Argument Based Machine Learning Applied to Law

Martin Možina[1], Jure Žabkar[1], Trevor Bench-Capon[2], and Ivan Bratko[1]

[1] Faculty of Computer and Information Science, University of Ljubljana, Slovenia
{martin.mozina,jure.zabkar,ivan.bratko}@fri.uni-lj.si
[2] Department of Computer Science, The University of Liverpool, Liverpool, UK
T.J.M.Bench-Capon@csc.liv.ac.uk

Abstract. In this paper we discuss the application of a new machine learning approach - Argument Based Machine Learning - to the legal domain. An experiment using a dataset which has also been used in previous experiments with other learning techniques is described, and comparison with previous experiments made. We also tested this method for its robustness to noise in learning data. Argumentation based machine learning is particularly suited to the legal domain as it makes use of the justifications of decisions which are available. Importantly, where a large number of decided cases are available, it provides a way of identifying which need to be considered. Using this technique, only decisions which will have an influence on the rules being learned are examined.

1 Introduction

In many areas of law - especially administrative law - many thousands of cases are routinely decided. Often these cases involve the exercise of some discretion, or involve some degree of operationalisation of the concepts used in the legislation, so that the decision can be based on ascertainable facts rather than the vaguer terms in which the legislation may be couched: for example aged over 65 rather than elderly [1]. We would generally wish to assume that such discretion or operationalisation is consistent so that like cases are decided in a like manner, that some kind of rule is being followed. Is there a way of deciding what the rule being followed is from an automated consideration of the data?

Such a question has relevance to a number of interesting and important issues:

- If there is well defined legislation which defines what the rule should be we may wish to ensure that the rule is being followed;
- If the domain is a discretionary one, we may wish to discover the rule itself;
- Similarly we may wish to discover the way in which the stated conditions have been operationalised;
- Some people have argued that the rule followed in practice is different from the rule which exists in theory (or which might be elicited from experts). For example Edwards [2] suggested that some areas of law exhibited a systematic bias. Such conjectures could be informed and justified were we able to discover the "real" rule from a database recording the actual practice.

If therefore we had a reliable technique to extract rules explaining the data in a field of law, it would have many interesting uses.

One problem with many experiments to explore the efficacy of techniques designed to extract knowledge from data is that they use data for which the relationships present are not known at the outset. As a result, what has been discovered, and what has been missed, cannot be established definitively. Often accuracy of classification is taken to validate the knowledge extracted, but this test is, as will be discussed below, rather one-sided. In the work reported here we will use a specially constructed data set, the properties of which are known, and which is thus able to serve as a measurable test of the technique.

The data we use has been used in several previous AI and Law experiments, reported in [3–5]. The use of this same data set allows for comparison between what can be derived using the various techniques.

The particular technique we will consider in this paper is Argument Based Machine Learning (ABML) [6]. The central idea is to augment a standard machine learning technique to accept, along with data, arguments explaining the classification of a small number of instances. It is hoped that this will improve both the efficiency of the learning process, and the quality of the rules learned. By "quality" we do not only mean statistical accuracy (probability of correct classification), but also comprehensibility: can rules learned be clearly understood and interpreted by a domain expert. The comprehensability of rules is essential in legal applications as such rules enable, besides decisions, also justifications for decisions made. Moreover ABML is particularly suitable to the kind of legal application described above. In law, decisions are typically accompanied by some justification - often very brief in routine cases - which can serve as the requisite argument. Given the volume of decided cases, however, we cannot be sure how many, or which, of these justifications to consider. ABML will point us towards those that need to be looked at.

2 Argument Based Machine Learning

Argument Based Machine Learning [6] is a novel approach to machine learning that draws on some concepts of argumentation. While the standard problem of machine learning from examples is to find a theory that explains given examples, in ABML some of these examples are accompanied by arguments, and the problem of ABML is to find a theory that explains examples using these arguments.

Arguments that support an example are typically reasons for believing why the class of an example is as given, or, especially in law, they can be also seen as the justification for a specific decision. The main motivation for using such arguments lies in two expected advantages:

1. Reasons (arguments) impose constraints over the space of possible hypotheses, thus reducing search complexity;
2. An induced hypothesis should make more sense to an expert as it has to be consistent with given arguments.

Regarding advantage (1) above, the fundamental difficulty of machine learning arises from the explosive combinatorial space of possible hypotheses. Using arguments, the computational complexity associated with search in the hypothesis space can be reduced considerably, and enable faster and more efficient induction of hypotheses. Regarding the other advantage, (2), there are many possible hypotheses that, from the perspective of a machine learning method, explain the given examples sufficiently well. But some of those hypotheses can be unnatural with respect to the current domain knowledge and therefore incomprehensible to domain experts and give little insight into rules being followed. On the other hand, using arguments should lead to hypotheses that explain given examples in similar terms to those used by the expert, and correspond to the actual justifications.

Superficially it may appear that the idea of ABML is similar to explanation based learning (EBL of DeJong and Mooney [7]) or explanation based generalisation (EBG of Mitchell et al. [8]). Both in ABML and EBG, just one or a few specific examples play a central role. But in fact ABML and EBG are completely different due to the following significant difference: EBG assumes that the complete domain theory is already given, and then it uses the given example to compile this given theory into a more efficient form. So in EBG there is no inductive generalisation, and strictly speaking - no learning (the result is not an induced definition of the target concept, but just the optimisation of an already known definition). In contrast to this, in ABML the target concept is not given, but it is induced from data.

