Do Thesauri Enhance Rule-Based Categorization for OCR Text?

Kazem Taghva*, Jeffrey Coombs
Information Science Research Institute, University of Nevada, Las Vegas,
Las Vegas, NV 89154-4021

ABSTRACT

A rule-based automatic text categorizer was tested to see if two types of thesaurus expansion, called query expansion and Junker expansion respectively, would improve categorization. Thesauri used were domain-specific to an OCR test collection focussed on a single topic. Results show that neither type of expansion significantly improved categorization.

Keywords: text categorization, thesaurus, optical character recognition, OCR, rule based, IREP, RIPPER

1. INTRODUCTION

The explosive growth in the number of texts has led to the creation of many programs to automatically classify documents.[1] There are two main approaches to such automatic text categorization: the *statistical* and the *symbolic*. Statistical categorizers apply concepts such as probability, regression, and support vector machines to the problem.[1–3] Symbolic categorizers, with their roots in Artificial Intelligence, hope to mimic the ways of human intelligence to solve the problem. There are two main types of symbolic categorizers, decision-tree and rule-based. The latter is the subject of this paper.

Several types of rule-based categorizers have been developed and evaluated.[4–8] In particular Wenzel and Hoch have applied the rule-based categorizer INFOCLAS to OCR text,[9] and Junker has studied the effectiveness of using thesaurus information to aid a rule-based system.[2] No one, so far as we know, has yet taken up our question: will a domain-specific thesaurus improve a rule-based categorizer on OCR text?

We consider two approaches to using a thesaurus to augment a rule-based categorizer. One uses the tactic of *query expansion* from information retrieval to expand texts with thesaurus terms.[10, 11] The second, following Junker,[2] attempts to use thesaurus terms as replacements during the rule-building process. The hope was that by introducing terms from a thesaurus developed specifically for the document collection to be categorized, rule learning might be aided by tapping that "knowledge base". Unfortunately, neither approach, we discovered, improved categorization significantly. We conclude that the use of a domain-specific thesaurus will *not* significantly improve the performance of a rule-based text categorizer on OCR text.

2. RULE-BASED AUTOMATIC TEXT CATEGORIZATION

The aim of automatic text categorization is to approximate the total function $f : D \times C \rightarrow \{0, 1\}$, where D is a set of documents $\{d_1 \ldots d_n\}$ and C is a set of predetermined categories $\{c_1 \ldots c_m\}$.[1] Thus, a categorizer tries to emulate the unknown "perfect" mapping of each document in a collection and each predetermined category to the correct decision that a given document either belongs in a given category or not.

Rather than applying a series of complex mathematical formulas to emulate this mapping, a rule-based categorizer will generate a series of conditional or "if-then" rules which stipulate when a document should be placed in a given category. The rule-based automatic text categorizer developed for this project consists of two parts. The first is the well-known expert system shell called CLIPS.[12, 13] The second is a C++ program which produces a complete CLIPS program which will attempt to categorize documents. This latter program is called a *Clips-Knowledge Acquisition eNgine for Text categorization* or C-KANT.

* taghva@isri.unlv.edu.

Proceedings of Electronic Imaging Science and Technology: Document Recognition and Retrieval X, SPIE Vol. 5010, Jan. 22-24 2003, Santa Clara, CA., pp. 111-119

