Automatic Categorization of Case Law

Paul Thompson
University of St. Thomas
2115 Summit Avenue, OSS301
St. Paul, Minnesota 55105
1 651 962-5554
PTHOMPSON@stthomas.edu

ABSTRACT
This paper describes a series of automatic text categorization experiments with case law documents. Cases are categorized into 40 broad, high-level categories. These results are compared to an existing operational process using Boolean queries manually constructed by domain experts. In this categorization process recall is considered more important than precision. This paper investigates three algorithms that potentially could automate this categorization process: 1) a nearest neighbor-like algorithm, 2) C4.5rules, a machine learning decision tree algorithm; and 3) Ripper, a machine learning rule induction algorithm. The results obtained by Ripper surpass those of the operational process.

Categories and Subject Descriptors
H.3.2 [**Information Storage and Retrieval**]: Information Storage – *Record classification.*

General Terms
Experimentation

Keywords
Text Categorization.

1. INTRODUCTION
Automatic text categorization trained on previous manual categorization has been the topic of much recent research.

This research has followed two main approaches [22]. The first uses human knowledge engineering to build a rule-based system for categorization. Among the most accurate results are those obtained by CONSTRUE, a rule-based system, using manually encoded rules to categorize incoming newswire text into 674 economic and financial news categories [15]. While highly effective, the development of this system reportedly took approximately four person-years. The second approach, requiring much less development time, but with less accurate results, uses machine learning. One machine learning approach based on a k-nearest neighbor, kNN, algorithm is memory-based reasoning. Creecy et al. [11] describe a system that automatically categorizes U. S. Census Bureau text into 232 industry codes (71% accuracy) and 504 occupation codes (62% accuracy). Masand et al. [25] discuss another memory-based reasoning application where incoming newswire text is categorized into 361 categories. The memory-based reasoning technique considers a text to be categorized as a query to a large training database of manually categorized texts. The one or more highest ranked texts retrieved are the nearest neighbor(s). The query text is categorized using the manually assigned categorization of the nearest neighbor(s) according to a simple scoring algorithm. More recently Yang [39] obtained a break-even point, where precision equals recall, of .85 on the Reuters newswire collection. Other statistical and machine learning approaches to text categorization include: Maron [24], Crawford et al. [10], Yang and Chute [40] Lewis and Gale [21], Apte et al. [1], Lewis et al. [23], Cohen and Singer [7,8], Hodges et al. [16], Moulinier et al. [27], Yang [38], Leung and Kan [19], Koller and Sahami [18], and Yang and Pedersen [41].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ICAIL-2001, St. Louis, Missouri USA.
Copyright 2001 ACM 1-58113-368-5/01/0005 $5.00.

This paper describes text categorization experiments in the legal domain. WESTLAW [36] is an online legal retrieval system containing, among other types of documents, case law. Case law is the written opinions of judges in court cases. Courts send judges' opinions to West Group to be published in book, CD-ROM, and online products. On WESTLAW cases are categorized into 40 broad, high-level categories, e.g., *bankruptcy*. Currently a process called *topical view queries* assigns these incoming cases, or opinions, to the categories. Topical view queries are Boolean queries manually constructed by domain experts, i.e., editors, who are also attorneys: a labor-intensive, expensive process. Editors create queries, and then iteratively test and refine them by retrieval against WESTLAW, until acceptable performance is achieved. The queries must be manually revised periodically to maintain this performance. Recall is considered more important than precision, because end users will be more concerned about missing relevant documents, than present irrelevant documents. This paper investigates three algorithms that potentially could automate this categorization process: a k-nearest neighbor-like (kNN-like) algorithm using WIN (see below), and two machine learning rule induction algorithms, C4.5rules [29] and Ripper [7,8].

Text categorization research often deals with newswire [1, 15, 25] or other relatively short, non-technical [11] text. Case law, and legal text in general [14], differs significantly both in terms of length, average length of a case is about ten pages, and in many sublanguage features [17]. In particular, two features of case law

text tend to make automatic categorization more difficult: 1) a case's tendency to discuss many issues tangential to the main topic, and 2) the use of highly stylized legal language pervading all topics.

