Finding Factors

Learning to Classify Case Opinions under Abstract Fact Categories

Stefanie Brüninghaus and Kevin D. Ashley
University of Pittsburgh
Learning Research and Development Center,
Intelligent Systems Program and School of Law
Pittsburgh, PA 15260
steffi+@pitt.edu, ashley+@pitt.edu

Abstract

This paper presents preliminary work towards automatically assigning to full-text opinion texts the applicable factors, that is, fact patterns influencing the outcome of a legal claim, to full-text opinion texts, which are used in CATO's model of case-based legal argumentation. In spite of the fundamentally different representation and methods for comparing cases in a CATO-style case-based reasoning system versus text-retrieval systems, the paper provides evidence that there exists a connection between the notion of similarity under both paradigms.

We consider the task as a classification problem, and apply machine learning methods, where CATO's Case Database of 147 trade secret law cases are used as training instances. Since the generalization power of purely inductive algorithms, which rely on representation from conventional text-retrieval, does not measure up to the complexity of the concepts corresponding to the factors, the learning algorithms' performance is not satisfactory yet. Trying to address this problem, we discuss techniques that will allow integrating domain specific knowledge, about the use of cases in legal argumentation and about the interrelations among the factors, which is expressed in CATO's Factor Hierarchy. Our hypothesis is that adding this knowledge will enhance performance, and that a successful system will facilitate building legal case-based reasoning systems and legal research.

1 Introduction

Large numbers of legal cases are available as unstructured, full-text documents in document collections maintained by legal publishers. Although huge effort is spent in maintaining indexing systems, like for instance, West's keynumber framework, the categories represented by the keywords are rather broad and tend to reflect the general practice area rather than the characteristics of a specific case. Legal case-based reasoning systems, on the other hand, rely on structured, symbolic representations that capture the meaning of a case, and that correspond to the way human professionals reason with those cases. Hitherto, assigning these indices is a tedious, manual effort, and thus, in developing a legal case-based reasoning system, tremendous time must be spent on building the case-base. Candidate cases have to be identified and read, the most pertinent cases selected, and then represented with some combination of frames and factors which capture relevant factual strengths and weaknesses (see, e.g., HYPO (Ashley 1990), BANKXX, (Rissland, Skalak, & Friedman 1993), and CATO (Aleven & Ashley 1996)).

This paper presents our work towards developing automatic methods for assigning abstract fact categories to legal case opinions so that the cases could then be used in a CATO-style case-based reasoning system. The project is strongly influenced by the model of legal case-based argumentation of CATO (Aleven & Ashley 1996). CATO is an intelligent learning environment for teaching case-based argumentation skills to law students (Aleven & Ashley 1997). Cases are represented in terms of factors, prototypical fact situations that tend to strengthen or weaken plaintiff's legal claim. In CATO's model of case-based argument, cases are selected and employed in analogical arguments how the court should decide a problem. The probblem is analogized to cases with favorable outcomes, and distinguished it from cases with unfavorable outcomes. The argumentation model includes six basic argument moves, in which a problem situation is compared and contrasted to previous cases in terms of abstract factors. Factors are factual strengths and weaknesses that tend to be relevant for a legal claim. Abstract legal issues are expressed in CATO's Factor Hierarchy (Aleven & Ashley 1996) which relates factors to more aggregated concepts and ultimately to legal issues raised by the legal claim. In CATO's Case Database contains the symbolic representation of 147 trade secret misappropriation cases in

ICAIL-97, Melbourne, Australia © 1997 ACM 0-89791-924-6/97/06.$3.50

terms of factors, along with the full-text opinions in which judges record their decisions and rationales for litigated disputes.

In the remainder of this paper, we introduce related research to define the problem further, as well as illustrate the advantages and disadvantages of possible solutions. Then, we outline our approach to the problem and describe the methodology underlying the text representation. To show that this representation makes sense, we establish a connection between the representation of legal cases in CATO's factor model and the Vector-Space Model developed in Information Retrieval. After that, we report a series of experiments in which we found that purely inductive methods from machine learning are not powerful enough for the complexity of the problem. In a set of hypotheses, we suggest and illustrate ways in which domain knowledge can be integrated in order to overcome the limitations discovered before. The paper concludes with a description of the practical impact the research can have.

2 Approaches for Analyzing Text Documents

Researchers in different fields have tried to extract information from text documents so that one can reason symbolically about their contents. The classification of texts is an ongoing research issue in information retrieval. In the TREC evaluations, the routing task involves assigning documents to a topic defined by (usually very large) sets of documents. Documents are labeled as relevant/non-relevant, from which query vectors are calculated using relevance feedback methods. Recent research has explored the use of machine learning algorithms for deriving text classifiers (Lewis *et al.* 1996; Callan 1996; Papka, Callan, & Barto 1996).

Finding a good representation for text documents is still an open issue in IR, and despite intense research, no single representation has been found consistently to increase retrieval effectiveness. (Lewis 1992) describes that as the "equal effectiveness paradox." For legal applications, however, the use of domain-specific representations seems to be a suitable way to improve performance, as demonstrated successfully in Flexlaw (Smith & Gelbart 1995) and SCALIR (Rose 1994). Both projects suggest that explicitly representing legal concepts, statutes and references to cases is beneficial when dealing with legal text.