Another difference between ABML and EBG is in the choice of learning examples. In EBG, examples are chosen by the trainer in the light of his knowledge of the domain of learning. In contrast to this, ABML accepts a data set of examples that come from observations in the domain. Some of these examples that are supposed to be particularly important are commented by the expert. These important examples may be chosen by the expert, as in EBG, or alternatively, they may be automatically chosen by the system in the light of difficulties encountered in learning the target concept.

2.1 ABCN2 Algorithm

Argument Based CN2 (ABCN2) is a realization of concepts described in the previous section. ABCN2 is an extension of the well-known rule-learning algorithm CN2 of Clark and Niblett [9]. The original CN2 algorithm is augmented so that it takes into account arguments that accompany some of the learning examples. We will start this section with a description of argumented examples in ABCN2, and continue with a detailed description of differences between the CN2 algorithm and its argument based counterpart ABCN2.

Argumented examples in ABCN2. In ABCN2, arguments that the domain expert may attach to selected examples have the form of the conjunction of attribute-value pairs. In this way the expert may state those features that he or she believes are important in the context of this particular example. The fact that such arguments apply to *individual* examples is of practical importance. It makes the expert's task easier: the expert can just state features believed to be important in the particular case, but not necessarily important and valid for the whole domain.

A learning example E in the usual form accepted by CN2 is a pair (A, C), where A is an attribute-value vector, and C is a class value. In addition to such examples, ABCN2 accepts also argumented examples. An argumented example AE is a triple of the form:

$$AE = (A,\ C,\ C \text{ because } Reasons)$$

As usual, A is an attribute-value vector and C is a class value. "C because *Reasons*" specifies reasons for the given class value. We refer to such reasons also as *argument.*

To illustrate the idea of argumented examples, consider a simple learning problem: learning about credit approval. Each example is a customer's credit application together with the manager's decision about credit approval. Each customer has a name and three attributes: PaysRegularly (with possible values "yes" and "no"), Rich (possible values "yes" and "no") and HairColor ("black", "blond", ...). The class is CreditApproved (with possible values "yes" and "no"). Let there be three learning examples shown in Table 1.

Name	PaysRegularly	Rich	HairColor	CreditApproved
Mrs. Brown	no	yes	blond	yes
Mr. Grey	no	no	grey	no
Miss White	yes	no	blond	yes

Table 1. Learning examples for credit approval.

The expert's argument for approving credit to Mrs. Brown can be: Mrs. Brown received credit because she is rich. Similarly, Miss White received credit because she pays regularly. The Brown example would in our syntax be written as:

$$((PaysRegularly = no,\ Rich = yes,\ HairColor = blond),$$
$$CreditApproved = yes,\ CreditApproved = yes \text{ because } Rich = yes)$$

In CN2, rules have the form:

$$\text{IF } Complex \text{ THEN } Class$$

where *Complex* is the conjunction of simple conditions, called *selectors.* For the purpose of this paper, a selector simply specifies the value of an attribute, for example $HairColor = blond$ or a threshold on an attribute value, for example $Salary > 5000$. A rule for our credit approval domain can be:

$$\text{IF } PaysRegularly = no \text{ AND } HairColor = blond \text{ THEN}$$
$$CreditApproved = yes$$

The condition part of the rule is satisfied by the attribute values of Mrs. Brown example, so we say that this rule *covers* this example.

Argumented examples in ABCN2 are like examples in the classical sense, but they can possibly be accompanied by arguments. To take into account arguments in examples, the definition of a rule *covering* an example needs to be refined. In the standard definition, a rule covers an example if the condition part of the rule is true for this example. In argument based rule learning, this definition is modified to:

A rule R *AB-covers* an argumented example E if:

1. All conditions in R are true for E (same as in CN2), and
2. All the reasons of the argument of E are included among conditions of R.

As an illustration of the differences between AB-covering and the usual definition of covering, consider again the Brown example with the argument that she received credit because she is rich. Now consider two rules:

Rule 1: IF $HairColor = blond$ THEN $CreditApproved = yes$.
Rule 2: IF $Rich = yes$ THEN $CreditApproved = yes$.

Both Rule 1 and Rule 2 cover the Brown example. However, Rule 1 does not AB-cover the example whereas Rule 2 does. The following rule, Rule 3, also AB-covers the example:

Rule 3: IF $HairColor = blond$ AND $Rich = yes$ THEN
$CreditApproved = yes$.

Thus ABML gives us a reason, based on the expert's decision processes, to choose between candidate rules.

CN2. The CN2 algorithm [9, 10] consists of a covering algorithm and a search procedure that finds individual rules by performing beam search. The covering algorithm induces a list of rules that cover all the examples in the learning set. Roughly, the covering algorithm starts by finding a rule, then it removes from the set of learning examples those examples that are covered by this rule, and adds the rule to the set of rules. This process is repeated until all the examples are removed.

