Toward Adding Knowledge to Learning Algorithms for Indexing Legal Cases

Stefanie Brüninghaus and Kevin D. Ashley

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

Abstract

Case-based reasoning systems have shown great promise for legal argumentation, but their development and wider availability are still slowed by the cost of manually representing cases. In this paper, we present our recent progress toward automatically indexing legal opinion texts for a CBR system. Our system SMILE uses a classification-based approach to find abstract fact situations in legal texts. To reduce the complexity inherent in legal texts, we take the individual sentences from a marked-up collection of case summaries as examples. We illustrate how integrating a legal thesaurus and linguistic information with a machine learning algorithm can help to overcome the difficulties created by legal language. The paper discusses results from a preliminary experiment with a decision tree learning algorithm. Experiments indicate that learning on the basis of sentences, rather than full documents, is effective. They also confirm that adding a legal thesaurus to the learning algorithm leads to improved performance for some, but not all, indexing concepts.

1 Motivation

Since almost all cases and other materials in the law are written, dealing with text has long been a focus of AI and Law research. At this year's and the previous ICAIL conferences, many sophisticated approaches have been presented for retrieving (Smith *et al.* 1995; Greenleaf *et al.* 1997), summarizing (Moens, Uyttendale, & Dumotier 1997), or filtering (Schweighoer, Winiwarter, & Merkl 1995) legal documents, to mention only a few applications. WestLaw and other commercially highly successful information retrieval systems are also available for legal professionals.

All these systems attempt to facilitate access to legal texts in order to help find the best cases for an attorney's information need. Where this involves retrieving cases for more advanced reasoning, like legal argumentation, however, merely retrieving documents is not sufficient:

Practitioners often compare partially matched cases, reason about the significance of cases, evaluate problems in the context of multiple cases, and combine cases and rules. AI representations of reasoning with cases are more effective to support these tasks. Legal case-based reasoning systems have been designed and implemented for a number of different problems, for instance worker's compensation (Branting 1991), trade secret law (Aleven & Ashley 1997), or tax problems (Rissland, Skalak, & Friedman 1993). They show great potential for a wider use in legal practice and education (Aleven 1997).

These systems still, however, have not bridged the gap between the opinion texts and the corrsponing symbolic AI representation. Their dependence on manually indexed cases is a serious obstacle. Since legal cases are long and complex, manual indexing is time-consuming and expensive. The wider availability of these systems has been prevented by the significant cost and effort for case-base development and maintainance. Surprisingly, little research has attempted to overcome this problem, with a few notable exceptions like SPIRE (Daniels & Rissland 1997).

In this paper, we discuss our progress toward automatically representing legal cases for their use in our CBR systsm CATO. We present our system SMILE[1], a machine learning approach, which employs a smaller-sized, manually-indexed collection of case squibs to help bootstrap the development and maintainance of a larger case-base.

In the following section, we will give a short introduction to our CBR application, and discuss available methods for index learning. We will then show why these methods are not appropriate, and suggest alternatives. We have experimented with some of the suggested improvements, and will present encouraging results.

ICAIL-99 Oslo, Norway Copyright (c) 1999 ACM 1-58113-165-8/99/6 $5.00

[1] SMart Index LEarner

2 Problems with Learning to Index Legal Cases for Case-Based Argumentation

When trying to convince the court to rule in a party's favor, lawyers analogize the problem to and distinguish it from previously decided cases. In doing so, they often compare and contrast cases in terms of prototypical fact patterns, which tend to strengthen or weaken their client's claim.

A model of this case-based argumentation in law has been implemented in CATO (Aleven 1997; Aleven & Ashley 1997) an instructional environment to teach argumentation skills to law students. CATO uses a set of 26 abstract fact patterns, or *factors*, to compare and contrast cases by means of eight basic argument moves. More advanced arguments can be made by symbolically reasoning about the significance of distinctions. These arguments use CATO's Factor Hierarchy, which contains additional knowledge about higher-level legal issues and concerns. CATO can generate arguments for any combination of cases from its Case Database of about 150 cases. In addition to the factor representation of the cases, it has a collection of case briefs, or squibs. Students use the cases in the Case Database to test generalizations, or theories about the domain. They also practice basic argument moves, and compare their own written arguments to those generated by CATO. The instruction with CATO has been shown to be effective when compared to a human instructor. It would be desirable to develop case bases for other domains.

