of Learner Models (LM). Open Learner Models (OLM) aim at enhancing both cognitive and metacognitive skills of a learner through guided content, externalization, scaffolding and negotiation. However, negotiation has been underutilized in the current OLMs. Negotiating or debating with others allows us to identify alternative perspectives [1][2]. According to the Constructivist Learning Theory "learning is a process of construction of knowledge through dialogues"[1]. Therefore in this study we will focus on "fully-negotiated" LMs [3] and propose the paradigm of Negotiation-Driven Learning (NDL) with the aspiration to enhance the role of negotiation as a problem-understanding technique and use it to promote metacognitive activity and enhance learning.

2 Problem Description

The negotiation aspect of the current implementations of OLMs is aimed at solving the problem of the conflict between the learner's beliefs and that of the system [4]. OLM relies upon the externalization of a learner's knowledge to promote metacognitive skills. Negotiation is generally related with the occurrence

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 689–691, 2014.

and resolution of conflict. Position-Based Negotiation (PBN) is employed which considers both the learner and the system to firmly commit to their positions. Only one party finally reaches their position or a compromised position is accepted by both parties [5].

A negotiation session is initiated by the system when there is a conflict between the learner's and system's sets of beliefs. A learner is also allowed to start a negotiation session when they do not agree with their LM. The learner is allowed to defend and retain their beliefs if they can justify them [3]. In order to justify themself, the learner has to provide evidence or justification. The system asks the learner a set of questions to prove the validity of their claim. These are closed-ended, directed questions that evaluate the knowledge of the learner about a certain topic. What is "tested" is not the "motivation" behind their claim but the learner's knowledge. This confines the scope of negotiation as more of a "problem solving" technique rather than a "problem understanding" technique.

3 Proposed Solution: Negotiation Driven Learning

Almost all implementations of fully negotiated LMs follow an ideology which constrains negotiation to just "negotiating to solve problems/conflict". In this context, negotiation is used as a supplement for "teaching" whereas in our point of view negotiation provides an excellent opportunity for "learning".

When a learner negotiates their LM with the system, they are actively involved in a dialog, intrinsically motivated to justify their claim, hence more likely to conceive new knowledge. This provides an excellent opportunity to indulge the learner in metacognitive-guided learning, where they build knowledge by actively using and enhancing their cognitive and metacognitive skills. NDL aims to take advantage of this precise opportunity by understanding and challenging the motivational state of the learner, indulging the learner in an active dialog which promotes self-regulated learning, and provides a chance to the learner to understand "what and how" they learn. Contrary to the concept of PBN, Interest-Based Negotiation (IBN) [1] is a process that aims at exploring underlying interests of the parties rather than their negotiating positions. Therefore the concept of IBN seems much more appropriate to be incorporated in NDL. The concept of IBN can allow us to use negotiation as means to understand the reasons behind the beliefs of a learner. IBN can play a vital role in NDL, since in NDL we are concerned with motivating the learner by trying to understand their reason for holding a particular belief which in turn can help identify why such beliefs are held and how can a learner be persuaded to change them.

3.1 Proposed System

The proposed system would generate a Behavioral Model (BM) of the learner as they interact with the system. The BM will include information about the interactions of the learner with the system; their interest in their respective LM,

their enthusiasm in discussing their LM, their help-seeking pattern and their confidence in their abilities. The behavioral model will be continuously updated through the Session Manager (SM) which would record interactions of the learner with the system in real-time.

Once the baseline BM of the learner is generated it will be used by the Automated Negotiation Agent (ANA) to understand the motivational state the learner is in and use this information to select the best suited negotiation strategy from the Plan Base (PB) to maximize learning. For example; if the BM of the learner shows that they are confident in their abilities but avoid discussing their LM with the system, the ANA will try to prompt the learner's attention towards their LM. Using "challenge" as a motivational trigger, the system will try to entice the learner into discussing why their beliefs differ from that of the system in an open-ended discussion. The goal would be to intrigue the learner into discussing their LM such that both the system and the learner understand why they hold certain beliefs and how they can overcome these differences.

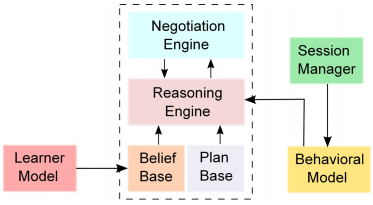


Fig. 1. Proposed Interest-Based Automated Negotiation Agent

4 Concluding Remarks

NDL follows the notion that learning is maximized by participation in the learning process and negotiation provides an excellent opportunity to challenge the learners which promotes metacognitive skills by motivating them to think more objectively about their learning. Although the research on NDL is in its early stages, we believe that the paradigm of NDL holds great potential as it opens up new perspectives of learning by using automated IBN to challenge and intrinsically motivate learners through discussions.