There are two versions of CN2: one induces ordered list of rules, and the other unordered list of rules. Our algorithm in this paper is based on the second version of CN2. In this case, the covering algorithm consists of two procedures, CN2unordered and CN2ForOneClass. The first procedure iteratively calls CN2ForOneClass for all the classes in the domain, while the second induces rules only for the class given. When removing covered examples, only examples of this class are removed [10]. Essentially, CN2ForOneClass is a covering algorithm that covers the examples of the given class.

ABCN2: Covering algorithm. We augmented CN2 for learning unordered rules with the ability to learn from argumented examples. We call the resulting algorithm ABCN2.

The first requirement for ABML is that an induced hypothesis explains argumented examples using given arguments. In rule learning, this means that for each argumented example, there needs to be at least one rule in the set of induced rules that AB-covers this example. This is achieved simply by replacing covering in original CN2 with AB-covering.

Replacing the "covers" relation in CN2 with "AB-covers" in ABCN2 ensures that both argumented and non-argumented examples are AB-covered. However, in addition to simpy AB-covering all the examples, we would prefer also explaining as many as possible non-argumented examples by arguments given for the argumented examples. Therefore, we propose a change in the covering algorithm, where CN2ForOneClass is changed into ABCN2ForOneClass (see Algorithm 1). The procedure starts by creating an empty list of rules, makes a separate set AES of argumented examples only, and sorts them according to the "quality" (defined later) of the given argument. Then it induces a rule, using procedure ABFind_Best_rule, to cover the first argumented example. ABFind_Best_rule is a modified beam search procedure that can accept examples, an argumented example and a target class, where the resulting rule is guaranteed to AB-cover the given argumented example. This rule is added to the rule set, and the procedure removes from AES argumented examples AB-covered by this rule. The removal of all positive examples is not necessary, as each of the argumented examples differently constrains the search and thus prevents ABCN2 from inducing the same rule again. When all argumented examples are covered, all positive examples AB-covered by rules are removed, and the remaining rules are learned using classical CN2ForOneClass.

Quality of rule or argument is a user-defined measure to estimate the goodness of a rule. Generally, this measure should reflect the accuracy of the rule when classifying new examples, and the generality of the rule (the more cases the rule covers the better). In our implementation, the "quality" of a rule is defined as an estimate of the probability of correct classification of new (not learning) cases by the rule. To estimate the probability of correct classification we use the m-estimate [11] of probability. This measure can be viewed as if assuming a prior, virtual example set of m examples which are distributed proportionally to the number of positive and negative examples in the training set. The quality of an argument is computed as the quality of a rule whose condition part is the argument.

ABCN2: Search Procedure. Algorithm 2 shows AB search procedure. The procedure takes a set of examples to learn from, an argumented example that needs to be AB-covered by the induced rule and a target class. In Algorithm 2 the underlined parts emphasize the differences between the original search procedure in CN2 and AB-search procedure:

```
Procedure ABCN2ForOneClass(Examples ES, TargetClass T)

Let RULE_LIST be an empty list.
Let AES be the set of examples that have arguments; $AES \subseteq ES$
Evaluate arguments of examples in AES (user-defined "quality" evaluation function) and sort examples in AES according to quality of their arguments.
while AES is not empty do
  Let AE1 be the first example in sorted AES.
  Let BEST_RULE be ABFind_best_rule(ES,AE1,T)
  Add BEST_RULE to RULELIST.
  Remove from AES examples AB-covered by BEST_RULE.
end while
for all RULE in RULE_LIST do
  Remove from ES examples AB-covered by RULE.
end for
Add rules obtained with CN2ForOneClass(ES,T) to RULE_LIST
```

Algorithm 1: Covering algorithm of ABCN2 algorithm that learns rules from examples ES for given class T

```
Procedure ABFind_Best_Rule(Examples ES, Example E, TargetClass T)

Let the set STAR contain argument of E.
Let $BEST\_CPX$ be argument of E.
Let $SELECTORS$ be the set of all possible selectors that are TRUE for E
while STAR is not empty do
  {Specialize all complexes in STAR as follows}
  Let $NEWSTAR$ be the set $\{x \wedge y \,\| x \in STAR, y \in SELECTORS\}$
  for every complex in $C_i$ in NEWSTAR do
    if $C_i$ is statistically significant(ES,T) and quality($C_i$) > quality(BEST_CPX) then
      replace the current value of $BEST\_CPX$ by $C_i$
    end if
  end for
  Let STAR be best N complexes from NEWSTAR.
end while
return rule: "IF $BEST\_CPX$ THEN T.
```

Algorithm 2: Algorithm that finds best rule from a set of argumented examples. The "quality" of a complex is evaluated by the same user-defined evaluation function as in ABCN2ForOneClass.

Initial value of variable Star is argument given to argumented example. A rule induced from an argumented example must AB-cover this example, therefore it will have to contain the reasons of the argument. The easiest way to ensure this is to start learning from it.

Only selectors that are true for argumented examples are considered. This ensures the coverage of the seed example by the induced rule.

Statistical significance. During search for a rule, many hypotheses are taken into account, therefore it makes sense to set pre-pruning α of the likelihood ratio statistic [9] higher than this is usually done. We chose to multiply it with the number of all possible specializations in the first step divided by the number of possible specializations in the first step that cover the argumented example. Moreover, to ensure that specialization is significant, the statistical difference of the class distribution of a new rule needs to be compared to the class distribution corresponding to the underlying argument rather than to the distribution in the whole data set.