A promising approach is to make use of recent progress in Machine Learning (ML): In the last years, learning from text has become a focus in this research field. In various approaches, learning algorithms have been used successfully to classify text documents under abstract concepts (Sahami *et al.* 1998), and progress has been made towards extracting information from text (Craven *et al.* 1998).

Motivated by these advances in research, we have taken up the idea of using ML to assign factors to legal cases based on the examples in CATO's Case Database (Brüninghaus & Ashley 1997). Each factor can be seen as a concept, and the opinions of the cases in the Case Database are the positive and negative examples, depending on whether the factor applies or not.

We experimented with a number of (existing) learning algorithms, which had been used with success for other text collections. All of these algorithms represent documents as a bag-of-words. Text is split into single words, or short sequences of words (n-grams), which are treated as independent tokens. Weights are assigned based on statistical properties of the collection and the document. The algorithms use advanced statistical methods, like bayesian models, to derive a classifier by weighting the examples' vectors. They work best on large collections of rather simple documents, like archives of usenet news.

We found that these algorithms did not perform as we had hoped for our text corpus (Brüninghaus & Ashley 1997). For most factors, they could not learn classifiers to discriminate reliably between positive and negative instances. The two main reasons for this failure, we think, are: (1) deficiencies in the bag-of-words representation, and (2) the complexity of legal opinions, in particular in relation to the number of training instances available.

Despite the fact that it is widely used in Information Retrieval, the bag-of-words or vector representation is not powerful enough to capture the subtleties of the language used in legal opinions. It does not consider information about the order and relation of words, and removes all propositions, adverbs and conjuncts.

The legal opinions we worked with are also unlike some of the short and relatively simple texts employed in other domains. They are very long and contain too much irrelevant information that tends to mislead classifiers. If the number of training instances is large enough, statistical methods can filter out irrelevant words, but the number of examples would have to be magnitudes larger than CATO's Case Database.

3 Recognizing Factors in Opinion Texts

A brief introduction to CATO's domain with examples shows the difficulties more clearly and helps to illustrate the suggested improvements. The documents we seek to classify automatically deal with trade secret law.

Trade secret law protects the owners of confidential commercial information from improper discovery or misappropriation of the information by competitors. In a lawsuit, the court must decide whether the information was protectable as a trade secret and whether it actually was misappropriated. In its opinion, the court records the facts of the case, the applicable law, and the reasoning that justifies its final decision about the outcome. The opinions frequently are published and may be cited in subsequent decisions.

The opinion texts are long, usually between two and twenty pages in length. They involve multiple topics. Often, the trade secret claim is not the only problem discussed. For instance, the court may deal with jurisdictional and procedural issues, or even other claims. Even within the discussion of the trade secret claim, not all passages are as relevant for classifying the text. For instance, the court may discuss the length of an injunction against defendant's further use of the information. These additional issues make it more difficult for a computer-based indexing technique.

Legal opinion texts are also characterized by very complex prose. Judges tend to express themselves in exceptionally long sentences with many branches. They often use domain-specific expressions and a unique style. The vocabulary and word-distributions differ from "regular" English, and many terms have a specific meaning in a legal context.

As discussed below, one way of capturing the specific word usages are legal thesauri (Burton 1992; Statski 1985). A legal thesaurus lists alternative terms or phrases for a legal concept which are not readily found in an all-purpose dictionary.

3.1 Example Factor: F15 Unique-Product

As mentioned above, CATO's factors are the goal concepts for our approach. They indicate the presence of a particular fact pattern in the case. Let us consider one of CATO's factors in more detail, to give a more concrete impression of what we are trying to accomplish in SMILE.

In trade secret law, the allegedly misappropriated information must meet certain criteria to be protectable as a trade secret. If, for instance, a similar product is available from other manufacturers and discloses the information, the information may be generally available and may not be protected against use by competitors.

In CATO this fact pattern is represented by a factor favoring plaintiff: F15, Unique-Product. The following definition is provided when working with CATO:

> Plaintiff was the only manufacturer making the product.
> This factor shows that the information apparently was not known or available outside plaintiff's business. Also, it shows that plaintiff's information was valuable for plaintiff's business.