References

1. Miao, Y.: An intelligent tutoring system using interest based negotiation. In: Control, Automation, Robotics and Vision 2008 (2008)
2. Veerman, A.L.: Computer-supported collaborative learning through argumentation. Doctoral Dissertation. Utrecht University, the Netherlands (2000)
3. Bull, S., Vatrapu, R.: Negotiated Learner Models for Today. In: ICCE 2012 (2012)
4. Bull, S., Pain, H.: "Did I Say What I Think I Said, And Do You Agree With Me?": Inspecting and Questioning the Student Model. In: Greer, J. (ed.) AIED 1995, pp. 501–508. AACE, Char-lottesville (1995)
5. Pasquier, P., Hollands, R., Dignum, F., Rahwan, I., Sonenberg, L.: An empirical study of interest-based negotiation. In: Proceedings of the Ninth International Conference on Electronic Commerce, Minneapolis, MN, USA, August 19-22 (2007)

Phenomenography of Student Perceptions of an Online Metacognitive Tool

Aaron Thomas

University of Florida

Abstract. The following phenomenographic study aims to explore student perceptions of an adaptive learning system employed within a large undergraduate cohort of chemistry students during the Fall 2013 term. In particular, the purpose was to determine what student perceptions were concerning the adaptive system and how these perceptions influenced the students metacognitive and self-regulatory behaviors.

Keywords: metacognition phenomenography blended learning.

1 Introduction

The present study explores how these perceptions of the metacognitive accuracy of the intelligent tutoring system impact student self- regulation and behavior within a blended course at a large tier-one university within the southeastern United States. .

2 Knowillage

The adaptive learning tool under consideration is the *Knowillage* adaptive learning system. The primary purpose of the online tool was to prepare individualized learning paths for each topic, e.g. periodicity or thermochemistry, based upon pre-test questions correlated to specific web based text pages concerning the topic. According to the software developers, *Knowillage* creates a learning path of individualized readings to address a student's deficits without wasting a student's time upon material already mastered as measured by the topic pre-test.

3 Theoretical Framework

Self-regulated learning theory provided a useful mechanism to describe "the process whereby learners personally activate and sustain cognitions, affects, and behaviors" [1, pg. 1]. Given that most learners exhibit poor metacognitive skills [2], the *Knowillage* system does not rely upon the learner's metacognitive skill to determine a useful learning path. The *Knowillage* system decides what instructional text or instructional materials are most relevant to the unique learning deficits of the learner as measured by a multiple choice pretest.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 692–694, 2014.

4 Research Question

From the preceding theoretical framework and unique educational conditions, the following research question arose:

Q1: How does variation in learner perception of the metacognitive tool *Knowillage* influence self-regulation of learning and gaming of the system?

5 Methods

In order to assess learner conceptualizations of Knowillage, a phenomeno- graphic methodology was employed. Phenomenography is both a theoretical perspec- tive and methodology taken by educational researchers to explore learner perceptions of phenomena or "people's conception of reality" [3, pg. 178]. In the present study, however, the large group of 300 engineering students precluded the possibility of deep interviews with the entire cohort so a survey was distributed to all 300 participants. For the present study, 91 of the 300 participants responded to the survey for a response rate of 30%.

6 Results

Q2: How does variation in learner perception of the metacognitive tool Knowillage influence self-regulation of learning and gaming of the system?

Review of the survey responses indicated that students were not using the *Knowillage* system in a manner anticipated by the system. Learners were misusing or rather using the tool by randomly clicking on answers to pre-tests and summative tests in a manner unexpected by the software developers. Learners were also regulating the context of their learning by enhancing their learning environment through the use of the optional hardcover textbook. In addition, many students claimed to have paid a private tutoring company to help them achieve success in the course.

52% reported that they read the textbook more because of *Knowillage*
47% admitted to maladaptive behaviors such as randomly answering pre-tests
45% reported that *Knowillage* did not provide the right amount of reading
55% reported that the practice problems were not related to the subject at hand
32% enlisted a private tutoring company to find success in the course

7 Discussion

It appears from the evidence that the students' perceptions of the tool and its legitimacy differed to a great degree with many students questioning the accuracy of the *Knowillage* system. The present study suggests that when learners are inserted into highly structured learning environments in an ecological setting, learners will engage in the regulation of the learning context and find ways to shape the learning environment into

a space that can meet their psychological self-regulatory needs. The present study has raised a number of future questions as to why were some students more suspicious of the *Knowillage* technology than others? Does this suspicion of technology correlate with greater self-regulatory behaviors, prior knowledge, or something else? At what point do students game the system as a result of an analysis of the diagnostic performance of a tool such as *Knowillage*? These questions remain to be explored in future work.