Flexlaw is an IR system designed for the particular requirements of legal research. Documents are represented by four quadrants, three of which are concept recognizers employing a controlled vocabulary. First, concepts from a hand-crafted dictionary are found in a table lookup. Then references to statutes and to case law are parsed. The remaining passages are assumed to contain the facts of the case (and not the court's reasoning). In query and document vectors, tokens from all four quadrants are merged without distinguishing between the quadrants they come from. Matching and ranking follows the vector-space model. SCALIR is a hybrid symbolic and connectionist system, implemented for the domain of copyright infringement. It consists of a network of different types of nodes and links. The nodes are legal cases, copyright statutes and concepts (West's key numbers and terms), connected by different types of links, which represent, e.g., citations or concepts involved in cases. Given a query as a set of nodes, which may comprise documents, statutes and terms, activation is spread along the links. The system retrieves the nodes with the highest activation value in response to the query. It also learns by updating the weights according to the user's browsing behavior.

In machine learning research, the main focus in text learning has been on filtering relevant documents, in particular Web pages (Pazzani, Muramatsu, & Billsus 1996; Craven *et al.* 1997), email messages and usenet news messages (Lang 1995). These approaches do not consider any domain specific knowledge or semantic structure, and rather represent documents as highly-dimensional vectors over the words. Statistical, probabilistic and information-theoretical methods which use word frequency counts have been employed for deriving the components' weights and for classifying the documents under different categories. Very recent research (Freitag 1997; Craven *et al.* 1997) aims at learning to instantiate objects and relations from a predefined ontology.

Similarly, Information Extraction (Cowie & Lehnert 1996), a subfield of natural language processing, is concerned with identifying specific pieces of information within documents for filling in predefined frames with a relatively simple structure. Since legal cases are much more complicated than, for example, the news-stories aimed at in IE, and often include adversarial arguments, the pattern matching and language processing techniques used in IE are not applicable here, as experiments reported in (Daniels 1996) confirm.

In recognition of this problem, a different approach has been taken in SPIRE (Daniels 1996). It addresses the same problem as our research, namely overcoming the current bottleneck in deriving a representation of legal opinion texts suitable for use in a case-based reasoning system. It uses annotated cases for identifying text passages that contain the information needed to fill in the slots of a frame-based case representation. These annotations are used by the relevance feedback module of the INQUERY retrieval system (Callan, Croft, & Harding 1992). The most similar text sections in a new case are retrieved and presented to a user. While Daniels wants to focus the human user on the pertinent sentences in a text for filling detailed slots in a frame-based representation, our goals are somewhat different. We intend to develop a system that classifies a new case under more abstract factual patterns and to make use of the information about relations among factors contained in the Factor Hierarchy, rather than locating relevant information in a case. As far as we know, Daniels' project does not employ background information about factors or the structure

of legal opinions, which we will include in our work.

3 Learning a Text Classifier for Factor Assignment

Automatically assigning factors to opinion texts is a classification task; when presented a new case a system has to decide for each factor whether the case belongs to the concept corresponding to the factor or not, i.e., classify it under that concept. The concepts are defined explicitly, by a definition of what the factors mean and when they apply, and implicitly, by the set of cases in CATO's Case Database in which a factor applies, and the complemetary set of cases in which the factor does not apply. The cases in which factor f_i applies form a class of instances under the concept $\mathcal{F}_i$. We are going to use machine learning algorithms and additional domain knowledge to find a weighted set of features that enable deciding whether a new trade secrets case belongs to concept $\mathcal{F}_i$.

More precisely, we can describe the learning task as follows (see also Fig. 2):
For each concept $\mathcal{F}_k$ corresponding to factor f_k:
Given: A set of positive instances C^k, a set of negative instances $\overline{C}^k$, and domain knowledge, e.g. in the Factor Hierarchy,
Learn: A classifier lc_k, that predicts for a new case c_i whether factor f_k applies, where lc_k consists of a weight component w_k and a threshold t_k, which may be given before learning: $lc_k = (w_k, t_k)$.

After the training phase, this classifier will then be used for deciding for each factor whether it applies to a new case, which was not in the training set. As input, the system will be given a full-text document, and output a prediction of which factors apply, as shown in Fig. 2. More formally, the overall system works as follows:
Input: The full-text opinion of a new case c_i,
Proceed for each factor f_k:

- Derive a feature/value vector $\mathbf{d}_i$ using the methods in Sec. 3.1,
- Apply the classifier learned for factor f_k to $\mathbf{d}_i$, which yields a classification-value $cv_{i,k}$.

Output: The prediction whether the factor f_k applies in case c_i. If the classification-value $cv_{i,k}$ is greater than zero, the prediction $p_{i,k}$ will be that factor f_k applies in case c_i, if it is less than zero, the prediction is that the factor does not apply.

3.1 Text Representation

Machine learning algorithms can not work with a text given as a character stream as input. They require feature/value vectors, where the features may be meaningful symbols or numeric characteristics of instances. Here, we are using methods from IR to bring texts into a suitable format.