Categorization of legal text is also an area that has received attention [3, 4, 13, 28, 32, 33]. In particular this research, as well as the present paper, show the importance of feature selection for the categorization of legal text. As Brueninghaus and Ashley point out, however, systems have not yet successfully bridged the gap between the text of case law and symbolic AI representation [3].

Sections 2 and 3 of the paper describe the algorithms and experimental methodology. Section 4 presents the results, which are discussed in section 5. Initial experiments showed the kNN-like algorithm to be inadequate, but results with C4.5rules and Ripper were much stronger. In fact, Ripper's, results surpass those of the operational topical view queries.

2. Algorithms

2.1 The kNN-like Approach

The kNN-like approach treats a case as a WIN query and categorizes it by retrieving the most relevant document(s) from a database in which each document is a textual representation of a category. This approach is called kNN-like, because ordinarily with kNN approaches the object to be categorized would be compared to like objects, e.g., a legal case would be compared to a legal case, rather than to textual descriptions of categories. Human editors use brief one or two paragraph descriptions as definitions of the 40 categories. Each of these descriptions is considered as a document for these experiments, and the WIN queries are run against a text collection consisting of these 40 documents. WIN is West's natural language retrieval engine accessible through WESTLAW [35]. Because entire cases are too long to be used as queries, term, or feature, selection was required. Two methods were used, both based on a term's (word stem) inverse document frequency (idf) value, calculated as $\ln(N/n)/\ln(N)$, where N is the total number of documents in a collection and n is the number of documents containing the term. The idf values were taken from another large case law test collection, since the collections used in these experiments were too small to provide reliable values. If the idf value was less than 1.9, but greater than 1.0, the term was considered for inclusion in the query. The lower bound test removed very common terms from consideration, while the upper bound test removed terms occurring only once in the larger collection. From 10 up to 300 terms per query in increments of 10 were chosen using either median, or top (highest), idf values. The WIN ranking algorithm [2] permits several scoring methods based on: a) overall document score, b) best paragraph score, or c) a combination of a) and b). Best paragraph score is derived by treating each document paragraph as a document and giving the whole document the score of its best paragraph. The combination method gives equal weight to document and best paragraph scores. All of these methods were used.

2.2 C4.5rules

C4.5rules is a rule induction algorithm that takes as input an unpruned decision tree created by the C4.5 algorithm [29]. Through training data C45.rules learns rules to categorize new instances. Features are selected to represent each document. C4.5rules can perform multi-way categorization, but more typically in text categorization, as in this study, if there are n possible categories, the problem is broken up into n separate binary categorization problems. For example, is this document an *insurance* document, or a *non-insurance* document? C4.5 creates a decision tree by first identifying the single text feature, and a test associated with the feature, which is most informative in terms of categorizing all training documents into the target, or non-target, category. C4.5 uses the information gain ratio, a normalized version of the mutual information between category and feature, i.e., word stem. The first feature selected becomes the root of the decision tree. A branch is taken for each document being categorized according to whether or not the test is passed. In a similar way additional features and tests are selected. C4.5rules converts C4.5's unpruned decision tree to an equivalent set of rules and then prunes the rules to avoid overfitting the training examples. C4.5rules optimizes error rate, i.e., the proportion of mis-categorizations, which gives equal weight to recall and precision. If one or the other of recall or precision is considered more important, then optimizing error rate is not a good criterion. This weakness of C4.5rules has been eliminated in C5.0 the successor to C4.5 and C4.5rules [30]. C4.5rules can be run with a variety of tree and rule parameter settings which control the shape of the tree generated and the aggressiveness with which it is pruned. Various combinations of parameter settings were used, but results with the default setting were comparable to those obtained with any other setting, and are the only ones reported here.

2.3 Ripper

Unlike C4.5rules, Ripper categorizes text using a set of rules without a prior decision tree [7,8]. Ripper splits its training data into growing and pruning sets. Then, starting with an empty set of rules, it adds conditions to the antecedents of rules until no negative examples are covered in the growing set and then deletes conditions to improve performance on the pruning set. The RIPPER algorithm has two main stages. First, a greedy process constructs an initial rule set. The second stage optimizes the rule set, i.e., increases its compactness and accuracy, using incremental reduced error pruning [6]. In this stage, for each of the original rules in turn, two alternative rules are considered: a replacement rule and a revision rule. The replacement rule is grown and pruned, starting from the empty rule, so as to minimize error of the entire rule set. The revision rule starts with the original rule, rather than the empty rule, but is treated in a similar way.