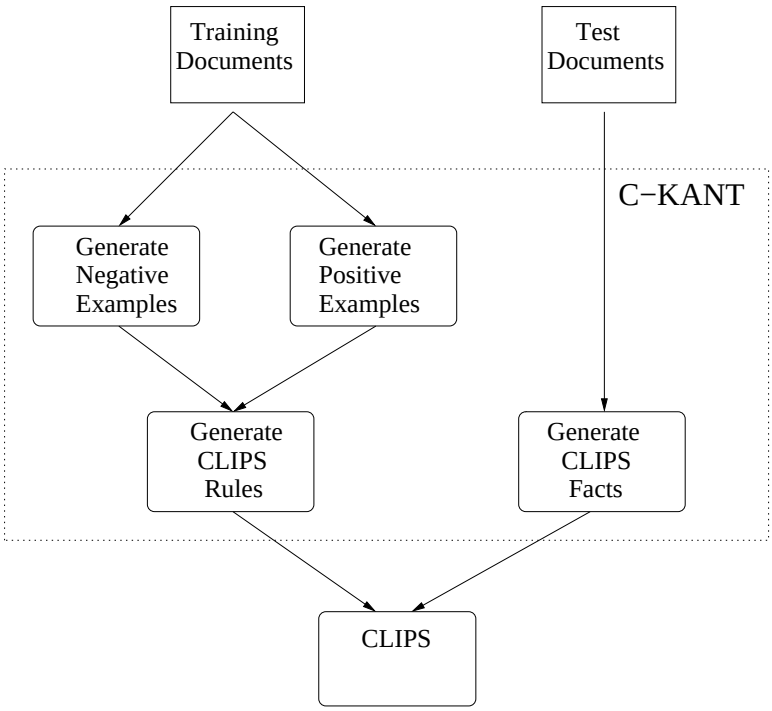

Figure 1. C-KANT's Structure

An expert system has three basic components: (1) a set of facts, (2) a collection of rules, and (3) an inference engine which can draw conclusions from the first two. An expert system shell only provides the inference engine. Facts and rules must be provided from outside, and these could be created by human experts as in traditional expert systems. However, one of the primary difficulties in creating an expert system is determining and expressing the expertise of the human expert in a form which the shell can use. It is extremely expensive to interview experts, and then to program and test the resulting system.

A program which can generate rules automatically is called a *knowledge acquisition* system or engine.[12] In the case of text categorization, the usual approach is to have human experts first categorize a relatively small subset of the documents in question. This subset of documents constitute the *training* set. Then the system will use one of a number of machine learning techniques to generate rules for predicting in which category new documents should be placed. Figure 1 shows how the knowledge acquisition engine C-KANT provides rules and facts for the CLIPS expert system.

3. RULE LEARNING IN C-KANT

The main machine learning technique used in C-KANT is Fürnkranz' *Incremental Reduced Error Pruning* (IREP)[14] optimized using Cohen's RIPPER algorithm.[5] Both are briefly outlined in this section.

IREP proceeds in two stages. First, it attempts to *grow* a rule, and then it tries to *prune* that rule. IREP begins growing rules by constructing a rule with an empty antecedent such as

$$\text{---} \in d_j \rightarrow d_j \in c_i.$$

This rule states that if any term is in a document d_j, then that document will belong to category c_i. Each unique term from a given set of documents, listed in a dictionary for this purpose, is then added to the antecedent in a "greedy" fashion. For example, suppose the dictionary contains terms $t_1, t_2, \ldots, t_n$. Then for each term t_k, IREP constructs a rule

$$t_k \in d_j \rightarrow d_j \in c_i.$$

Each rule so constructed will cover a certain number of positive and negative examples. A rule *covers* an example iff the example satisfies the antecedent of the rule. Thus, if the rule

$$t_1 \ \in \ d_j \rightarrow d_j \ \in \ c_i$$

is under consideration, it covers all the documents, positive and negative, which contain the term t_1. A rule with an empty antecedent is satisfied by any example.

To determine which candidate rule is "best," each new rule r_{i+1} is compared to the rule determined at the previous stage, r_i, using the *information gain* formula[15]:

$$Gain(r_i, r_{i+1}) \equiv T_{i+1}^{+} \times \left(-\log_2 \frac{T_i^{+}}{T_i^{+} + T_i^{-}} + \log_2 \frac{T_{i+1}^{+}}{T_{i+1}^{+} + T_{i+1}^{-}} \right). \tag{1}$$

T_m^{+} is the number of positive examples which are covered by rule r_m and T_m^{-} is the number of negative examples covered.