It should be noted that argument based ML is not limited to CN2. Various standard ML techniques can be extended in a similar way to their argument-based variants. The main point of the experiment in this paper is not to compare ABCN2 with other ML techniques (other than CN2), but to compare a ML method with its argument-based enhancement. To this end, the comparison of main interest in this paper is between CN2 and its AB enhancement ABCN2.

3 Application to Legal Domain

3.1 The data set

As previously mentioned the data set used in these experiments is that first used in [3]. The data concerns a fictional welfare benefit. The benefit is payable if six conditions are satisfied. These conditions were chosen to represent different kinds of condition that are found in the legal domain, so that we can see whether the different form of conditions affects their discoverability.

The notional benefit was a fictional welfare benefit paid to pensioners to defray expenses for visiting a spouse in hospital. The conditions were:

1. The person should be of pensionable age (60 for a woman, 65 for a man);
2. The person should have paid contributions in four out of the last five relevant contribution years;
3. The person should be a spouse of the patient;
4. The person should not be absent from the UK;
5. The person should have capital resources not amounting to more than 3,000;
6. If the relative is an in-patient the hospital should be within a certain distance: if an out-patient, beyond that distance.

These conditions represent a range of typical condition types: 3 and 4 are Boolean necessary conditions, one which should be true and one false; 5 is a threshold on a continuous variable representing a necessary condition, and 2 relates five Boolean variables, only four of which need be true. 1 and 6 relate the relevance of one variable to the value of another: in 1 sex is relevant only for ages between 60 and 65, and in 6 the effect of the distance variable depends on the Boolean saying whether the patient is an in-patient or an out-patient. We can see these six conditions either as explicit conditions or as ways of making operational concepts such as elderly, sufficient contribution record, close relative, presence in the UK, insufficient capital resources, and attributable expenses respectively.

The underlying rules used to generate the synthetic data were not told beforehand to those co-authors of this paper that carried out the learning experiments with ABCN2. So although the rules were known to one of the co-authors (TBC), the experiment was done as if they were not known.

A possible criticism of this experiment could be based on the fact that our experimental data was artificial rather than real world. However, in ML experimentation with artificial data is quite common and acceptable in testing ML methods. It has the advantage over the use of real-world data in that the experiment is better controlled and the success of learning is easier to assess. Therefore, quite normally in ML research, the first assessment of a new ML method is done with artificial data, and these results are used as an indication of how the method would perform in practice on real-world data. The obvious limitation of using artificial data, on the other hand, is that such data might not be sufficiently representative of real-world applications.

The data was generated using a program written in Common LISP. For this experiment a data set of 2400 records was used: 1200 satisfying all of the conditions, and equal numbers of the remainder being designed to fail. For records designed to fail one of the conditions, satisfaction or otherwise of the remaining conditions was decided randomly for each condition separately. There are thus twelve attributes relevant to the decision: age, sex, the five contribution conditions (called cont5, cont4, cont3, cont2 and cont1), spouse, absent, capital, distance and inpatient. In addition to these attributes each record contains fifty two irrelevant attributes, half of which are continuous and half Boolean. An ideal set of rules would be:

1. IF age < 60 THEN qualified = no;
2. IF age < 65 and sex = m THEN qualified = no;
3. IF any two of cont5, cont4, cont3, cont2 and cont1 = n THEN qualified = no;
4. IF spouse=no THEN qualified=no;
5. IF absent=yes THEN qualified=no;
6. IF capital>3000 THEN qualified=no;
7. IF inpatient=yes AND distance> 750 THEN qualified=no;
8. IF inpatient=no AND distance$\leq$ 750 THEN qualified=no;

Probably we should expect (3) to be expressed as ten separate rules containing each pair of the contribution factors, which would be a total of sixteen rules to describe the problem fully.

3.2 Experiment in Argument Based Machine Learning

As already mentioned the algorithm used in the experiments was ABCN2, argument based CN2. The original data (2400 records) was randomly split into a learning set containing 70% of the cases, and a test set (the remaining 30%) used to assess the accuracy of the generated rules on new cases.

The first set of rules was generated from examples without any arguments. So the resulting rules were as if generated with CN2. These rules were:

1. IF capital $>$ 2900 THEN qualified = no;
2. IF age $\leq$ 59 THEN qualified = no;
3. IF absent = yes THEN qualified = no;
4. IF spouse = no THEN qualified = no;
5. IF cont4 =no AND cont2 = no THEN qualified=no;
6. IF age $>$ 89 THEN qualified = no;

Of these (1) - (5) are correct (or very close). Nine contributions rules and the two distance rules are missing and rule (6) is wrong. There is thus considerable scope for improvement. None the less a high degree of accuracy is achieved by these six rules: 99% for both the learning and the test sets. Accuracy of classification is not, however, of prime interest: the motivations given in section 1 make it clear that it is the interpretability of the rules discovered that is of primary interest.