Some typical examples of sentences indicating that F15 applies, found in cases in CATO's Case Database, are:

- Innovative introduced evidence that Panl Brick was a unique product in the industry. (from *Innovative v. Bowen*)
- It has a unique design in that it has a single pole that a hunter can climb instead of having to climb the tree. (from *Phillips v. Frey*)
- Several features of the process were entirely unique in transistor manufacturing. (from *Sperry Rand v. Rothlein*)
- The information in the diagram is not generally known to the public nor to any of Tri-Tron's competitors. (from *Tri-Tron v. Velto*)
- It appears that one could not order a Lynchburg Lemonade in any establishment other than that of the plaintiff. (from *Mason v. Jack Daniel Distillery*)

3.2 Evidence for Factors

These examples illustrate how evidence for factors is typically found in a few places, in the form of sentences or short passages. Moreover, the sentences relevant to a factor generally follow a small number of patterns, focus on a limited set of issues and use similar wording. Experts can relatively easily identify the passages and sentences that pertain to the factors assigned when indexing new cases. In fact, they often underline the sentences in the text relevant for a factor.

It is clear from the examples, however, that deciding whether a factor applies in a case requires information more detailed than vectors over the content-words. One needs to look at the sentence or passage as an entire context. In legal texts, moreover, the negation or restriction of statements is very important for the legal consequences. In the *Mason* case example, the negation of "order" is crucial: "It appears that one could not order a Lynchburg Lemonade in any establishment other than that of the plaintiff." After all, if one could order the product somewhere else, it would not be unique.

This example suggests that a better representation of the examples is needed. In other domains, for instance for the classification of newswire articles, removing stopwords is sensible. These terms will not contribute much, if at all, to the performance of a classifier. By contrast, in legal opinion texts certain words, normally treated as stopwords, are too important to remove.

Thus, an algorithm for learning to assign indices has to have a number of properties. A successful approach has to address the problem that legal documents are very long, complex, and contain a lot of irrelevant information. The learning algorithm has to be applicable to rather small numbers of instances, rather than requiring thousands of training examples. It has to allow for a more powerful representation than the commonly used bag-of-words methods. Finally, it must be possible to include domain specific background knowledge, like a legal thesaurus.

4 Better Techniques for Finding Factors

Based on the requirements outlined above, our approach in SMILE is to:

1. use sentences instead of entire documents as examples,
2. use algorithms that learn rules, rather than weighting vectors over the entire vocabulary, and
3. add knowledge to the induction process.

4.1 Focus on Smaller Units of Information

Rather than running learning algorithms on full-text documents, it is more apprpriate to break up the document into smaller sentences. In our application, sentences are the appropriate unit to convey the relevant information.

Marking up the sentences has three advantages. (1) It allows focussing on smaller and more relevant examples, namely the marked-up sentences or passages pertaining to a factor. This requires one manually to mark up the sentences referring to a factor. The marked-up sentences will, in effect, become the examples for training the learning algorithm. Though a manual process, this step adds little work for an expert indexing the cases.

(2) The use of marked-up sentences instead of the complete opinions as training examples offers computational advantages. Methods that combine a large number of possibly useless features are not appropriate. For learning to classify full-text documents, algorithms that take a vector over the entire vocabulary have been used successfully, but they work best with large numbers of examples. Many symbolic learning algorithms (in particular those that use a more powerful representation) get bogged down by large numbers of attributes. Reducing the complexity of the examples allows us to add knowledge from parsing.

(3) Finally, learning from marked-up sentences facilitates including domain knowledge, in the form of a domain-specific thesaurus. If the examples are sentences, the knowledge contained in a thesaurus can be better applied in the relevant place.

A similar idea, namely to focus on sub-passages (although not sentences) within a document, has been used before, in the SPIRE system (Daniels & Rissland 1997). The goal of SPIRE is also to assist a user in indexing cases for a case-based reasoning system. Cases in SPIRE are not indexed by factors, but represented as frames. Many of the features involve quantitative information such as the amount of money to be paid or the length of a bankruptcy plan. For each indexing concept, SPIRE has a library of manually marked passages drawn from its case opinions. It uses this information to identify the most promising passages in a new document to be indexed.

To accomplish that goal, SPIRE forwards the marked passages, as well as the document to be indexed, to the INQUERY system. With its internal relevance feedback algorithms, INQUERY generates a new query from the marked passages. This query is used to retrieve the highest ranked passages within the new document. It presents these passages to the user to assist in indexing the new case.

SPIRE has in part motivated us to pursue the idea to work with sentences. The main difference from SMILE is that SPIRE is based on existing and entirely domain-independent information retrieval methods. It does not attempt to modify these methods, and does not rely on background knowledge. It also does not rely on full sentences, but rather on fixed-length sequences of words.