When the term t_k with the highest gain is found, the algorithm will attempt to enhance the rule containing t_k alone by considering rules which use t_k conjoined with other terms in the dictionary. So for each $l \neq k$, IREP evaluates the gain of

$$t_k \ \in \ d_j \wedge t_l \ \in \ d_j \rightarrow d_j \ \in \ c_i$$

In this case, both t_k and t_l must appear together in the same document for that document to count as a covered example. IREP will continue to add atoms of the form $t_\alpha \ \in \ d_j$ to the antecedent of the rule until there is no more information gain from doing so or other stopping conditions are met, as we will see below.

After a rule with the maximum information gain is grown, IREP tests to see if a shorter and more efficient version of the rule is as good as the original, longer version. To do so, it attempts to *prune* the rule. Pruning is necessary to avoid the problem of *overfitting* the training data.[5, 14] Overfitting occurs when rules train so closely to the training data that they do not generalize well to new examples. Pruning is meant to reduce the growth of rules so that their shorter versions might better cover unseen examples.

In the pruning stage, the data set for a category, consisting of positive and negative examples, is divided into two sets: a data set for growing and a data set for pruning. Usually the ratio is two-thirds of the data for a category is used for growing and one-third for pruning. This was the ratio used for C-KANT's tests.

IREP will take the rule learned in the growing stage and successively remove atoms one by one to determine if the removals degrade the rule. The measure of the "goodness" of a rule r_i in this case is determined by the formula:

$$f(r_i) = \frac{U_i^{+} - U_i^{-}}{U_i^{+} + U_i^{-}} \tag{2}$$

where U_i^{+} is the number of positive examples in the pruning set covered by pruned rule r_i and U_i^{-} is the number of negative examples covered. The pruned rule with the maximum value for $f(r_i)$ is kept and that rule is added to the rule set for the category.

C-KANT allows the option of optimizing the basic IREP algorithm by using modifications from Cohen's RIPPER.[5, 16] Experiments showed that these modifications tended to improve performance and these were used for the experiments reported here.

There are two main changes to the IREP algorithm in RIPPER. The first concerns the stopping criterion for learning a rule set. The second modification is the addition of multiple optimizing runs which attempt to replace rules in a learned rule set with more effective rules. Both modifications make extensive use of the *Minimum Description Length (MDL)* measure.

The MDL measure is based on the communication model of information typical of information theory. In general it is a measure of the length of a theory T and the data D on which T is based. The description length of T is the cost of a message encoding T, the *theory cost*, and the cost of encoding D given T is true.

The first cost represents the complexity of the theory and the second the extent to which the theory fails to account for the data.[17]

The philosophical justification for MDL comes from Ockham's razor, which claims that the simpler a theory the better. The MDL principle states that the theory with the shortest description length is the better theory. RIPPER uses the description length measure in three ways. First, it provides a heuristic for stopping the rule-building loop in IREP. Deciding when to stop the IREP cycle is more an art than a science. For C-KANT, the rule-building cycle terminates if either (1) all of the positive examples have been covered, or if (2) all the negative examples have been covered, or if (3) no progress has been made, that is, no rule was found in a given cycle that covered any new positive examples, or (4) if the MDL of the current rule set and its examples are more than 64 bits longer than the smallest set so far. Cohen found that 64 bits was optimal.[5]

The second use of MDL is in the compression of rules. Each rule in a rule set is examined beginning with the last rule added. Any rule which increases the description length is deleted thus "compressing" the rule set. The third use of the description length measure is in additional optimizing steps. After IREP finishes constructing a rule set, the rule set is optimized to reduce the size of the set and hopefully increase its accuracy. Each rule r in the rule set, in the order it was added, is compared with two alternative rules, a *replacement* rule r' and a *revision* rule r''. The replacement rule,r', is created by growing and pruning a new rule. Pruning is governed by the goal of minimizing error (defined by equation 2) over the entire rule set by comparing the rule set with r to the rule set with r' in place of r. On the other hand, the revision of r, r'', is grown by greedily adding terms to r instead of the empty rule. Finally r is replaced by one of the three r, r', r'' depending on which has the shortest description length after compression.[5]

After optimization the rule set may end up covering fewer examples. For this reason IREP is called again on the uncovered examples. Cohen determined after experimentation that running the optimization step twice was optimal.[5]

4. EVALUATING TEXT CATEGORIZATION

Automatic text categorizers are measured in terms of their "effectiveness." Effectiveness is typically defined using the *contingency table model*. In this model, it is assumed that each document in a collection has been assigned at least one category by an expert and that the expert's decision is correct.