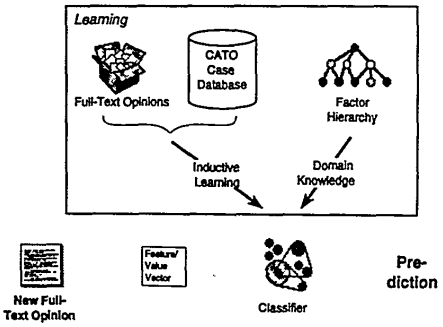

Figure 2: Overview of Classifier's Operation

For our task, we are applying Lexical Analysis, Term Weighting and Document Matching (Frankes & Baeza-Yates 1992). In the Lexical Analysis, texts are split up into single words, or tokens, and stop-words[1] are removed. The tokens are reduced to words stems by removing suffixes[2] with Porter's Algorithm. In addition, we take pairs of adjacent non stop-words into account. A sentence like "The secret recipe is considered to be a trade secret" will be translated into the frequency vector tf = (consid 1, recip 1, secret 2, trad 1, tradscret 1, secretrecip 1). Term Weighting assigns weight $d_{i,j}$ for each word i in document j which reflects the relative importance of the word for the document, and is based on the observation, that the more often the word occurs in a document, the more important it is, but the more documents the words can be found in, the less important it is. Thus, the weight is calculated as the frequency $tf_{i,j}$ of the word within the document multiplied by the logarithm of the collection size divided by the number of documents in the collection which contain word i. In the Document Matching step, similarity of a pair of documents is defined as the cosine of the angle between their vectors, which is calculated as the inner product of the document vectors.

$$d_{i,j} = \frac{tf_{i,j}}{\log(\mathrm{N}/df_i)}, \quad \mathrm{sim}(x,y) = \frac{\mathrm{x}\,\mathrm{y}}{\|\mathrm{x}\|\,\|\mathrm{y}\|} = \frac{\sum x_i y_i}{\sqrt{\sum {x_i}^2}\sqrt{\sum {y_i}^2}}$$

3.2 Connection between factor model and vector-space model

In the factor model cases are represented by a small set of factors, that reflect the factual strengths and weaknesses of a case. These cases are compared symbolically, and sets of factors shared between cases are an indicator of similarity. Apparently, this is fundamentally different from the vector-based definition of similarity above, which raises the question whether the two models capture orthogonal properties of cases. Therefore, we ran an experiment on the 147

[1]Frequent words like "a", "and", "the"
[2]Words endings like "-ing", "-ness", "-tion"

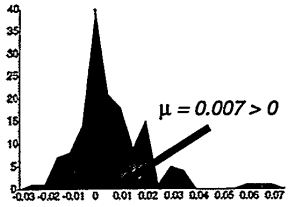

Figure 3: Frequency Distribution of $\Delta_{top\text{-}tier,all\text{-}cases}$

trade secret cases represented in Cato's Case Database, and the (full-text) opinions for those cases, in which we investigated whether cases that are more similar under CATO's Factor Model are also more similar under the IR Vector-Space Model. Following HYPO's definition (Ashley 1990), cases are considered *relevantly-similar* if they have at least one factor in common. These cases can be found in HYPO's claim lattice, where the problem situation is the root node, and *more-on-point* cases sharing more inclusive sets of factors with the problem situation are closer to the root node. We defined the *top-tier* cases as those that either share all factors of the problem situation, or are more-on-point than all other cases in a branch of the claim lattice. They can be found in the root node and the first layer of the claim lattice.

For each case in CATO's Case Database, we calculated the average similarity avg-$sim_{top\text{-}tier}$ to the *top-tier* cases by taking the mean of the cosines between the the cases and document vectors of the *top-tier* cases. Accordingly, we calculated the average similarity to the *relevantly-similar* cases and to *all-cases*. For many cases, the average similarity to *all-cases* was smaller than to *relevantly-similar* and *top-tier cases*, but we also found a number of cases where this did not hold. We used statistical inference methods to confirm our hypothesis that:

$$\text{avg-sim}_{top\text{-}tier} \succ \text{avg-sim}_{relevantly\text{-}similar} \succ \text{avg-sim}_{all\text{-}cases}$$

In other words, our hypothesis was that overall, the *top-tier* cases are more similar to a problem situation than the *relevantly-similar* cases, which are more similar than *all-cases*.[3] For conducting a matched-pairs t-test, we calculated for each of the 147 cases the difference of the average-similarity $\Delta_{top\text{-}tier,all\text{-}cases}$. If all variance in the difference between *all-cases* and *top-tier* cases is due to chance, the frequency distribution of Δ will be a Bell curve with mean 0. However, when we plot the frequency distribution of $\Delta_{top\text{-}tier,all\text{-}cases}$ (see Fig. 1), the curve "leans to the right" (is skewed), and the mean $\mu \approx 0.007$ is greater than 0. Accounting for variance and sample size, a t-test calculates how confidently one can be that the differences observed are not random, and that the average-similarity over *top-tier* cases is greater than over *all-cases*. We found the results of the t-test to be statistically significant ($p = 0.00$). In other words, we found empirical evidence that the more similar cases are in the factor model, the more similar they are in the vector-space model.

[3]The following description will focus on the test between *top-tier* and *all-cases*, the other two relations were established in the same way.

4 Initial Experimental Results

4.1 Experiment Design

In order to assess the suitability and limitations of purely inductive learning methods for the given problem, we implemented a set of machine learning algorithms that seemed to be particularly suitable for the learning problem at hand, and ran some initial experiments. Our data set was the collection of 147 trade secret cases from CATO's Case Database represented by the applicable factors and the corresponding full-text opinions. The opinions were converted into feature vectors as described in Section 3, which could be used as an input to the algorithms.