Ripper has two extensions [7,8] specifically intended for text categorization. First, instead of representing documents as feature vectors, as does C4.5rules, documents can be represented by a single set-valued feature, i.e., the set of word stems contained in the document. The set-valued representation is much more efficient than the vector representation used by C4.5rules, but only allows Boolean features. The second extension is the use of a loss ratio, i.e. the ratio of the cost of false negatives to the cost of false positives, which allows greater weight to be given recall or precision. For example, if it is twice as costly for the algorithm to categorize a true *insurance* document as a *non-insurance* document, as it is to categorize a true *non-insurance* document as an *insurance* document, then recall will tend to be higher and precision lower. In these experiments the loss ratio was used, but the set-valued feature

representation was not, since experiments with C4.5rules indicated that frequency-based features outperformed Boolean features.

3. Experimental Methodology

3.1 Training and Test Collections

Three test collections were used with these experiments. The first, used with the kNN-like approach, consisted of 125 cases, each with assigned categories. The second, also used with the kNN-like approach, was a 37 case subset of the first test collection. The third, used with the other algorithms, consisted of the merger of the first test collection with 7,535 additional cases. This much larger collection was needed to provide a large training collection for the C4.5 and Ripper machine learning algorithms. All editorial enhancements, e.g., headnotes, were removed from the test cases. Headnotes are editorially created summaries of each point of law discussed in a case. Editor/attorneys assigned multiple categories, typically 2 to 4, to each case as part of production WESTLAW processing. Editorial enhancements were removed, because the goal of the research was to determine how well cases could be classified before receiving any editorial enhancements. For the first two test collections categories were also ranked. The categorization scheme consisted of 40 broad legal categories, or topics, e.g., *bankruptcy*, or *criminal justice*.

The kNN-like experiments did not use a training collection. Categories were represented by either: topical definitions or headnotes. Although headnotes were excluded from test cases, as discussed above, they were used in some of the kNN-like experiments as an alternative representation of the categories. The topical definitions, typically about half a page long, as well as headnotes, are written by West editors. Headnotes are classified according to the West Key Number Classification System [36]. Each headnote focuses on a single point of law contained in a case – a much narrower focus than that of any of the 40 broad topics. This focus mismatch was overcome by assigning headnotes from multiple Key Numbers, which among them represented the coverage of the topical area. Since it was an expensive process, West editors supplied a list of from 13 to 21 Key Numbers for 5 of the 40 topics. Headnotes assigned to each Key Number were randomly selected to provide the text used to represent each of the 5 topics. Only 37 of the 125 test cases had categories corresponding to these five topics, so only these test cases were used in the second test set. It was hypothesized that the larger quantity of headnote, as compared to the definitional, text for a topic, as well as the nature of the text, i.e., that fact that the text was more like that of the case to be categorized, would improve performance.

Experiments were run with either 10 or 100 headnotes per Key Number test collections. Since few of the 37 test cases had more than one category among the 5 topics, experiments were only done where all relevant categories were considered. Experiments were also run in which the 5 topics were represented by both headnote and definitional text. A document was the entire set of headnotes corresponding to a given topic. Paragraphs were defined to be either each individual headnote, or all headnotes with the same Key Number.

For the experiments with C4.5rules, and Ripper the training collection consisted of 7,535 unique cases with categories manually assigned by editor/attorneys. Some experiments were run with the same 125 document test collection used for queries with the kNN-like algorithm, but in most experiments the 7,535 and 125 document collections were merged and 10-way crossvalidation done. The merged collection was divided into 10 blocks with similar distributions of categories and 10 runs made, each time using a different block as the test set and training on the other 9. The final result is the average of the 10 runs. The C4.5 utility for crossvalidation was also used with Ripper, so that results could be compared directly. For the operational *topical view query* process, no training collection was required. The Boolean queries were run against the merged collection used by C4.5rules and Ripper. Because the topical view queries used field restrictions for fields not included in the C4.5rules cases, two variants of the queries were used. The first variation, unaltered, used the queries exactly as used in production. The other variation, altered, restricted the Boolean query to the opinion field, except for the initial part present in most queries, which used an early field, typically the title field, to anchor the search to the beginning of the case. The second variation allowed for a more direct comparison with the machine learning algorithms, since no fields reflecting editorial enhancement were used with these algorithms.