Although SPIRE's simplicity is a strength of the approach, it also leads to limitations. We discussed before why factors are hard to find, and what distinguishes legal documents from most other texts. SPIRE does not address the wide expressive variety of legal language or the problem of interpreting negation. Given the adversarial nature of legal discourse, where a judge may discuss the evidence favoring the application of a factor and the evidence favoring a conclusion that it does not apply, this is an important limitation. For instance, SPIRE would not be able to deal with an example like "No other manufacturer offered the product," which is exactly what we intend to overcome.

4.2 Using a Rule Learning Algorithm

We have found that one can manually construct some rules to determine whether a factor applies to a sentence. The rules use only a few, highly relevant features. For example, one such rule is: "if the sentence contains 'unique' and 'product', then f15 applies." Therefore we think, a symbolic algorithm like ID3, which generates a tree-structure with implicit rules, or another rule induction algorithm, is most promising.

Symbolic rule-learning algorithms are generally better suited for smaller sized collections like ours. As noted above, algorithms based on powerful statistical models, in particular those described in (Brüninghaus & Ashley 1997), are less appropriate for our application, since the underlying models require large numbers of examples (in the magnitude of thousands) to work reliably. For the 120 or so cases included in the experiment reported here, we have 2200 sentences in the squibs, but only about 50 to 100 positive examples for each factor.

Decision tree learning algorithms have been used before to classify texts, e.g., newswire articles (Lewis & Riguette 1994; Moulinier, Raskinis, & Ganascia 1996). For these applications, the basic decision tree technique has been quite successful. However, there, it has been sufficient to use only a few, obvious words. For finding factors, more powerful patterns are necessary. As described below, the patterns contain certain stopwords and linguistic information about the role of words. Also, none of these previous approaches attempted to include background knowledge, like a thesaurus.

4.3 Integrating an Application-Specific Thesaurus

Attorneys often use legal thesauri (Burton 1992; Statski 1985), which list synonyms and sometimes definitions for terms used in legal documents. This is probably motivated in part by the fact that judges appear to make an effort not to repeat themselves when drafting their decisions. A learning algorithm by itself can not cope with synonymity. For instance, it cannot infer that "covenant" is another word for "contract." This limitation can be approached by applying a thesaurus to detect synonyms.

Legal publishers like WestGroup maintain some form of a thesaurus internally and use it for purposes of expanding queries. Fortunately, we have the opportunity to employ a copy of the WestLaw Thesaurus for our experiments. It is organized as sets of synonyms, where each word belongs to between one and six of about 20,000 synonym sets. Examples relevant to trade secret law are:

- clandestine concealed disguised hidden secret undisclosed unrevealed
- commodity goods inventory material merchandise product stock supplies
- admission disclosure discovery revelation

5 Design and Implementation of SMILE

5.1 Case Mark-Ups

For the experiments reported here, we marked up the squibs of CATO's cases. An example is the *Mason* case [2], which has the factors F1, Disclosure-In-Negotiations, F6, Security-Measures, F15, Unique-Product, F16, Info-Reverse-Engineerable, and F21, Knew-Info-Confidential. This is a short passage from its squib:

> [f15 f16 Despite its extreme popularity (the drink comprised about one third of the sales of alcoholic drinks), no other establishment had duplicated the drink, but experts claimed it could easily be duplicated. f15 f16] In 1982, Randle, a sales representative of the distillery, visited Mason's restaurant and drank Lynchburg Lemonade. [f1 Mason disclosed part of the recipe to Randle in exchange, Mason claimed, for a promise that Mason and his band would be used in a sales promotion. f1][f21 Randle recalled having been under the impression that Mason's recipe was a "secret formula." f21]

For our classification approach, the sentences bracketed [fxx ... fxx] are positive instances for the factor xx; all other sentences are negative instances. A factor can be considered to apply to a case if at least one of the sentences is a positive example of the factor.

[2] *Mason v. Jack Daniels Distillery*, 518 So.2d 130 (Ala.Civ.App. 1987)

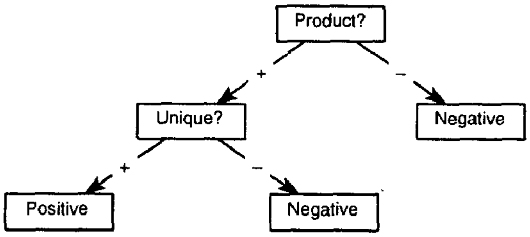


Figure 1: Decision tree for simple sentences

5.2 Decision Tree Induction

These sentences are then used as training examples for a machine learning algorithm.

Ideally, judges would describe the facts of a case in simple and straight forward phrases. They would always use the same words or expressions, never use negation, etc. The the positive (+) and negative (–) examples given to a classifier might look like this:

+ The product was unique.