We separated the cases into test and training sets for carrying out k-fold ($k = 5$) cross-validation experiments (Mitchell 1997). As an evaluation measure, we used the accuracy of the algorithms, which is defined as the percentage of test instances classified correctly. The drawback of this measure is that it favors labeling cases as negative when positive instances in the training set are scarce. Precision (percentage of instances labeled as positive where a factor applies) and recall (percentage of instances where a factor applies that were recognized as positive) are the common measures in IR, but only focus on collecting the relevant documents in a collection, which is not the primary goal in a classification task. We therefore decided to take accuracy as the main criterion, but in addition to interpret the results in the light of precision and recall over all factors.

To find a baseline for the algorithms' performance, we used a random function. Without any prior knowledge, and not using any kind of learning, training examples are classified by chance, and with a probability of 0.5 labeled as positive instances of the factor. This strategy achieved respectable values for recall - since even for the most difficult factors, it retrieved about half of the relevant cases. For practical use, however, this retrieval strategy would not be acceptable. We also included a more informed strategy, which takes into account that many of the factors have only very few positive examples. All cases presented to the system were labeled as "factor does not apply". Although this strategy achieved excellent accuracy, recall is zero, and precision undefined. Thus, both strategies are not acceptable for practical use, but show what level of performance can be achieved without further knowledge.

4.2 Algorithms

We are working with raw texts, which are converted to a high-dimensional feature/value-vectors, and not with a symbolical representation or even keywords. This has an impact on the methods and algorithms we can use. The most widely used inductive learning methods, in particular decision trees and rule induction algorithms, are particularly suitable for problems where the number of features is relatively small and has discrete values; thus, they are not appropriate for our problem. Likewise, training neural networks using the backpropagation learning rule is very time consuming and does not grow linearly with the number of input values. We thus focused on methods that were derived from IR techniques, namely *Rocchhio* and TFIDF-Prototype. In addition, we tried a set of algorithms that were proved theoretically to be particularly suitable for large numers of input features and was used sucessfully in related problems. These algorithms are Weighted-Majority, Winnow (Golding & Roth 1996), Exponentiated Gradient and Widrow-Hoff (Callan 1996). The learning rules for those algorithms are listed in Appendix A.

In addition, we ran a set of experiments with prind, a probabilistic version of TFIDF-Prototype (Mitchell 1997; Joachims 1996), and a naive bayesian classifier using Libbow.[4]

4.3 Results

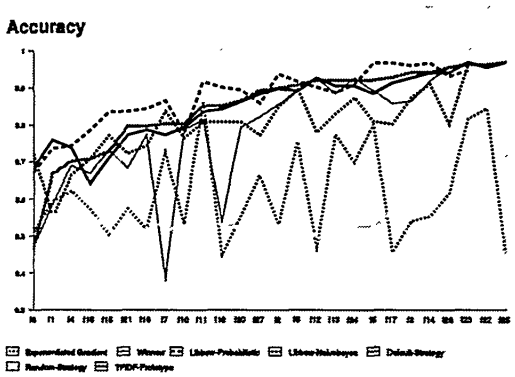

Figure 4: Results of the experiments

The results of our experiments are displayed in Fig. 4. In this, and the following figures, we list CATO's factors on the x-axis and plot the accuracy of the algorithms over the factors. The factors are arranged in decreasing order by the number of cases in which they apply; the leftmost factor F6 (Security-Measures) applies to 79 cases in the case-base, while the rightmost factor, F25 (Info-Reverse-Engineered) applies to only 5 cases.

Figure 4 shows that the machine learning algorithms we tested do not provide a good solution. No algorithm achieves a consistently high accuracy and at the same time desirable precision and recall. We did not observe a trade-off between precision and recall, as one would expect from information retrieval (Turtle 1995). Instead, algorithms did well either on accuracy, with weak performance on precision and recall, or had fair precision and recall rates, but low accuracy. Since many algorithms were designed with the goal to maximize accuracy (and not precision or recall), it is to be expected that they tend to classify too many examples as negative, in particular for factors that only have very few positive examples.

The algorithms did not find the cases to which factors apply for which there were very few training examples, perhaps because the cases which share the infrequent factor with the problem situation tend to have multiple other factors as well. We suspect these other factors "overshadow" the rare factor, in that they are discussed more thoroughly, and thus, the corresponding terms are weighted stronger in the document vectors.

5 Hypotheses on Applying Domain Knowledge for Document Classification

As illustrated in Sec. 2, determining which factors apply in a case is a hard problem which, in spite of ongoing research in different fields, remains unsolved. Probably the ideal solution would be a natural language understanding system. The complexity of legal case opinions, however, is still beyond the grasp of currently available systems. The experiments presented in Sec. 4 show that also a purely inductive approach is not sufficiently powerful. Thus, in the ongoing research on this project presented here, we will employ methods from different research fields, integrate background knowledge about the definition of and the relations among the concepts, and add domain knowledge about the structure and elements found in the legal opinions. The following paragraphs introduce our ideas in the form of hypotheses. To illustrate the ideas, we will use a case from CATO's Case Database, *Aries Information Systems v. Pacific Management Systems.* In this case, the court has to decide whether former employees misappropriated plaintiff's trade secret by developing a similar software system in their new employment. It is represented by factors F4 (Agreed-Not-To-Disclose), F6 (Security-Measures), F15 (Unique-Product) and F18 (Identical-Products).