3.2 Feature Selection: C4.5rules and Ripper

In the text categorization literature algorithms are described which: a) select a single set of features for all categories, e.g., [22]; or b) select a separate set for each category, e.g., [1, 26]. For both C4.5rules and Ripper features were selected from a set of manually assigned Keywords taken from the West Legal Directory [37]. These Keywords were associated with roughly the same 40 categories of this study. From the 900 unique Keyword stems the 300 with the top, or median (see section 3.1) idf scores were retained. For each category a set of features was derived by merging these 300 stems with the stems associated specifically with that category. For example for a given category there might be 25 stems. Merging these 25 stems with the 300 might result in a feature set of 314 unique stems. Various methods of calculating feature values were explored, including binary, e.g., the feature is present, or absent, and simple frequency, but ultimately the best results were achieved with expected frequency, calculated as:

attribute value = (frequency of stemmed term in document) / (expected frequency of stemmed term)

expected frequency of stemmed term = ((collection frequency of stemmed term) * (document length)) / (collection length of stemmed term)

collection frequency = number of occurrences of stemmed term in collection

document length = number of non-stop word, term occurrences in document

collection length = sum of document lengths of all documents in collection having at least one occurrence of stemmed term.

3.3 Evaluation

Four measures were used to evaluate the kNN-like algorithm: search length, ranking precision, recall and precision, and break-even point. Search length is a ranked retrieval measure of how many irrelevant documents must be examined in a ranking before a pre-specified number of relevant documents are found [9]. Ranking precision for a given rank is the proportion of relevant topics among all topics retrieved up to that rank. Standard recall and precision measures were also calculated. These are summarized in tables 1a and 1b as the 11 point average, i.e., the average precision interpolated at 10% levels of recall from 0 to 100 percent. Finally, in order to compare the results of this study with other research, the break-even point, i.e., the highest value where recall equals precision [22] was calculated.

In the experiments with C4.5rules, Ripper and the topical view query process, all categorization problems were binary. Thus a document *A*, which had both the categories, *bankruptcy* and *finance and banking*, would be considered a bankruptcy document when the categorization problem was *bankruptcy* or *non-bankruptcy*, and a *finance and banking* document, when the problem was *finance and banking* or *non-finance and banking*. Unlike with the 125-document test collection, there was no ranking of multiple categories with respect to a document. Accordingly recall, precision, and f-measure [34] scores were used to evaluate these runs, without the additional measures used with the kNN-like approach.

4. Results

4.1 kNN-like Approach

Tables 1a and 1b summarize the best results for the kNN-like runs. "SL" indicates search length. In run names "allrels" indicates that all relevant topics were used, while “onerel” indicates that only the single most relevant topic was considered relevant and "top" indicates the top idf feature selection method, where stems having the highest idf values were selected. Percentage in top *n* ranks is the frequency with which at least one relevant case is correctly categorized by rank *n*. An asterisk indicates an approximate break-even point. Where only one category was considered relevant, search length measures how many irrelevant topics, on average, were ranked ahead of the relevant topic Since the headnote collections had only 5 topics, each run had 100 percent top 5 performance.

Table 1a Selected 40 Topic KNN-LIKE results.