– His product was identical to plaintiff's.

– The recipe was always locked away in a unique safe.

+ Plaintiff employed unique security measures.

If the sentences were as simple as in this example, applying a decision tree algorithm would work perfectly. In inducing the tree, an algorithm like ID3 recursively selects the attribute that best discriminates between positive and negative examples and splits up the training set, according to whether this attribute applies. The process is repeated until it has a set of only positive or negative examples. Here, ID3 would first split up the examples into those that have the word "product", and those that don't. It would then try to find a way to distinguish between the first and the second example, and select the word "unique". The corresponding decision tree is shown in Figure 1.

Of course, judges do not restrict their factual descriptions in this way. In the next section, we discuss how adding knowledge from a legal thesaurus and adding linguistic knowledge may help.

5.3 Adding a Thesaurus

To illustrate our intuitions about adding a thesaurus, let's assume, we have positive (+) and negative (–) examples of some concept:

+ He signed the contract.

+ He signed the covenant.

– He signed the postcard.

– He signed the book.

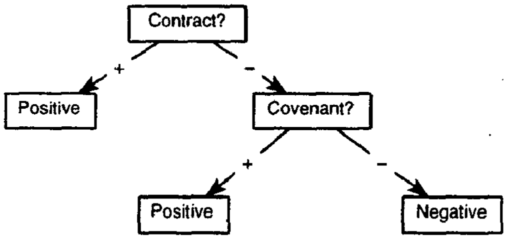


Figure 2: Decision tree learned without knowledge from a thesaurus

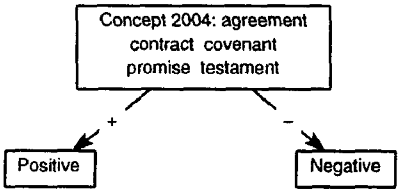


Figure 3: Decision tree learned when knowledge from a thesaurus is added to examples

Half of the examples is positive, half negative. No single term can discriminate between positive and negative examples. A decision tree algorithm would create a tree as in Figure 2, branching out too much. The knowledge to recognize that covenant and contract are synonyms is missing, and there is no reliable way to make that inference. With the help of a thesaurus, however, it is possible to induce a better tree.

There are two ways to include the information of a thesaurus. First, we can use a thesaurus to discover synonyms while inducing a decision tree, without the need to change the representation of the examples. Instead of learning a tree, where the nodes are words, we can learn a tree where the nodes are categories from the thesaurus. The relevant category in the WestLaw Thesaurus for this example is:

- agreement contract covenant promise testament

If we modify the learning algorithm accordingly, ID3 will choose this category to discriminate perfectly between positive and negative examples, shown in Figure 3. This tree will also correctly classify examples which use the term agreement instead of contract or covenant.

The main benefit of having a simpler decision tree can not be shown in a simple example. A well-known problem with ID3 is its tendency to overfit the data. In particular with many attributes available, the algorithm learns overly specific trees, which often misclassify new examples. Using the thesaurus in a way suggested in Figure 3 counteracts the overfitting, and thereby can lead to better performance.

A second way to include a thesaurus is to change the representation of the examples, and expand them in advance by adding all possible synonyms before the learning algorithm is applied. Our positive examples would then appear like:

+ He signed the contract + agreement covenant promise testament.

+ He signed the covenant + agreement contract promise testament.

Now, a tree can be easily learned for the example sentences, by choosing either the term covenant or contract to distinguish between positive and negative examples. This also results in a simpler tree, and thereby can help to avoid overfitting.

This second approach helps to generalize further than using the categories in the WestLaw Thesaurus. Consider cases where two sentences contain similar words, like "undisclosed" and "confidential", which are both indicative of factor F6, Security-Measures. Although they are not in the same synonym set, they would both be expanded to include the term "secret", and thus be recognized as synonyms by the algorithm.

On the other hand, this second approach has the distinct disadvantage that it does not deal with polysemy, multiple meanings and uses for one word. One will have to add the synonyms for all possible meanings. Overgeneralizations and wrong inferences may occur and affect performance. For sentence with the word artery, the following categories apply:

- artery freeway highway interstate parkway ...
- artery capillary vein venule vessel

The term artery, when used in a text about document about traffic would then be a positive example for both "highway" and "vein," which is undesirable.