After inducing these rules with CN2, our plan was to give arguments to some of the examples in the learning set and using ABCN2 to induce better rules. The main problem was to select examples that, when argumented, would best help the learning program to induce better rules. To this end we designed an iterative procedure, each iteration consisting of the following steps:

1. *Find an example that needs to be argumented.* This step involves a search for the most "problematic" example (e.g. outlier) in the learning set that renders the learning difficult. For this task we ran a repeated (10 times) internal (that is inside the training set) random sampling 70/30, where the first part (70%) was used for learning and the second (30%) for testing. The example that was most often misclassified was chosen as the example that needed to be argumented.
2. *If problematic example was not found* (in step 1), then stop the iteration.
3. *Give arguments to the chosen example.* An expert gives reasons why the example is in one class and not in the other.
4. *Induce rules on the learning set using ABCN2.*

In the case of our legal data, the most frequently misclassified example failed on the contributions condition. The argument given for this example was that cont5, cont4 and cont1 are all false. When this argumented example, together with all the other (non-argumented) examples was given to ABCN2, two additional contribution rules were induced:

$$\text{IF } cont5 = no \text{ AND } cont4 = no \text{ AND } cont1 = no \text{ THEN } qualified = no;$$

IF $cont2 = no$ AND $cont3 = no$ THEN $qualified = no$;

and the accuracy increased slightly.

In a third learning iteration, an argument was added to an additional misclassified case in which distance was too great for an inpatient. This time the erroneous rule (6) disappeared and was replaced by a rule relating to inpatiency and distance (although with an approximate threshold) and another contributions rule:

IF $inpatient = yes$ AND $distance > 735$ THEN $qualified = no$;
IF $cont1 = no$ AND $cont5 = no$ THEN $qualified = no$;

Iterations four, five and six added three more arguments based on failure of the contribution conditions, which resulted in a different contributions rule. In iteration seven, the argument was given that distance was too small for outpatient. This produced the rule

IF $inpatient = no$ AND $distance \leq 735$ THEN $qualified = no$;

and rearranged the contribution rules somewhat.

At this point there were no misclassified examples in the internal validation tests any more, and so no further argumentation with our iterative procedure was possible. The final accuracy on the test set was 99.8%. This means that one out of 720 test cases was misclassified, and all the rest were classified correctly. The final set of rules were:

1. IF $capital > 2900$ THEN $qualified = no$;
2. IF $age \leq 59$ THEN $qualified = no$;
3. IF $absent = yes$ THEN $qualified = no$;
4. IF $spouse = no$ THEN $qualified = no$;
5. IF $cont4 = no$ AND $cont2 = no$ THEN $qualified = no$;
6. IF $inpatient = yes$ AND $distance > 735.0$ THEN $qualified = no$;
7. IF $inpatient = no$ AND $distance \leq 735$ THEN $qualified = no$;
8. IF $cont3 = no$ AND $cont2 = no$ THEN $qualified = no$;
9. IF $cont5 = no$ AND $cont3 = no$ AND $cont1 = no$ THEN $qualified = no$;
10. IF $cont4 = no$ AND $cont3 = no$ AND $cont1 = no$ THEN $qualified = no$;
11. IF $cont5 = no$ AND $cont4 = no$ AND $cont1 = no$ THEN $qualified = no$;

(1) - (5) are all good rules and remain from the first pass. (6) and (7) have the right format, but the threshold is slightly inaccurate. The remaining four rules approximate the ten ideal contribution rules. The total number of argumented examples after the seven iterations was seven. The one misclassified example was (omitting the irrelevant attribute values): (($age = 84$, $sex = male$, $cont1 = no$, $cont2 = yes$, $cont3 = no$, $cont4 = yes$, $cont5 = yes$, $spouse = yes$, $absent = no$, $capital = 130$, $distance = 1320$, $inpatient = no$), $qualified = no$). This example is misclassified because the particular combination of contribution conditions is not covered by the approximate contribution rules induced.

4 Experiments with noisy data

Learning from artificial data sets is usually considered easier than learning from real world data sets. One reason for this is that artificial data are typically noise-free whereas real world data typically contain noise. So to cope with real world data, a learning method has to be able to deal with noise. In this section we investigate the question how robust our learning algorithm ABCN2 is with respect to noise in data. To this end we artificially introduced random noise of varying severity in our learning data as described below. Intuitively we expected that background knowledge in the form of arguments should improve the method's resistance to noise in comparison with CN2. This expectation was confirmed by the experiments described in this section.

The experimental procedure for this experiment was as follows. First, we split the data set to learning set (70%) and test set (30%). Then we added random noise into the learning set, induced rules with both CN2 and ABCN2, and measured the accuracy of both sets of rules on the (noise-free) test set. To study how the severity of noise affects the success of learning, we repeated the experiment for various rates of noise. The chosen rates were: 0%, 2%, 5%, 10%, 20%, and 40%. A noise rate p means that with probability p the class value of each learning example is replaced by a random value drawn from {yes, no} with distribution (0.5, 0.5). Each time CN2 and ABCN2 were run with correspondingly "noised" examples plus the same noise-free argumented examples as in section 3.2. This whole procedure was repeated 10 times in order to obtain confidence intervals of the estimated average classification accuracy for both CN2 and ABCN2 for each noise rate.