Given an expert's judgment as to the correct category c_i for a document d_j, and an automatic categorizer's guess as to which category document d_j belongs, there are four possibilities. The expert and categorizer can both agree that d_j is in c_i, a result called a *true positive*. The expert can claim that d_j belongs to c_i but the categorizer disagrees, which is a *false negative* result. Or, the categorizer may conclude that d_j belongs in c_i but the expert says it does not, which is a *false positive*. Finally both the expert and the categorizer can agree that d_j does *not* belong in c_i, a *true negative*.

For a given category c_i the number of decisions for each of these four types is counted. Let TP stand for the number of true positives counted for c_i, FP the false positives, FN the false negatives and TN the number of true negatives. These numbers can now be represented by a *contingency table*:

		Expert's Judgment		
		$d_j \in c_i$	$d_j \notin c_i$	
Categorizer's Guess	$d_j \in c_i$	TP	FP	$TP + FP$
	$d_j \notin c_i$	FN	TN	$FN + TN$
		$TP + FN$	$FP + TN$	N

Two important measures borrowed from information retrieval, *recall* and *precision*l,[18,19] can be defined in terms of the entries and marginal values of the contingency table above:

$$recall = TP/(TP + FN) \tag{3}$$

$$precision = TP/(TP + FP) \tag{4}$$

The recall measure is the ratio of the documents correctly categorized over all the documents the expert thought *should* be in the category. Precision is the ratio of the documents correctly categorized over all the documents the categorizer *in fact* placed in the category. Both values are real numbers from 0 to 1.

To use precision and recall to evaluate a set of classifications, the individual precisions and recalls are averaged. Although there are at least two ways of computing these averages, the one usually preferred by researchers is *microaveraging*:

$$microaverage\ precision = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FP_i)} \tag{5}$$

$$microaverage\ recall = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FN_i)} \tag{6}$$

where subscripts indicate that TP_i, FP_i, etc., are measures for the i*th* category and n the total number of categories.[1,19]

For many automatic text categorization systems it is possible to define and use the *breakeven point* of the precision and recall curves as the measure of effectiveness. Rule-based categorizers do not typically return a confidence value. Instead they make a boolean decision that a document belongs to a category or not. Such categorizers place a document in a category or not depending on whether the antecedent of the relevant rule is satisfied by the document's attributes. For this reason, another value called the F_β *function* is used to define effectiveness in rule-based systems. For $0 \leq \beta \leq \infty$,

$$F_\beta = \frac{(\beta^2 + 1) \times P \times R}{\beta^2 \times P + R}, \tag{7}$$

where P and R are averages of precision and recall. β is typically set to 1 because Moulinier and Yang have shown that the breakeven point of a given categorizer is always less than or equal to F_1.[8,20]

5. THE TEST COLLECTION

The test collection used in this study is a subset of a large collection of documents donated to the Information Science Research Institute (ISRI) by the Department of Energy (DOE). Since many of the documents were originally hard copy, they had to be converted to an electronic form using scanning and Optical Character Recognition (OCR) technology. ISRI developed a representative sample of approximately 2,600 OCR documents (140,000 pages) called the Licensing Support Network (LSN) Prototype to study the properties of OCR text in information retrieval and text categorization.[21]

Documents in the LSN concern topics associated with nuclear waste management. A subset of the LSN documents were manually separated by human experts with training in geology and biology into categories defined by the Nuclear Regulatory Commission.[22] This subset, called *Big-DOE*, consists of 1619 documents which were split into 1074 training and 545 testing documents. Porter stemming as well as stop-word removal were applied to the documents.