Run (No. of terms)	SL	11 pt Average	Break-even Point	Percent in Top Rank	Top 2	Top 3	Top 4	Top 5
Definitional-onerel (250)	2.6	.581	.581	40.0	59.2	72	77.6	82.4
Definitional-allrels (300)	0.4	.635	.57*	69.6	88.8	97.6	99.2	100
Definitional-top-allrels (130)	0.4	.661	.605*	75.2	89.6	97.6	98.4	98.4

Table 1b Selected 5 Topic KNN-LIKE results

Run (No. of terms)	SL	11 pt Average	Break-even Point	Percent in Top Rank	Top 2	Top 3	Top 4	Top 5
Definitional-top (20)	0.3	.904	.894	86.5	91.9	91.9	100	100
Headnotes10 (80)	0.8	.776	.764	67.6	78.4	89.2	94.6	100
Headnotes100 (290)	0.6	.809	.809	67.6	83.8	91.9	97.3	100
Headnotes10/ Definitional-top (120)	0.2	.922	.917	86.5	91.9	97.3	100	100

Based on the 11 point average the best definitional text result on all 40 topics was .661, achieved by Definitional-top-allrels with 130 terms. Contrary to expectation, using headnote text did not lead to improvement over results with definitional text. Using 100 headnotes per Key Number resulted in somewhat better performance than the use of 10 headnotes, but still far below that obtained using definitional text. Combined definitional and headnote text gave slightly better performance than use of definitional text alone.

4.2 C4.5rules, Ripper, & Topical View Query

Table 2 shows average categorization results for 18 topics. The baseline is C4.5rules, release 8, using simple term frequency feature values. The other methods are: C4.5rules and Ripper using expected frequency normalization for feature values; and the operational topical view queries, both unaltered and altered, as discussed above. Ripper was used with its loss function set to give recall four times the value of precision, L0.25, or without a loss function, L1. All C4.5rules and Ripper runs used the WLD

(West Legal Directory Keywords) features. Results are presented in terms of recall, precision, error, and f-measure with $\beta = 2$, a single measure combining recall and precision which, with $\beta = 2$, gives twice as much weight to recall as precision. The error is the proportion of mis-classified cases, i.e., each false positive and false negative is counted. This is the parameter optimized by the C4.5rules algorithm.

As shown in table 2, the topical view query process, whether altered or unaltered, did better than either C4.5rules method for recall. For the f-measure with $\beta = 2$, however, the differences are less, especially when comparing the expected frequency version of C4.5rules to the altered topical view query process. Ripper, on the other hand, outperforms the topical view process, whether using altered, or unaltered queries. Ripper performs slightly better than C4.5rules, even without the loss function. Ripper, using L0.25, achieves an 11.77% improvement over the unaltered topical view query process in terms of recall, or a 10.24% improvement for the f-measure with $\beta = 2$.

Table 2. C4.5rules, Ripper, and Topical View Queries Average Performance for 18 Topics

Categorization Method	Recall	Percent Change	Precision	Percent Change	Error	Percent Change	F-measure $\beta = 2$	Percent Change
C4.5r r8 (baseline)	.4739		.6734		3.80		0.4954	
C4.5r r8 exp. frequency	.5139	8.44	.6798	0.95	3.68	(3.16)	0.5348	7.95
Ripper L1	.5283	11.48	.6926	2.85	3.54	(6.84)	0.5446	9.93
Ripper L0.25	.6732	42.06	.5188	(22.96)	4.50	18.42	0.6376	28.70
TVQ: Unaltered	.6023	27.09	.5468	(18.80)	4.79	26.05	0.5784	16.75
TVQ: Altered	.5705	20.38	.5500	(18.32)	4.78	25.79	0.5540	11.83

Table 3. Ripper and Unaltered TVQ Performance for 18 Topics

	Ripper			TVQ		
Category	Recall	Precision	F: 2	Recall	Precision	F:2
Bankruptcy	.8193	.6888	.7884	.6331	.8980	.7426
Business Organizations	.5254	.4295	.4996	.3430	.5086	.4097
Commercial Law	.4130	.3790	.4044	.3007	.3254	.3163
Criminal Law	.8538	.6974	.8165	.8493	.5324	.6545
Energy	.6338	.5237	.6036	.5436	.4094	.4671
Estate Planning	.7424	.6597	.7219	.6716	.4623	.5477
Finance & Banking	.6519	.4769	.6049	.5366	.5308	.5337
Government Benefits	.2586	.3652	.2531	.3841	.2637	.3127
Insurance	.8058	.6052	.7541	.8124	.5130	.6289
International Law	.6054	.5197	.5803	.4797	.7172	.5749
Military Law	.7889	.5154	.7039	.9000	.4417	.5926
Native Americans	.8705	.6981	.8275	.6897	.7407	.7143
Products Liability	.5960	.5991	.5941	.4205	.7651	.5427
Professional Malpractice	.8178	.6296	.7694	.7193	.6447	.6799
Real Property	.6349	.4814	.5948	.5682	.5505	.5592
Taxation	.9092	.7100	.8596	.8215	.7305	.7958
Tort Law	.6380	.4126	.5741	.6171	.3834	.4730
Transportation	.5526	.4610	.5275	.5512	.4248	.4798
Average	.6732	.5188	0.6376	.6023	.5468	0.557