Figure 1 shows the results of both methods CN2 and ABCN2. Both methods were run with the default settings of their parameters. These standard values are: m=2 in m-estimate, α=0.001(likelihood ratio statistics threshold), and minimal coverage of a rule is set to 2 (that is, a rule to be acceptable has to cover at least two examples).

This figure shows that, as expected, ABCN2 clearly outperforms CN2 when rate of noise in data increases. The difference in average accuracies between ABCN2 and CN2 jumped from 0.3% at 0% noise to 3.3% at 20% and to 1.7% at 40% noise. ABCN2 outperformed CN2 on 0%, 20%, and 40% noise with high statistical significance (t-test,p<0.001; see Table 2). ABCN2 was also the better method for other noise rates but with lower significances.

The ability to handle large amount of noise is very important in the kind of routine legal decision making we are addressing. Errors rates are very high: Groothuis and Svenson[12] report experiments which suggest that 20% may be a low estimate of incorrectly decided cases. The problem is internationally widespread. The US National Bureau of Economic Research reports of a particular benefit[1]:

> "The multistage process for determining eligibility for Social Security Disability Insurance (DI) benefits has come under scrutiny for the

[1] From Web Page: *http://www.nber.org/aginghealth/winter04/w10219.html*

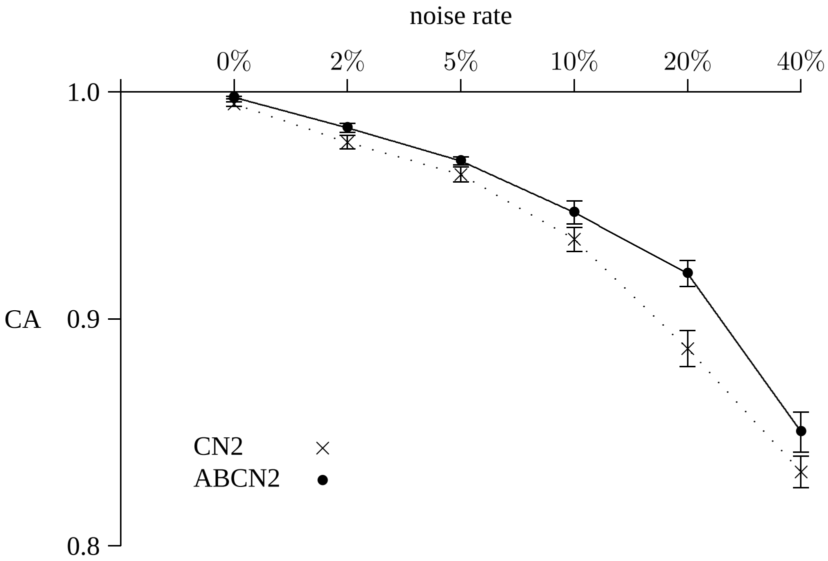

Fig. 1. CN2 and ABCN2 compared on learning data sets with different proportions of noise in class variable.

noise	CN2	ABCN2	Sig.
0%	0.9947±0.0010	0.9976±0.0005	<0.001
2%	0.9778±0.0030	0.9842±0.0020	0.002
5%	0.9636±0.0034	0.9696±0.0017	0.005
10%	0.9351±0.0053	0.9469±0.0051	0.037
20%	0.8869±0.0079	0.9200±0.0056	<0.001
40%	0.8326±0.0068	0.8503±0.0088	0.001

Table 2. Results of CN2 and ABCN2 on noisy data. The first column shows noise rates, second and third contain average accuracy and standard error of accuracy estimate for CN2 and ABCN2 respectively, and the last column gives significances of differences between averages computed with pairwise t-test.

> length of time the process can take - 1153 days to move through the entire appeals process, according to a recent Social Security Administration (SSA) analysis - and for inconsistencies that suggest a potentially high rate of errors. One inconsistency is the high reversal rate during the appeals process - for example, administrative law judges, who represent the second level of appeal, award benefits in 59% of cases. Another inconsistency is the variation in the award rates across states - from a high of 65% in New Hampshire to a low of 31% in Texas in 2000 - and over time - from a high of 52% in 1998 to a low of 29% in 1982."

Again an official UK Publication produced by the Committee of Public Accounts[2]:

> "Finds that the complexity of the benefits system remains a major problem and is a key factor affecting performance. Skills of decision-makers need to be enhanced through better training and wider experience. Too few decisions are right first time, with a error rate of 50% for Disability Living Allowance. There are also regional differences in decision making practices that may lead to payments to people who are not eligible for benefits."

This makes it clear that robustness in the face of large amounts of noise is essential if a learning techniques is to be applied to data from this domain

5 Comparison with other learning methods

In this section we will compare this experiment with previous learning experiments with the same data set. We will compare three learning techniques here:

- Argument Based Machine Learning (ABML), this paper
- Neural Nets (NN), in [3]
- Defeasible Logic (DL), in [5].

DL uses an algorithm called HeRo to derive a minimal theory in Defeasible Logic. Bench-Capon [3] describes experiments with NN using a number of net topologies, the most successful of which used a single hidden layer of 12 nodes in a fully connected network. There has also been an experiment in learning Association Rules (AR) [4]. We will not include AR in our comparison because of significant differences in their experimental setup. Irrelevant attributes were omitted from the learning data in that experiment making the learning easier, and the format of the rules produced was somewhat different. DL does not handle continuous variables and so such variables are mapped into Booleans using user-supplied thresholds.