6. ADDING THESAURI TERMS TO RULES

One approach used in information retrieval to achieve better results is to enhance the indexing or querying of a large document collection by adding terms from thesauri created for these tasks.[11,23] Such a thesaurus was created by professional thesaurus builders for the LSN database.[21,24]

There are three basic inter-term relationships used in thesauri: equivalence, hierarchical, and associative.[23] Equivalent terms include synonyms, abbreviations, and specially coded terms. For example, in the LSN thesaurus, *US DOD* is related in this way to *Department of Defense*. Usually, one term in this relation is a preferred usage and the other non-preferred. In the LSN thesaurus, *Department of Defense* is preferred to the abbreviation *US DOD*. The preferred term is prefixed by *USE* and the non-preferred by *UF*.

Query Expansion: Microaverage Changes						
	Recall	Precision	F_1	Δ Recall	Δ Precision	Δ F_1
def	0.941	0.564	0.706			
BT1	0.949	0.557	0.702	+ 0.005	− 0.007	− 0.004
BT3	0.941	0.541	0.687	+ 0.000	− 0.016	− 0.019
NT1	0.938	0.557	0.699	− 0.003	− 0.007	− 0.007
NT3	0.932	0.564	0.703	− 0.009	+ 0.000	− 0.003
RT1	0.936	0.535	0.681	− 0.005	− 0.029	− 0.025
RT3	0.949	0.542	0.690	+ 0.005	− 0.022	− 0.016
UF1	0.954	0.556	0.702	+ 0.013	− 0.008	− 0.004
UF3	0.951	0.554	0.701	+ 0.010	− 0.010	− 0.001
UN1	0.940	0.552	0.696	− 0.001	−0.012	− 0.010
def = default (no expansion)						
BTn = nth level BT thesaurus; NTn = nth level NT thesaurus						
RTn = nth level NT thesaurus; UFn = nth level UF thesaurus						
UNn = nth level BTn + NTn + RTn + UFn						

Figure 2. Changes in Recall, Precision, and F_1 over Big-DOE using Query Expansion

The second type of inter-term relationship is the hierarchical. Hierarchical terms are related with respect to levels of superordination and subordination. The superordinate term will be prefixed with *BT* for "broader than." So, the term *Metamorphic Rocks* will be prefixed with *BT* in relation to the subordinate *Amphibolites*. The prefix *NT* appears before subordinate terms.

The third type is the associative relationship. This group is reserved for terms which have some kind of conceptual relationship but that relationship cannot be characterized by either of the previous two. Terms which are associatively related are marked by the abbreviation *RT* for "related to." In the LSN thesaurus, the phrase *Metamorphic Rocks* is related to *Basement Rock* in this way.

The LSN thesaurus was broken into four thesauri: the BT thesaurus, NT thesaurus, RT thesaurus, and the UF thesaurus. The first consists of all terms which have the hierarchical "broader than" relationship, and the second, the terms with the "narrower than" hierarchical relationship. The third contains "related to," that is, associatively related, terms, which are neither synonyms nor hierarchically related. The UF thesaurus contains synonyms in the "use for" relationship.

7. QUERY EXPANSION

We applied two approaches for using thesaurus terms. The first follows the model of query expansion from information retrieval.[10,11] In this "query expansion" approach, thesaurus terms were added to both training documents and testing documents using the same thesaurus for each. After expanding the documents, the rule learning algorithm was applied. Since the set of terms representing a document changed as various thesauri were applied, different rules were learned.

We wondered in particular if hierarchically related terms might not aid the categorizer by helping the documents to cluster. For example, suppose one document contains the term *amphibians* and another *birds*. C-KANT will add *vertebrates* to both documents when using the BT thesaurus. If originally the documents had nothing common, now they would both contain the term *vertebrates* thus perhaps giving the rule-learning algorithm reason to think that the two documents were related.