Adding Domain Knowledge to Inductive Algorithms: We conjecture that the performance of learning algorithms for classifying opinion texts can be improved by adding domain knowledge. The sources of this domain knowledge are the factors and their interrelations in CATO's model, information about the general structure and content of opinions,

[4] Libbow is a code package developed by Andrew McCallum for text classification experiments. It is available from `http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html`.

as well as domain knowledge captured in a dictionary. Our goal is to combine inductive methods that learn from previous examples and analytical learning methods that use prior domain knowledge to deduce general hypotheses (Mitchell 1997) and to apply methods that use background knowledge to guide the inductive learning process (Aronis, Provost, & Buchanan 1996). As inputs, the inductive algorithms typically receive a very highly dimensional vector over all the words in the collection. Domain knowledge about the definition and meaning of factors can also help to initialize the algorithms by using textual descriptions of factors, thus overcoming the lack of training instances. Finally, limiting the algorithm's inputs to the most predictive words can filter out noisy information that otherwise may weaken performance.

Identifying Legal Concepts: A more knowledge-rich representation of documents may also lead to better performance. Domain concepts, citations or document structure, which go beyond mere words have not yet been applied in machine learning on texts. By identifying legal concepts mentioned in the document and screening out irrelevant terms, a system will more reliably find cases dealing with similar legal issues, since the document matching and ranking can focus on the pertinent parts of the document vectors. Here is the initial part of the (normalized euclidian length) vector representing the *Aries* opinion:

```
aries ari 0.0556161908023696 poba 0.03814573920559
42 pacific 0.0120623814326339 softwar 0.0073975981
6737666 respondent 0.00729206190657269 ariinformat
ion 0.00646699893050809 system 0.00593029306349872
eveloper0.00564256177002941 appellant 0.0052186578
9587887 economicvalue 0.00455113442550662 reasonab
leffort 0.00442244280609885 softwaresystem 0.004252
16254517972 popovich 0.00388019935830485 fami 0.00
378743714371726 client 0.00369879908368833 compute
rsoftwar 0.00368536900508237 rock 0.00359337630188
```

This document representation of the *Aries* opinion, used by traditional information retrieval methods, contains a number of highly weighted terms that are characteristic of the factors that apply in the case, and the issues raised, like `reasonableffort` which indicates that the opinion discusses whether plaintiff had taken measures to keep the information secret. There are, however, a number of terms that are not directly connected with the factors, like `softwaresystem`, which should be filtered out of the representation.

Assigning Issues First: Another technique potentially useful for improving the performance of a learning algorithm is assigning issues first, and then factors. It should reduce the hypothesis space, focusing on the factors related in the Factor Hierarchy to the applicable issues. Similar to the approach in EBNN (Mitchell, O'Sullivan, & Thrun 1994) the idea is to decompose the problem into less complex subproblems by first identifying issues in a text and then use this as input to subsequent classifiers which identify more specific factors presented in the text. We believe that (1) deciding the issues a case involves is easier than assigning the more detailed factors, and (2) once issues have been identified, the Factor Hierarchy's relations of issues to factors can be used to focus learning on particular factors.

Accounting for Factor Meaning and Relations: Taking into account what the factors mean should support predictions of the likelihood that a factor applies in situations where training instances are scarce. In the CATO database, some factors apply only in a few cases. For each of factors F3, F5, F14, F22, F23, F25, and F26 there are 5 to 7 examples. These numbers are too small for learning algorithms to generalize well from the examples. In fact, a reasonable strategy for achieving good classification accuracy would be always to predict that the factor does not apply in a text. (We actually observed this behavior in the experiments described in Section 4.) In order to overcome the problem of small numbers of training examples, we intend to employ heuristics based on the meaning of factors and on their relationships expressed in the Factor Hierarchy and in cases in CATO's Case Database. Based on either semantic or empirical grounds, one may infer that certain factors naturally occur together (or not). By querying the case base, one can find as an empirical matter if particular pairs of factors have ever co-occurred in the past. The Factor Hierarchy expresses semantic relations among factors via more abstract parent concepts (i.e., issues and other high-level factors.) The relation among factors and high-level factors can be consistent or inconsistent, strong or weak. The idea is to use these relationships to find confirming evidence for a factor in a text and to test the credibility of an assignment of factors.