References

1. Zimmerman, B.J., Schunk, D.H.: Handbook of self-regulation of learning and performance. Taylor & Francis (2011a)
2. Marton, F.: Phenomenography—describing conceptions of the world around us. Instructional Science 10(2), 177–200 (1981)
3. Pashler, H., Bain, P.M., Bottge, B.A., Graesser, A., Koedinger, K., McDan-iel, M., Metcalfe, J.: Organizing instruction and study to improve student learning. IES practice guide. NCER 2007-2004. National Center for Educa-tion Research (2007)

Toward Sense Making with Grounded Feedback

Eliane Stampfer Wiese and Kenneth R. Koedinger

Human-Computer Interaction Institute, 5000 Forbes Avenue
Pittsburgh, PA 15213 USA
{stampfer,krk} @cs.cmu.edu

Abstract. Grounded feedback aims to facilitate sense making by reflecting students' symbolic input in a linked concrete representation that is easier to reason with. Initial studies led to explorations of what prior knowledge is necessary to support that reasoning. Specifically, we tested if it is obvious to students that a sum is larger than its two positive addends. It is not! Thus, concrete representations for sense making may fail because students lack prerequisite knowledge we may assume they have. More generally, these results suggest that skilled qualitative reasoning may often come after, not before, quantitative reasoning.

Keywords: grounded feedback, fraction addition, graphical representations.

Effective instruction elicits students' prior knowledge and facilitates useful connections between what students already know and what they are learning. To that end, my work presents and investigates grounded feedback, in which student inputs (e.g., 5/12) are reflected in a more familiar representation that is easier to reason with (e.g., a fraction bar). Prior work shows experimental support for such feedback over right/wrong immediate feedback (e.g., [1]), but does not provide a full theoretical characterization of grounded feedback or design recommendations for its implementation. My proposed work on grounded feedback will: continue to examine its effects with empirical, controlled classroom studies; explicitly define it and situate it in a theoretical framework; build a theoretical model of how students use grounded feedback to make sense of new information; and delineate design recommendations for its implementation. This paper focuses on the design recommendations.

Our work on grounded feedback examines middle school students learning fraction addition (Fig. 1 shows a tutor example). The grounded feedback consists of rectangular fraction bars that reflect the symbolic values that students enter. This common representation (e.g., [2]) is intended to elicit students' prior knowledge of magnitude and make salient important fractions concepts (e.g., one cannot add fractions by simply adding the numerators and denominators). An experiment with 5th graders showed student learning with grounded feedback, and some benefits over immediate right/wrong feedback [3]. However, that study also revealed students' difficulty interpreting the feedback: they often indicated that a problem was solved even when their proposed sum did not line up with the combined length of the given addends. This finding suggests that 1) the students did not realize that two addends equal their sum and/or 2) some aspect of the representation blocked students' use of that knowledge.

S. Trausan-Matu et al. (Eds.): ITS 2014, LNCS 8474, pp. 695–697, 2014.

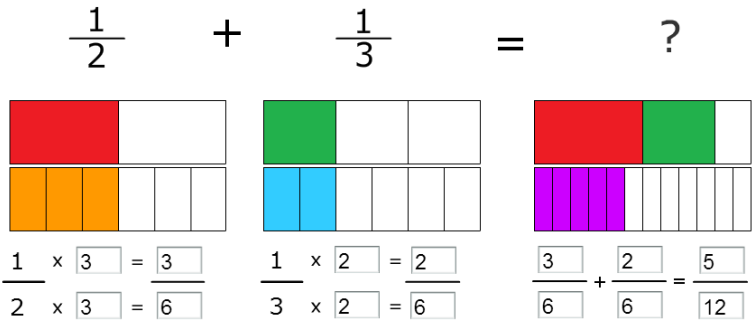


Fig. 1. Fraction Addition Tutor showing correct conversion, incorrect sum (correct sum is 5/6). Top row of fractions and red and green fraction bars are given, second row of bars dynamically shows students' inputs as they are typed in the boxes at the bottom.

A follow-up study at the same school supported hypothesis 2 [4]. 5th grade students in that study saw fraction addition equations on paper, represented with rectangles, symbols, or both. Half of the equations were correct, and half proposed a sum that was smaller than one of the addends. Students indicated if the equations were true or false. The students could tell if fraction addition equations were correct or not when the addends and sum were presented as rectangles, but performance decreased when symbolic fractions were shown with the rectangles. The fraction symbols appear to impede students' recollection or use of the addend-sum relationship.