Table 3 shows the performance of Ripper with L = 0.25 and that of the unaltered topical view query process for each of the 18 topics. The better performance of Ripper, as compared to the topical view queries is statistically significant at the .001 level (a normal deviate of 3.54) for the f-measure and at the .01 level (a normal deviate of 2.59) for recall, using the sign test.

5. Discussion

The results of the kNN-like approach experiments using the 5 and 40 category collections are not directly comparable. This makes it more difficult to assess the relative merits of using definitional, as opposed to headnote text, for topic representation,. The bridge between the two sets of experiments is the comparison of the performance of the 5 topic headnote collections to that of the definitional text for the same 5 topics. This comparison shows that definitional text outperforms headnote text, though combined headnote and definitional text gave the best results for the 5 topic collections.

An interesting aspect of the improved performance of top (highest), as compared to median, idf term selection was that far fewer terms were required to achieve the best results. For example *definitional-top-allrels* performed best with 130 terms, while the corresponding median idf run, *definitional-allrels*, required 300 terms for its best performance. This suggests that improved term selection, might yield better results. In particular selecting terms from an entire case may be suboptimal, since cases tend to include much discussion that is removed from the main focus. If the most representative paragraph(s) of that main focus could be automatically determined and used as the source of candidate terms, much better categorization may be possible. Headnote performance might also improve with these queries. Since headnotes are very specific, the lack of focus resulting from use of an entire text, may have had a more serious negative effect on headnote as compared to definitional representation. It is also possible that improvement might come from using longer definitional text.

The kNN-like results are not as strong as those obtained with C4.5rules or Ripper. In a comparison on the 125-document test collection the kNN-like algorithm achieved an average recall of .2606 with precision of .8369, as compared to C4.5rules results of .4112 for recall and .7957 for precision. Given that recall is considered more important than precision for this application, C4.5rules clearly outperforms the kNN-like algorithm. In a recent article Quinlan [30] discusses an improvement to the C4.5rules algorithm in the handling of continuous attributes (release 8). In the experiments reported here release 8 gave slightly better performance than release 5. In another study in the legal domain C4.5rules and Ripper were used to categorize statutes text [12]. It is hypothesized that machine learning techniques might be more successful with statutory text than with case law, because: 1) statutes tend to focus on a single topic, and 2) use more consistent language in that statutes are legislatively drafted, while case law reflects the writing styles of many individual judges.

In these experiments an attempt has been made to learn effective statistical categorizers, using features which had been manually selected by editors for other purposes. In contrast much of the work in AI and Law has focused on the construction of deeper knowledge representation. The work of Rissland and Daniels [13, 33] suggests that combining case-based reasoning techniques with passage retrieval techniques from information retrieval can lead to better feature selection. Other recent research in AI and Law also shows the benefits to be derived from improved feature selection [3, 4, 28, 33].

6. Conclusion

This study shows that Ripper outperforms both the *topical view query* process currently in production use and the C4.5rules algorithm. The kNN-like algorithm, on the other hand, does not perform as well as the other methods. Ripper's recall is substantially better than that of either C4.5rules or the operational process. Furthermore, it is reasonable to conjecture that enhancements to the Ripper approach could significantly improve its performance, while the manual *topical view query* process, at least without considerable expensive manual effort, may have already neared its peak level of performance, given the effort put into moving it into production. Although this study focuses on categorization accuracy, rather than on efficiency, efficiency is a concern. Besides introducing a cost function, comparable to Ripper's loss function, C4.5rules's successor, C5.0 [31], uses a sparse data representation, which allegedly reduces storage requirements and increases processing time by an order of magnitude. Further experiments with C5.0 and with Ripper's set-valued feature representation may be run to determine whether processing can be made more efficient without sacrificing accuracy.