As an example, take factor F23, Waiver-Of-Confidentiality, which applies to only six cases in the database. The concept is very hard to find, since it can involve diverse fact situations. It is much easier to train a classifier to identify factors F1, Disclosure-In-Negotiations, or F21, Knew-Info-Confidential, which apply in many more cases. F23 has some useful relationships with these easier-to-learn factors. There would rarely be a waiver of confidentiality without a disclosure in negotiations. It would also be unlikely to find a waiver of confidentiality in a case where there is a non-disclosure agreement (represented by F4) or knowledge that information is confidential. These relationships can be expressed as probabilities, and then be used with Bayes' Law to help assess whether F23 is in fact present:

$$\mathrm{P}(F1|F23) = high, \mathrm{P}(F4|F23) = low, \mathrm{P}(F21|F23) = low$$

We will also test whether the factors assigned to a case are consistent, by making use of the fact that some of the factors are strong evidence for a high-level factor and others are strong evidence against the same high-level factor. If the factors assigned to a text are inconsistent, the classification may have to be revised. Of course, there are some exceptional cases where despite this semantic conclusion, the factors do in fact co-occur. As a partial check for exceptions,

if an inconsistent factor assignment is detected, the program will check if any other cases in the case base have the same pattern. If there is such a case, then the inconsistency of the factor assignment will be ignored in light of the historical precedent. Moreover, if it is possible to determine from analysis of the text which side won, plaintiff or defendant, then a system can also reason about whether the assigned factors are consistent with the result. If only pro-defendant factors were found, but plaintiff won, the factor assignment is suspect. Further uses of knowledge about meaning and relations among factors can be found in the following sections.

Applying Knowledge About Opinion Texts: Two hypotheses we will consider are fairly specific to legal texts. Identifying quotations from the relevant statutes and restatement commentaries may justify an inference that the court focuses on a particular issue. Many trade secrets cases quote parts of the Restatement of Torts' definition of a trade secret and its conditions for when plaintiff's information is a trade secret. In *Aries*, the court cites the Minnesota Trade Secrets Act (Minn.Stat. §325C.01):

> "Trade secret" means information, including a formula, pattern, compilation, program, device, method, technique, or process, that: (i) derives independent economic value, actual or potential, from not being generally known to, and not being readily ascertainable by proper means by, other persons who can obtain economic value from its disclosure or use, and (ii) is the subject of efforts that are reasonable under the circumstances to maintain its secrecy.

One may infer that in *Aries* an issue is whether the information used by defendant was a trade secret, and whether plaintiff had taken appropriate means to protect the secrecy of the information. The first issue corresponds to the legal-conclusion factor F101 (Info-Trade-Secret), which we can here assume to favor plaintiff since the next issue raised relates to high-level F102 (Efforts-To-Maintain-Secrecy). It is therefore likely that the case will have factors related to those issues.

Information about the structure of the text of an opinion can be employed beneficially in identifying important and less relevant sections. While not following a rigid format, legal opinions do have a basic structure: first the facts of the case are described, then the applicable law is stated and applied to the facts, and finally, the legal consequences are concluded. If a system can detect this structure, the information contained in the different parts can be used more specifically. The applicable law is usually a good indicator for the issues raised, while the factors themselves are mainly determined by the facts of the case. Paragraphs discussing the amount of damages payable to plaintiff are not related to the factors, and should be ignored for our task, since they introduce irrelevant information, or noise.

It is plausible that identifying cases cited in an opinion text can help assign factors to opinion texts where it is known which factors apply to the cited case. This information should support an inference that the citing case deals with similar issues and has similar factors to those which apply in the cited case. In the *Aries* case, for example, the court cites three cases, *Christopher v. DuPont* and *Electro Craft v. Controlled Motion* in the same sentence, and *Jostens v. National Computer Systems* in a later paragraph. The case's two main issues are whether the information is a trade secret and whether the defendant wrongfully obtained it. For the first issue, the court cites *DuPont*, which has factors F6 and F22, and *Electro Craft*, which has factors F4, F5, F6, F15, F18. Both cases have one high-level factor in common, F101 and the shared factor related to this is F6. The section in the text in which defendant's obtaining the information is examined contains a reference to *Jostens*. In *Jostens*, factors F4, F5, F6, F17, F20, F27 apply, which all belong to high-level factor F101. Only factor F4 also relates to the high-level factor F115, Notice-of-Confidentiality, which is not subsumed by Info-Trade-Secret in the Factor Hierarchy. This suggests that *Aries* has F6 (the only factor shared by *Christopher* and *Electro-Craft*) relating to the issues whether the information was a trade secret, and F4, the factor in *Jostens* related to the issue whether there was a confidential relationship between the parties.

Finally, it is likely that an appropriate combination of some or all of the knowledge-based methods for learning to assign factors to opinion texts and some of the known inductive or statistical learning algorithms can improve performance. As we gain more experience with the strengths and weaknesses of the various methods in relation to the kinds of texts we encounter, we should be able to achieve some improvements by composing various methods.

6 Conclusion

In this paper, we presented evidence for a connection between the Factor Model underlying case-based argumentation in CATO, and the Vector-Space Model developed in IR. This suggests that the methods developed in IR can be employed beneficially in deriving a symbolic representation of legal cases from text. However, our experiments with machine learning methods using 147 cases from CATO's Case Database as training data showed that domain-independent inductive methods are not powerful enough reliably to assign factors to opinion texts. In order to overcome this problem, we hypothesize that including knowledge about the domain and about the use of cases in legal argumentation can be beneficial.

If successful, this research can help increase the practical impact of legal case-based reasoning methods, as it facilitates building case bases. Thus far, indexing is a laborious manual effort, that impedes building case-based reasoning systems. A system that can use domain knowledge and a moderately sized case base to assign indices to new cases will increase significantly the availability of legal case-based

reasoning systems.