The rules discovered in DL were:

[2] Getting it right: Improving Decision-Making and Appeals in Social Security Benefits. Committee of Public Accounts. London: TSO, 2004 (House of Commons papers, session 2003/04; HC406)

IF $distance > 700$ AND $inpatient = no$ THEN $qualified = no$;
IF $spouse = no$ THEN $qualified = no$;
IF $absent = yes$ THEN $qualified = no$;
IF $age < 60$ THEN $qualified = no$;
IF $capital > 300$ THEN $qualified = no$;

These rules are a subset of the rules discovered by ABML: they fail to recognise the contributions conditions, and also omit the rule governing inpatients.

The neural network did not produce rules, but the contribution of the various inputs was estimated using the technique described in [3]. The most important contributing attributes, in descending order of importance, are shown in Table 3.

Factor	Influence	Degree
Spouse	positive	0.995
Absent	negative	0.984
Cont5	positive	0.920
Capital	negative	0.882
Cont1	positive	0.875
Cont4	positive	0.819
Cont2	positive	0.809
Cont3	positive	0.797
Age	positive	0.779
Registered	positive	0.776
Score	positive	0.720
Sex	negative	0.646

Table 3. Important inputs in [3].

This experiment did recognise the importance of the contribution conditions, but the inpatient condition and distance to hospital were not seen as of any significance. Also two irrelevant attributes, registered and score, were seen as more important than sex. Note also that the degree of influence drops quite steeply before sex is found.

Both DL[5] and NN[3] report a very high level of accuracy (although details of accuracy evaluation are not given). This was also obtained by the traditional machine learning method of CN2. That it is possible to train a system to classify with an excellent degree of success is also the experience of other neural nets work such as Split-Up [13] and the experiments of Borges et al. [14]. Such success, however, should not blind us to the problems: none of the techniques produced rules which faithfully reflected the conditions for the benefit. This should, perhaps, be unsurprising given that all machine learning methods respect some sort of minimum description length principle. However, this suggests that their use to classify actual cases would be of dubious acceptability, since they would make some systematic errors, and so discriminate against particular classes of individuals.

5.1 Identification of attributes

None of the three systems were able to adequately discover the significance of the sex of the claimant. NN did recognise it as of some small importance, but ranked two of the irrelevant attributes as more important. It should be noted, however, that in the case of our particular data set, the learning systems should not really be blamed for this omission. The learning data only contains male applicants between 60 and 65 that fail to qualify also for other reasons, in addition to sex = male. So for correct classification of this data set, attribute sex happens not to be necessary. This shows that even with a large data set some nuances may not be covered.

Whereas NN recognised that the contributions attributes were of significance, DL does not consider any of the contribution attributes. CN2 identified one contribution rule, and some guidance through argumentation enabled ABCN2 to extend its consideration of these factors.

The other rule which gave problems to all three approaches was the rule relating distance to inpatiency. NN gave no significance to these attributes, and ABML required explicit guidance from arguments to pick up on the two rules. DL correctly identified the rule for outpatients, but did not recognise the significance of distance for inpatients.

Continuous attributes can also present problems, often because the techniques cannot handle such attributes. DL - in common with other systems such as Split-Up - preprocessed the data by mapping ranges into Booleans. This does, of course, require some domain knowledge to ensure that useful ranges are chosen. ABCN2 does comparatively well, correctly identifying automatically the thresholds for age and capital, although being a little inaccurate on the more difficult distance attribute.

We may observe that certain kinds of rules pose few problems: Booleans, whether required to be true or false, and thresholds which must or must not be exceeded, are easily identified by all techniques. The inpatiency-distance rule, however, is hard to detect: as this is essentially of the form of an XOR rule, this may be unsurprising since such concepts have always posed difficulties for neural nets and for machine learning generally. Perhaps, however, such concepts might be thought somewhat contrived.

More serious is the contributions rule, which does reflect a type of concept frequently encountered in law. Concepts in which a number of factors - none of them individually necessary or sufficient - must be considered are not uncommon in law. For example in several European countries including the UK and Sweden, determining whether a cohabiting couple should be treated as married for welfare benefits purposes is assessed in this way. It is to such concepts that much work on case based reasoning in AI and Law has been directed. Such concepts are, however, extremely "unfriendly" for rule based learning. It should be noted that 'additive' hypothesis languages (e.g. naive Bayes, or linear inequalities, also implicitly included in neural nets) are much more appropriate for the formulation of this condition. For example, the single linear inequality:

$$cont1 + cont2 + cont3 + cont4 + cont5 >= 4$$

would express the contributions rule, whereas ten propositional rules are required. The superior performance of NN on this condition supports this view. The difficulties, however, should serve as a caution to their application in domains were concepts with this style of characterisation are suspected. Rule learning would be greatly helped in learning "additive hypotheses" by construction of new attributes of the "additive" form as indicated above.

On the positive side, all the techniques discussed here have proved robust in the face of the large number of irrelevant attributes. A useful implication of this is that when setting up such learning exercises one can, if using suitable techniques, be as inclusive of information as possible. This is a desirable feature of systems designed to learn in legal settings where the omission of a crucial attribute would be very damaging.