7. ACKNOWLEDGMENTS

The author wishes to thank Jim Flood and Jack Conrad for their help with the software environment for these experiments.

8. REFERENCES

[1] Apté, C.; Damerau, F.; Weiss, S. M. Towards language independent automated learning of text categorization models. SIGIR '94 Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval Dublin, Ireland, 23-30.

[2] Broglio, J.; Callan, J. P.; Croft, W. B.. INQUERY system overview In Proceedings of the TIPSTER TEXT PROGRAM (Phase 1) San Francisco: Morgan Kaufmann, 47-67, 1994.

[3] Brueninghaus, S. and Ashley, K. D. Toward adding knowledge to learning algorithms for indexing legal cases. In International Conference on Artificial Intelligence and Law (ICAIL-99) (Oslo, Norway), 9-17.

[4] Brueninghaus, S. and Ashley, K. .D. Finding factors: Learning to classify case opinions under abstract fact categories. In International Conference on Artificial Intelligence and Law (ICAIL-97) (Melbourne, Australia), 123-130.

[5] Cohen, W. W. Efficient pruning methods for separate-and-conquer rule learning systems. In Proceedings of the 13th International Joint Conference on Artificial Intelligence (Chambery, France), 1993.

[6] Cohen, W. W. Fast effective rule induction Machine Learning: Proceedings of the Twelfth International Conference San Francisco: Morgan Kaufmann, 1995.

[7] Cohen, W. & Singer, Y. Context-sensitive learning methods for text classification. In Frei, H.-P., Harman, D., Schauble, P., & Wilkinson, R. (Eds.) Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '96),. Konstanz, Germany: Hartung-Gorre Verlag, 307-315.

[8] Cohen, W. & Singer, Y. Context-sensitive learning methods for text classification. ACM Transactions on Information Systems 17, 2, 141-173, 1999.

[9] Cooper, W. S.. Expected search length: a single measure of retrieval effectiveness based on the weak ordering action of retrieval systems. American Documentation 19, 1, 30-41, 1968.

[10] Crawford, S. L.; Fung, R.; Appelbaum, L. A.; Tong, R. M. Classification trees for information retrieval In Birnbaum, L. A. and Collins, G. C. (eds.) Machine Learning Proceedings of the Eighth International Workshop (ML91) San Mateo, CA: Morgan Kaufmann, 245-249.

[11] Creecy, R. H.; Masand, B. M.; Smith, S. J. Waltz, D. L. Trading MIPS and memory for knowledge engineering Communications of the ACM 35, 8, 48-64, 1992.

[12] Curran, T. and Thompson, P. Automatic Categorization of Statute Documents Proceedings of the 8th ASIS SIG/CR Classification Research Workshop (Washington, D.C1997), 19-30.

[13] Daniels, J. J. and Rissland, E. L. Finding legally relevant passages in case opinions. In International Conference on Artificial Intelligence and Law (ICAIL-97) (Melbourne, Australia).

[14] Danet, B. Language in the legal process. Law & Society Review 14, 3, 445-564, 1980.

[15] Hayes, P. J. Intelligent high-volume text processing using shallow, domain-specific techniques In Jacobs, P. (ed.) Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval Hillsdale, N.J.: Lawrence Erlbaum, 227-241, 1992.

[16] Hodges, J., Yie, S., Reighart, R., & Boggess, L. An automated system that assists in the generation of document indexes. Natural Language Engineering, 2, 137-160, 1996.

[17] Kittredge, R. and Lehrberger, J. (eds.). Sublanguage: studies of language in a restricted domain. de Gruyter, New York, 1982.

[18] Koller, D. & Sahami, M. Hierarchically classifying documents using very few words. In Fisher, D. H. (ed.) Machine Learning: Proceedings of the Fourteenth International Conference (ICML '97) San Francisco: Morgan Kaufmann, 170-178.