An automatic indexing system in combination with IR can also improve legal research, as suggested, e.g., in (Turtle 1995). The documents retrieved by a conventional IR system are presented to the user ordered by the system's relevance scores, which depend largely on the query and do not directly reflect how useful the retrieved cases are for a legal argumentation task. If a system could assign factors or issues to the retrieved cases and then arrange them in a HYPO-style claim lattice, this would be a valuable addition for a professional in assessing usefulness of the retrieved documents.

Acknowledgements

We would like to thank Vincent Aleven for his invaluable contributions to the research presented, which would not have been possible otherwise, and for giving us access to the CATO system and the CATO Case Database.

References

Aleven, V., and Ashley, K. 1996. How Different is Different? Arguing about the Significance of Similarities and Differences. In *Proceedings of the 4th European Workshop on Case-Based Reasoning*, 1–15.

Aleven, V., and Ashley, K. 1997. An Empirical Evaluation of an Intelligent Learning Environment for Case-Based Argumentation. In *Proceedings of International Conference on Artificial Intelligence in Education.*

Aronis, J.; Provost, F.; and Buchanan, B. 1996. Exploiting Background Knowledge in Automated Discovery. Technical report, Intelligent Systems Laboratory, University of Pittsburgh. ISL 96-5.

Ashley, K. 1990. *Modeling Legal Argument, Reasoning with Cases and Hypotheticals.* MIT-Press.

Blum, A. 1995. Empirical Support for Winnow and Weighted-Majority based algorithms: results from a calendar-scheduling domain. In *Proceedings of the 12th International Conference on Machine Learning.*

Callan, J.; Croft, W.; and Harding, S. 1992. The INQUERY Retrieval System. In *Proceedings of the 3rd International Conference on Database and Expert Systems Applications*, 78–83.

Callan, J. 1996. Document Filtering with Inference Networks. In *Proceedings of the 19th Annual International ACM SIGIR Conference.*

Cowie, J., and Lehnert, W. 1996. Information extraction. *Communications of the ACM* 39(1):80–91.

Craven, M.; Freitag, D.; McCallum, A.; Mitchell, T.; Nigam, K.; and Yank Queak, C. 1997. Learning to Extract Symbolic Knowledge from the World Wide Web. Submitted to ML-97.

Daniels, J. 1996. Retrieval of Passages for Information Reduction. PhD Proposal, University of Massachusetts, Amherst.

Frankes, W., and Baeza-Yates, R. 1992. *Information Retrieval - Data Structures & Algorithms.* Prentice-Hall.

Freitag, D. 1997. Machine Learning for Information Extraction from Online Documents. PhD Proposal, Carnegie-Mellon University, Pittsburgh.

Golding, A., and Roth, D. 1996. Applying winnow to context-sensitive spelling correction. In *Proceedings of the 13th International Conference on Machine Learning.*

Joachims, T. 1996. A Probabilistic Analysis of the Rochio Algorithm with TFIDF for Text Categorization. Technical report, Carnegie Mellon University. CMU-CS-96-118.

Kivinen, J., and Warmuth, M. 1994. Exponentiated Gradient versus Gradient Descent for Linear Predictors. Technical report, University of California Santa Clara.

Lang, K. 1995. Learning to Filter Netnews. In *Proceedings of the Twelfth International Conference on Machine Learning.*

Lewis, D.; Shapire, R.; Callan, J.; and Papka, R. 1996. Training Algorithms for Linear Text Classifiers. In *Proceedings of the 19th Annual International ACM SIGIR Conference.*

Lewis, D. 1992. *Representation and Learning in Information Retrieval.* Ph.D. Dissertation, University of Massachusetts, Amherst.

Mitchell, T.; O'Sullivan, J.; and Thrun, S. 1994. Explanation-Based Learning for Mobile Robot Perception. In *Proceedings of the Eleventh International Conference on Machine Learning.*

Mitchell, T. 1997. *Machine Learning.* Mc Graw Hill.

Papka, R.; Callan, J.; and Barto, A. 1996. Text-Based Information Retrieval Using Exponentiated Gradient Descent. In *Neural Information Processing Systems.* CIIR technical report IR-101.

Pazzani, M.; Muramatsu, J.; and Billsus, D. 1996. Syskill & Webbert: Identifying Interesting Web Sites. In *Proceedings of the 13th National Conference on Artificial Intelligence*, 54–61.

Rissland, E.; Skalak, D.; and Friedman, T. 1993. Case Retrieval Through Multiple Indexing and Heuristic Search. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence.*

Rose, D: 1994. *A Symbolic and Connectionist Approach to Legal Information Retrieval.* Hillsdale, NJ: Lawrence Earlbaum Publishers.

Smith, J., and Gelbart, D. e. 1995. Artificial Intelligence and Legal Discourse: The Flexlaw Legal Text Management System. *Artificial Intelligence and Law* 2(1):55–95.

Turtle, H. 1995. Text Retrieval in the Legal Word. *Artificial Intelligence and Law* 2(1):5–54.

A Algorithms Used

The algorithms discussed in this appendix all are used for training linear classifiers (Lewis *et al.* 1996). That means, the prediction of the system is based on a classification value *cv* that is calculated as the dot product between the document vector (where absent terms have weight 0) and the weight vector w representing the classifier. Some methods also include a fixed or learned threshold. In the escription of the algorithms, we assume that document i has the feature/value vector $\mathbf{d}_i$, and that $y_{i,k}$ indicates whether factor f_k applies. If the factor applies, the value of $y_{i,k}$ is 1, if not, the value is 0.