5.2 Use of domain knowledge

NN does not use domain knowledge. DL uses it to discretise continuous attributes. ABML makes its use, in the form of arguments, an essential part of the strategy. The work suggests that such domain knowledge is likely to be required if all relevant attributes are to be identified. Since, typically this knowledge is readily available in the form of justifications for decisions, its use should not present a problem.

Advantages of ABML can be seen when making comparison with knowledge intensive techniques used in Case Based reasoning systems such as CATO [15]. The analysis effort required to represent the cases used in the CATO system is very large indeed. Moreover, attempts to automate this process have met with very limited success (e.g. Bruninghaus and Ashley [16]). The key problem is the volume of decisions available, and the lack of any principle to guide selection of those which will add to our knowledge and those which will merely duplicate what has gone before. Moreover it may be that particular decisions may give a misleading emphasis to particular attributes. Because ABML goes as far as it can without guidance, it is able to direct attention to the decisions that will have an effect in closing some gap in the knowledge, or rectifying some misunderstanding. We may thus be sure that we are only looking at decisions which will have an impact, and which are really necessary to complete the analysis. Perhaps the main strength of ABML is that it does focus our attention on particular decisions in this way.

6 Concluding remarks

In this paper we have described the application of a new machine learning technique to a legal problem, making use of a data set which has been the subject of previous experiments in AI and Law. The proposed technique is ABCN2, which is argument based enhanced version of well known algorithm CN2. We found that our technique performed at least as well as the others in terms of accuracy - which is very well indeed. Additionally, we showed superior robustness to noise when compared to classical CN2. However, it is not the accuracy of classification that is most important to meet the requirements stated in the introduction to this paper: it is the quality, in terms of interpretability, of the rules generated that is crucial. In the initial learning stage, without arguments, the performance of our technique was similar to that of the others. We, however, can inform further iterations with domain knowledge taken from the justification of particular decisions, and this is able to give a great improvement to the quality of the rules. What is important here is that the specific elements of domain knowledge to be used are identified by the technique (rather than the trainer), so that they are called into play on as and when they are found to be needed, and attention can be focused on a small number of critical decisions identified by the technique. Additionally, we explored the robustness of ABCN2 in the face of erroneous decisions. It was found in the experiment that ABCN2 is more robust against noise than its non-argument original CN2. This is important because many of the administrative law applications to which we would wish to apply the technique exhibit quite large error rates [12].

References

1. Bench-Capon, T.: Knowledge Based Systems Applied To Law: A Framework for Discussion. In: Knowledge Based Systems and Legal Applications. Academic Press. (1991) 329–342
2. Edvards, L.: Modelling law using a feminist theoretical perspective. Law, Computers and Artificial Intelligence **4** (1995) 95–110
3. Bench-Capon, T.: Neural nets and open texture. In: Fourth International Conference on AI and Law, Amsterdam, ACM Press (1993) 292–297
4. Bench-Capon, T., Coenen, F.: An experiment in discovering association rules in the legal domain. In: 11th International Workshop on Database and Expert Systems Applications, Los Alamitos, IEEE Computer Society (2000) 1056–1060
5. Johnston, B., Govenatori, G.: Induction of defeasible logic theories in the legal domain. In: Ninth International Conference on AI and Law, Edinburgh, ACM Press (2003) 204–213
6. Bratko, I., Možina, M.: Argumentation and machine learning. In: Deliverable 2.1 for the ASPIC project (2004)
7. DeJong, G., Mooney, R.: Explanation-based learning: and alterative view. Machine Learning **1** (1986) 145–176
8. Mitchell, T., Keller, R., Kedar-Cabelli, D.: Explanation-based generalization: a unifying view. Machine Learning **1** (1986) 47–80
9. Clark, P., Niblett, T.: The CN2 induction algorithm. Machine Learning Journal **4** (1989) 261–283
10. Clark, P., Boswell, R.: Rule induction with CN2: Some recent improvements. In: Machine Learning - Proceeding of the Fifth Europen Conference (EWSL-91), Berlin (1991) 151–163

11. Cestnik, B.: Estimating probabilities: A crucial task in machine learning. In: Proceedings of the Ninth European Conference on Artificial Intelligence. (1990) 147–149
12. Groothius, M., Svensson, J.: Expert system support and juridical quality. In: Jurix, Amsterdam, IOS Press (2000) 1–10
13. Zeleznikow, J., Stranieri, A.: Knowledge discovery in the split up project. In: Proceedings of the Sixth International Conference on AI and Law, New York, ACM Press (1997) 89–97
14. Borges, F., Borges, P., D., B.: A connectionist model to justify the reasoning of a judge. In: Proceedings of Jurix, IOS Press, IOS Press (2002) 113–122
15. Aleven, V.: Teaching Case-Based Argumentation Through a Model and Examples. PhD thesis, University of Pittsburgh (1997)
16. Brüninghaus, S., Ashley, K.D.: Toward adding knowledge to learning algorithms for indexing legal cases. In: ICAIL '99: Proceedings of the 7th international conference on Artificial intelligence and law, New York, NY, USA, ACM Press (1999) 9–17