[19] Leung, C.-H. & Kan, W.-K. A statistical learning approach to automatic indexing of controlled index terms. Journal of the American Society for Information Science, 48, 55-66, 1997.

[20] Lewis, D. Representation and learning in information retrieval. Ph.D. Thesis, Computer Science Department, University of Massachusetts, Amherst, Technical Report 91-93, 1992.

[21] Lewis, D. & Gale, W. A Sequential algorithm for training text classifiers. In Croft, W. B. & van Rijsbergen, C.J. (Eds.). Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR '94) Berlin: Springer-Verlag, 3-12.

[22] Lewis, D. D. and Ringuette, M. A comparison of two learning algorithms for text categorization. Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval Las Vegas, Nevada: Information Science Research Institute, 81-93, 1994.

[23] Lewis, D., Shapire, R., Callan, J., & Papka, R. Training algorithms for linear text classifiers. In Frei, H.-P., Harman, D., Schauble, P., & Wilkinson, R. (Eds.) Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '96) Konstanz, Germany: Hartung-Gorre Verlag, 307-315

[24] Maron, M. E. Automatic indexing: an experimental inquiry Journal of the Association for Computing Machinery. 8, 3, 404-417, 1961.

[25] Masand, B.; Linoff, G.; Waltz, D. Classifying news stories using memory based reasoning. SIGIR '92 Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval Copenhagen, Denmark, 59-65.

[26] Moulinier, I. Feature selection: a Useful preprocessing step. 19th Annual BCS-IRSG Colloquium on IR Research, 1-11, 1997.

[27] Moulinier, I., Raskinis, G., Ganascia, J.-G. Text categorization: a symbolic approach. In Information Science Research Institute, University of Nevada, Las

Vegas (Ed.). Proceedings Fifth Annual Symposium on Document Analysis and Information Retrieval (). Las Vegas: University of Nevada, Las Vegas, 87-99, 1996.

[28] Pannu, A. S. Using genetic algorithms to inductively reason with cases in the legal domain. In Proceedings of the 5th International Conference on AI and Law, 175-184, 1995.

[29] Quinlan, R. C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo, CA., 1993.

[30] Quinlan, J. R. Improved Use of Continuous Attributes in C4.5 Journal of Artificial Intelligence Research. 4, 77-90, 1996.

[31] Quinlan, R. http://www.rulequest.com, 1997.

[32] Rissland, E. L. and Daniels, J. J. Using CBR to drive IR In International Joint Conference on Artificial Intelligence (IJCAI-95) (Montreal, Canada), 400-407.

[33] Schweighofer, E. and Winiwarter, W. Intelligent information retrieval: KONTERM – Automatic representation of context related terms within a knowledge base for a legal expert system. Proceedings of the 25th Anniversary Conference of the Istituto per la documentazione giuridica of the CNR: Towards a Global Expert System in Law, (Padua, Italy), 1994

[34] van Rijsbergen, C.J. Information Retrieval 2nd ed. Butterworths, London, 1979.

[35] West Publishing Company. WESTLAW Reference Manual, 5th ed. West Publishing Company, St. Paul MN, 1993.

[36] West Publishing Company. West's Analysis of American Law. West Publishing Company, St. Paul MN, 1994.

[37] West Publishing Company. West's Law Finder: A Legal Resources Guide. West Publishing Company, St. Paul, MN, 1995.

[38] Yang, Y. Sampling strategies and learning efficiencies in text categorization. In AAAI Spring Symposium on Machine Learning in Information Access, 88-95, 1996.

[39] Yang, Y. An evaluation of statistical approaches to text categorization. Carnegie Mellon University School of Computer Science technical report CMU-CS-97-127, 1997.

[40] Yang, Y. and Chute, C. G. An application of Least Squares Fit Mapping to text information retrieval In SIGIR '93 Proceedings of the Sixteenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval Pittsburgh 281-290, 1993.

[41] Yang, Y. and Pedersen, J. A comparative study of feature selection in text categorization. In Fisher, D. H. (ed.) Machine Learning: Proceedings of the Fourteenth International Conference (ICML '97) San Francisco: Morgan Kaufmann, 412-420.