The current study investigates students' understanding of the addend-sum relationship. 160 fifth-grade students were given 20 minutes for a 34-question test, administered by their teacher. Question order was determined randomly, and half of the test forms were printed in reverse order. This analysis is of the five questions which tested sense making with addition. Problems presented a correct addition equation with positive numbers, with a note that the equation was correct, and asked if the sum was bigger than the two addends (options: True, False, and Can't tell from the information given). Students were randomly assigned to test forms that presented this question either with whole numbers, decimals, fractions, or variables (see Fig. 2). Means for each number type reveal that students do not have a general understanding of the addend-sum relationship (percent correct for: whole numbers, 79; decimals, 75; fractions, 61; and variables, 51). An ANOVA with the total addend-sum score (dependent measure) and test form (fixed factor) showed a significant effect of number type ($p<.01$), indicating that students' recognition of the addend-sum relationship is highly dependent on the type of number in the equation.

These results indicate that many students do not have a general conception of the addend-sum relationship for positive numbers - "addition makes bigger". Because whole numbers and decimals can be compared directly, students can solve those problems by comparing the numbers (e.g., .7 > .083) without knowing the addend-sum relationship. It is harder to directly compare unlike-denominator fractions, and impossible for variables. Students' performance with fractions and variables reveals a lack of understanding that helps explain why students had trouble with the grounded feedback. Students may notice that the purple bar in Fig. 1 is smaller than the red-and-green bar, but if they do not recognize that two addends must equal their sum, they may not interpret the rectangles as indicating that 5/12 is incorrect. A corollary is, for positive numbers, that the

sum is always larger than each addend. Knowledge of this relationship is not fully in place even for whole numbers. The missing understanding of the addend-sum relationship may partially explain the results from [4]. Given students' difficulty in recognizing the addend-sum relationship with fractions, perhaps those symbols distracted students from using that relationship. Additionally, this work suggests that in math, qualitative understanding might come after, not before, quantitative expertise (e.g., students may not understand the addend-sum relationship until they have extensive practice adding numbers of many types).

This body of work highlights the difficulty in designing feedback that elicits students' prior knowledge toward greater sense making (and thus greater understanding). Grounded feedback is intended to help students evaluate their own work by presenting a feedback representation that students can easily reason with. For example, it is easier for students to compare two fractions when they are represented as rectangles instead of symbols. Still, students must take the outcome of that comparison and reason with it, and if the concepts or skills needed for such reasoning are not fully in place, students may not be able to make full use of the grounded feedback. On the other hand, students' work with grounded feedback may reveal gaps in their prior knowledge that would not be evident from their interaction with immediate right/wrong feedback. If grounded feedback tutors can detect what type of prior knowledge students are missing, embedded activities could provide targeted instruction on those concepts, which students could then practice as part of the normal tutor.

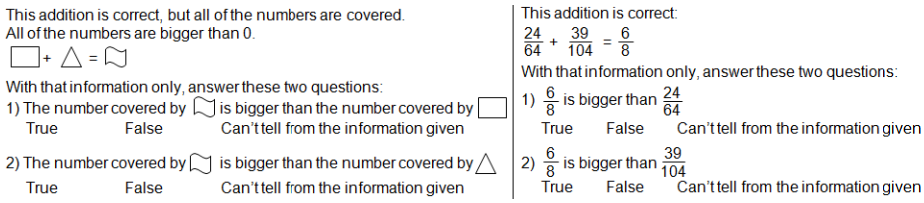


Fig. 2. Example addend-sum questions, with variables (left) and fractions (right)

Acknowledgements. This work was supported in part by the PSLC through NSF award SBE-0836012, and by Graduate Training Grant #R305B090023 awarded to CMU by the Department of Education.

References

1. Nathan, M.J.: Knowledge and Situational Feedback in a Learning Environment for Algebra Story Problem Solving. Interactive Learning Environments 5, 135–159 (1998)
2. Suh, J., Moyer, P.S., Heo, H.-J.: Examining Technology Uses in the Classroom: Developing Fraction Sense Using Virtual Manipulative Concept Tutorials. J. of Interactive Online Learning 3(4), 1–21 (2005)
3. Stampfer, E., Koedinger, K.R.: Tradeoffs between Immediate and Future Learning. European Association for Learning and Instruction (2012)
4. Stampfer, E., Koedinger, K.R.: When seeing isn't believing: Influences of prior conceptions and misconceptions. Cognitive Science Society (2013)

Author Index