5.4 Design for Integrating Linguistic Information

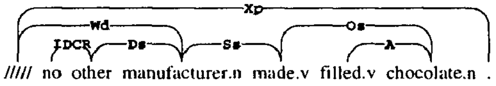
The most promising way to include linguistic information about the relation of words in sentences and to represent negation, is to integrate a parser into the learning system. While the version of SMILE evaluated here does not include linguistic information in the representation, this seems the best place to discuss the anticipated use of a parser in our program. Assume we have the examples:

+ No other manufacturer made filled chocolate.

– He was a manufacturer who made hunter stands.

The only reliable way to discriminate between the two examples is to include the fact that in the positive sentence "manufacturer" is modified by "no other" as an attribute. (The terms "filled chocolate" and "hunter stand" most likely occur in only one case, and therefore will be pruned away.) To avoid learning overly specific trees, they would therefore be deleted from the available attributes.

The attribute can be found by parsing the sentence, e.g. using CMU's Link Parser. (See `http://www.link.cs.cmu.edu.`) The output for the positive instance in the example above would be:



From this parser output, various information can be derived. The subject, object and verb of the sentence are identified, the words' part-of-speech is tagged, and, most interesting for the task at hand, the combination no-other is labeled as determiner of a noun (Ds), and as an idiomatic string stored in the dictionary (IDC).

Similarly, information about phrases, or the role of attributes in the sentence, can be derived and used in learning a decision tree. For instance, the phrase "filled chocolate" is indicated by an adjective link (A) between "filled" and "chocolate".

6 Experiment

To find out how well some of our ideas work, we conducted a preliminary experiment. In a simplified environment, we tested a prototype implementation of the ID3 decision tree induction algorithm. Our goal was to find out whether using sentences instead of full-text opinions would work, and whether there would be any benefit from adding a thesaurus. For the time being, we did not investigate in what ways the representation can improved, for example, by including linguistic knowledge. We are currently working on this as the next phase of our experimentation.

6.1 Experimental Setup and Assumptions

For this experiment, we use a subset of CATO's factors:

- F1, Disclosure-In-Negotiations
- F6, Security-Measures
- F15, Unique-Product
- F16, Info-Reverse-Engineerable
- F18, Identical-Products, and
- F21, Knew-Info-Confidential.

We have selected these factors because we anticipated that they would provide a range of difficulty for the learning algorithm. We expected F15, Unique-Product, to be found much more easily than F6, Security-Measures. As discussed above, courts employ a small set of patterns and some standard phraseology in discussing a product's uniqueness. By contrast, the squibs identify a very wide variety of different fact situations from which it may be inferred that F6 applies.

It is important to note that we have omitted some factors which we believe would be even harder to learn from examples than F6. F5, for instance, Agreement-not-specific, is even difficult for a human to discover, and asserting its presence requires more abstract inferences. Probably only very advanced natural language understanding would be appropriate. Also, the Case Database contains only five examples where this factor applies, so it is not a good candidate to show the applicability of a new method.

To simplify the problem further, we used CATO's squibs, rather than the full-text opinions as training examples. The squibs are short summaries, about one page of text. Their primary function is to restate the case facts succinctly. The drafters of the squibs had CATO's factors squarely in mind in preparing the summaries of the case facts. Thus, finding factors in the squibs is a significantly easier problem than finding factors in the full-text opinions. We adopted this simplification, however, to get a set of consistently marked-up examples, to avoid having the learning algorithm get bogged down computationally, and to satisfy our curiosity as to

	f15	f21	f1	f6	f16	f18
Precision	48.78%	32.14%	30.00%	80.55%	44.44%	58.97%
Recall	71.42%	50.00%	60.00%	81.69%	54.54%	63.88%

Table 1: Precision and recall for finding factors in cases

whether the method would work with the squibs before we undertook scaling up to the more complex opinions. We believe that the effects observed on squibs will also apply to opinions, but that experiment remains to be carried out.

6.2 Implementation

For the experiment, we implemented the basic ID3 algorithm (Mitchell 1997; Quinlan 1993) without any modifications, and without methods to prune the learned trees.

To generate the examples for the experiment, we split the squibs into sentences. Depending on their markup (see Section 5.1), they were labeled as positive or negative examples.

The cases are treated as binary attribute vectors, to simplify the representation. We only use the words that occur in the positive examples as attributes, which significantly decreases complexity. For this experiment, we also removed stopwords, and performed only very minor morphological corrections, like removing the plural-s suffix. In short, we have used a "bag-of-words" representation in this preliminary experiment to test the value of adding knowledge from a legal thesaurus. As described above, we plan to improve upon this representation in subsequent work.