A.1 TF/IDF-Prototype

For each factor f_k, a prototype vector for the concepts "factor applies" and "factor does not apply" is derived by calculating an average vector for the document vectors of the positive instances in C_k and the negative instances in $\overline{C}_k$,

respectively. The classification of new instances from the test set follows the more similar prototype (using the cosine measure). The classification-value cv for a new case i is difference of the cosine between the document vector d_i and the prototype for C_k and the cosince between d_i and the prototype for $\overline{C}_k$.

$$\text{proto}-\text{pos}_j = \frac{\sum_{d \in C_k} d_i}{N_{\mathcal{F}}}, \quad \text{proto}-\text{neg}_i = \frac{\sum_{d \in \overline{C}_k} d_i}{N - N_{C_k}}$$

In our experiments, this algorithm had the best performance. It did well for factors that apply to many cases, but failed in finding the positive instances for factors that apply to only very few cases.

A.2 Rocchio

Rocchio was invented early in IR research as a method for relevance feedback, where the goal is to modify a user's query so that retrieves more documents like those selected. It is very similar to the TF/IDF-Prototype algorithm. An initial query, resp. vector representing the concept corresponding to factor f_k, is modified by taking into account documents judged as relevant and non-relevant (C_k and $\overline{C}_k$) such that the new query retrieves more documents like those judged relevant and less like those judged irrelevant. Currently, we do not use an initial query vector, though. The importance assigned to the initial query (α), to the relevant documents retrieved (β) and to the irrelevant documents (γ) can be set flexibly. The classifier vector for factor f_k is calculated by:

$$\text{rocchio}_i = \alpha q v_i + \beta \frac{\sum_{d \in C_k} d_i}{|C_k|} - \gamma \frac{\sum_{d \in \overline{C}_k} d_i}{|\overline{C}_k|}$$

A.3 Widrow-Hoff

Least Mean Squares, or Widrow-Hoff, is a single-neuron neural network model trained in online mode, updating the weight vector subsequently with the training examples. The weight update rule minimizes the squared error by using its gradient, similar to the learning rule for the backpropagation algorithm. The weight uppdate rule for the j-th component of the classifier for factor f_k after case i using learning rate η is:

$$\text{wh}_{i+1,j} = \text{wh}_{i,j} - 2\eta\,(\text{wh}_i \cdot \text{d}_i - \text{y}_{i,k})\,\text{d}_{i,j}$$

A.4 Exponentiated Gradient

The Gradient Descent rule, used in many neural network applications, was the starting point for the Exponentiated Gradient learning algorithm. For each training instance, the weights of the classifier are adjusted such that the squared error for the training instance is minimized. Like Widrow-Hoff, it is a single-neuron learning algorithm, but uses a multiplicative update rule. Theoretical results suggest it is preferable in situations where only few input components are relevant for the predictions (Kivinen & Warmuth 1994). Experiments have shown that Exponential Gradient can improve retrieval performance in IR systems (Papka, Callan, & Barto 1996). The learning rule for the j-th component of the classifier after case i with learning rate η:

$$eg_{i+1,j} = eg_{i,j}^{-2\eta\,(\text{eg}_i \cdot d_i - y_{i,k})\,d_{i,j}}$$

A.5 Winnow

Winnow can be thought of as an algorithm in which the "opinions of experts" that give predictions for the classification of an example are combined (Blum 1995), and a case is classified depending on whether the combined, weighted "votes" exceed a given threshold. In our application, the "experts" are the terms in the documents, they give a positive vote for a document if the term is present in the document. Initially, the components of the classifier vector have equal weights (usually summing up to 1), and a threshold (usually 1) is given (Golding & Roth 1996). For each instance in the training set, the following Vote/Update Circle is performed, depending on whether a training instance is classified correctly or not, the weights of the terms present in that instance are updated multiplicatively (typically, the weights are doubled in the promotion step, and halved in the demotion step):

Vote
  foreach word j
    if $d_{i,j} > 0$
      then $vote$ += $winnow_j$
      else $vote$ -= $winnow_j$
  if $vote > threshold$
    then $pre = 1$ /* predict factor applies */
    else $pre = 0$ /*predict factor does not apply */
Weight Update
  if $pre \neq y_{i,k}$
    then /* prediction is not correct */
      if $y_{i,k} = 1$
        then /* factor applies */
          foreach j where $d_{i,j} \not> 0$ do $winnow_j$ /= 2
          foreach j where $d_{i,j} > 0$ do $winnow_j$ *= 2
        else /* factor does not apply */
          foreach j where $d_{i,j} > 0$ do $winnow_j$ /= 2
          foreach j where $d_{i,j} \not> 0$ do $winnow_j$ *= 2
    else /* prediction is correct */
      do nothing

We believe that additional domain knowledge in the form of textual descriptions of the factors can be used for initializing the weights and make up for the limited number of training instances available.

A.6 Weighted Majority

Weighted Majority (Mitchell 1997; Blum 1995) is another multiplicative update algorithm very similar to Winnow. The major difference to Winnow is that the weights of those terms make an incorrect vote are decreased, but never increased, no matter whether the overall prediction is correct or not.