The experiments were run as a 5-fold cross validation, as suggested, for instance, in (Quinlan 1993; Mitchell 1997). In five rounds, we left out one fifth of the examples as test data, and used the rest as training examples. This way, each sentence from the squibs was used exactly once in testing. In order to maintain a uniform class distribution, we performed the random partitioning of positive and negative examples seperately.

6.3 Results

We were interested in the effect of two techniques, namely using marked up sentences instead of the full documents to discover the factors, and adding a domain specific thesaurus. The results of the experiments suggest both are beneficial.

First, using a rule-learning algorithm for marked-up sentences as examples seems to be a good approach to tackle the problem. It reduces complexity, but still, the individual sentences contain enough information to be useful as examples for the factors. In our previous experiments where we attempted to learn factors from full-text opinions, the statistical learning methods could only learn the goal concepts to a very limited degree. Most of the time the classifiers could not discover the positive instances, which lead to low precision and recall. Here, however, the decision tree algorithm could achieve precision and recall of up to 80% for finding which factor applies to a case, as shown in Table 1.

Part of this is certainly due to the fact that we used only the case summaries, and not the much longer and more complex full-text opinions as our examples in these experiments. However, in most cases the sentences marked as positive instances for a factor in the squibs are very similar to the corresponding sentences in the full-text opinions, In fact, in some cases they were copied verbatim from the full-text opinions. In the much more compact squibs, we have fewer negative examples per document, but the complexity and length of the positive examples does not differ dramatically. This certainly increases the likelihood that using marked-up sentences will also be appropriate for the full opinions.

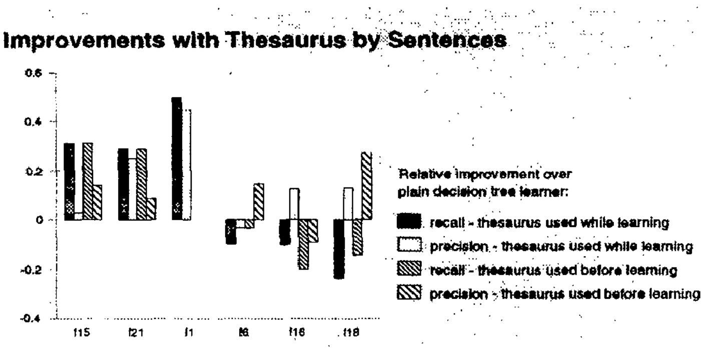


Figure 4: Relative performance improvement by adding a thesaurus

One may notice that, even though these results are positive, there is still room for improvement. This can be expected, since in the experiments, we did not include any linguistic information. In Sections 3.2 and 5.4, we discussed why in particular negation is needed for finding factors in legal texts, and how we are planning to integrate the necessary linguistic knowledge.

Even more interesting is the next result of our experiment. It shows that adding a thesaurus helps when classifying sentences under factors, but that the usefulness depends on the factor.

In Figure 4, we show the relative improvement of using the thesaurus both while (solid color) and before (striped) learning when compared to the plain decision tree algorithm. We calculated the difference in precision between the learner that used the thesaurus and the one that did not, and divided by the precision for the plain learning algorithm. The result is the relative change in precision, and allows us to compare the effects across factors. We did the same for recall, and for both ways of integrating the thesaurus.

The graph indicates that for factors F15, Unique-Product, and F21, Knew-Info-Confidential, adding the thesaurus clearly improves performance. This confirms our intuitions, which we illustrated in Section 4. For F1, Knew-Info-Confidential, adding the thesaurus while learning is useful, while adding it before learning is without effect. For F16, Info-Reverse-Engineerable, and F18, Identical-Products, adding the thesaurus increases precision, and decreases recall. It seems that the thesaurus makes the algorithm more "conservative" in discovering positive instances. The thesaurus is also not useful for factor F6, Security-Measures, which could be expected. In a commercial context, there is a wide variety of measures to keep information secret. They are often very practical matters not related to legal concepts. Therefore, a thesaurus of legal terms has little effect.

7 Conclusion

In this paper we presented our work toward automatically indexing legal documents. Our ultimate goal is to identify certain indexing concepts in case texts to help the construction and maintainance of case-based reasoning systems. Using a collection of indexed cases as examples, we apply machine learning techniques. However, methods that were successfully used in other domains are not directly applicable, because legal documents are exceptionally long and complex. Moreover, the representation normally used for simpler texts is not powerful enough to capture the language used in case opinions.

To overcome these problems, we suggested focusing on individual sentences related to the factors. This reduces the complexity of the examples, and enables us to use a better and more powerful representation. We illustrated our ideas on adding domain knowledge in the form of a legal thesaurus and linguistic information from parsing the sentences.

In a preliminary experiment, we tested whether using single sentences is appropriate and whether they contain enough information to be used as examples for a learning algorithm. We found that a simple decision tree induction algorithm could learn quite well to classify single sentences under the factors, and thereby find factors in case squibs. This result is not directly comparable to the experiments we have reported before. For practical reasons, we ran the experiments on short case summaries from CATO, and not the full-text opinions. In future work, we will test whether individual sentences also make better examples for full-text opinions.

The most interesting part of our experiments was whether adding a legal thesaurus would lead to the performance improvements over the plain algorithms. Although we could not observe better results for all factors, the results suppport the intuitions discussed in the paper. For factors, where the underlying fact situations are fairly concrete and uniform, the additional knowledge leads to better precision and recall. If a factor covers a wide variety of real-world situations, adding a thesaurus does not help.

Our next steps will be to integrate a parser, and add linguistic information to the examples. In this way, we intend to better capture the language used in legal texts. Overall, we think that using sentences as examples for a learning algorithm, adding domain knowledge in the form of a legal thesaurus, and adding linguistic information will help toward our goal of indexing legal texts automatically.

Acknowledgements

This research has been supported by the National Science Foundation, under Grant IRI96-19713. We would like to thank West Group and in particular Peter Jackson for making the WestLaw Thesaurus accessible to us.

References

Aleven, V., and Ashley, K. 1997. Evaluating a Learning Environment for Case-Based Argumentation Skills. In *Proceedings of the Sixth International Conference on Artificial Intelligence and Law (ICAIL-97)*.

Aleven, V. 1997. *Teaching Case-Based Argumentation through a Model and Examples*. Ph.D. Dissertation, University of Pittsburgh.

Branting, L. 1991. Building explanations from rules and structured cases. *International Journal on Man-Machine Studies* 34(6).

Brüninghaus, S., and Ashley, K. 1997. Using Machine Learning for Assigning Indices to Textual Cases. In *Proceedings of the Second International Conference on Case-Based Reasoning (ICCBR-97)*.

Burton, W. 1992. *Legal Thesaurus*. Simon & Schuster Macmillan.

Craven, M.; Freitag, D.; McCallum, A.; Mitchell, T.; Nigam, K.; and Slattery, S. 1998. Learning to Extract Symbolic Knowledge from the World Wide Web. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*.

Daniels, J., and Rissland, E. 1997. What you saw is what you want: Using cases to seed information retrieval. In *Proceedings of the Second International Conference on Case-Based Reasoning (ICCBR-97)*.

Greenleaf, G.; Mowbray, A.; King, G.; Cant, S.; and Chung, P. 1997. More Than *Wysh*ful Thinking: AustLII's Legal Inferencing via the World Wide Web. In *Proceedings of the Sixth International Conference on Artificial Intelligence and Law (ICAIL-97)*.

Lewis, D., and Riguette, M. 1994. A comparison of two learning algorithms for text categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR-94)*.

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

Moens, M.-F.; Uyttendale, C.; and Dumotier, J. 1997. Abstracting of Legal Cases: The SALOMON Experience. In *Proceedings of the Sixth International Conference on Artificial Intelligence and Law (ICAIL-97)*.

Moulinier, I.; Raskinis, G.; and Ganascia, J. 1996. Text categorization: A symbolic approach. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR-96)*.

Quinlan, R. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufman.

Rissland, E.; Skalak, D.; and Friedman, T. 1993. Case Retrieval Through Multiple Indexing and Heuristic Search. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*.

Sahami, M.; Craven, M.; Joachims, T.; and McCallum, A., eds. 1998. *Learning for Text Categorizations, Papers from the AAAI-98 Workshop*. AAAI Press.

Schweighofer, E.; Winiwarter, W.; and Merkl, D. 1995. Information Filtering: The Computation of Similarities in Large Corpora of Legal Texts. In *Proceedings of the Fifth International Conference on Artificial Intelligence and Law (ICAIL-95)*.

Smith, J.; Gelbart, D.; McCrimmon, K.; Athertin, B.; MacClean, J.; Shinehoft, M.; and Quintana, L. 1995. Artificial Intelligence and Legal Discourse: The Flexlaw Legal Text Management System. *Artificial Intelligence and Law* 2(1).

Statski, W. 1985. *West's Legal Thesaurus and Dictionary*. West Publishing.