Because terms are added to documents, there can be *levels* to this addition. For example, if *animals* appears in a document, then the NT thesaurus would add *vertebrates*. If consulted again, NT will now add all NT-related terms for *vertebrates*, such as *amphibians*, *birds*, etc. Once these terms are added, all the NT-related terms of those terms will be added as well.

Junker Expansion: Microaverage Changes						
	Recall	Precision	F_1	Δ Recall	Δ Precision	Δ F_1
def	0.941	0.564	0.706			
BT1	0.943	0.563	0.705	+ 0.002	− 0.001	− 0.001
NT1	0.945	0.560	0.703	+ 0.004	− 0.004	− 0.003
RT1	0.945	0.560	0.703	+ 0.004	− 0.004	− 0.003
UF1	0.943	0.564	0.706	+ 0.002	+ 0.000	+ 0.000
UN1	0.943	0.565	0.707	+ 0.002	+ 0.001	+ 0.001
def = default (no expansion)						
BTn = nth level BT thesaurus; NTn = nth level NT thesaurus						
RTn = nth level NT thesaurus; UFn = nth level UF thesaurus						
UNn = nth level BTn + NTn + RTn + UFn						

Figure 3. Changes in Recall, Precision, and F_1 over Big-DOE using Junker Expansion

C-KANT is able to train using a user-specified number of levels. In the following we will generally distinguish tests with respect to the number of levels consulted in the thesaurus. So, for example, BT2 will mean that 2 levels of BT terms were added to the documents.

Figure 2 lists the microaveraged changes in precision, recall, and the F_1 measure for various types of thesaurus expansions using the query expansion approach. In particular it compares a default test on the Big-DOE test collection with tests using expansions at one and three levels of the BT, NT, RT, and UF Thesauri. In addition to expanding the texts using each of the four thesauri alone, one test was run using one level of all four, called UN1.[10] In all tests, terms were stemmed and stop-words were removed.

Although there were modest gains in recall for some of the query expansion tests, these gains were balanced with a loss in precision. The change in the F_1 measure over the tests, shown in the ΔF_1 column of figure 2, is for the worse in each case where thesaurus terms were added. We cannot conclude, however, that the query expansion approach worsens the outcome. An analysis of variance (ANOVA) test over the results for each category revealed that the change in F_1 was not significant. Thus, the query expansion approach did not significantly worsen the categorizer, but neither did it improve it.

8. JUNKER RULE EXPANSION

In the Junker expansion, documents were not expanded. As a rule is constructed, however, if a term is found which appears in a thesaurus, it is replaced by a related term in a candidate rule. If C-KANT determined that a new rule was better in the sense that the rule had a higher information gain as defined by equation 1, the new rule replaced the original candidate.

The motivation of this approach is similar to that of query expansion. Junker and Abecker hoped for two benefits from a thesaurus. First, that the thesaurus might help cluster several rare features, as *birds* and *amphibians* might cluster to *vertebrates* in our earlier example. Second, they hoped that features outside of the training set might aid categorization.[2]

Consistent with Junker's work, thesaurus terms were very rarely selected as parts of rules. Hence the rule building algorithm very seldom determined that a given thesaurus term provided a better rule than its replacement term. Despite a modest gain in the UN1 test (figure 3), the results for the Junker expansion again were not significantly better than the default, as measured by an ANOVA test. The results were not significantly worse, either, indicating that neither approach significantly changed the effectiveness of the categorizer.

9. CONCLUSION

The results of the tests reported here suggest that even a thesaurus which is constructed for a specific collection will not aid a rule-based categorizer for OCR texts. The query expansion technique used here may have succumbed to the one of the general problems of query expansion. Just as adding terms to a query in information retrieval can increase irrelevant retrievals and lower precision, so the added terms may have blurred the boundaries between categories and made these boundaries more difficult to induce.

There are several avenues for further research on this topic. One reason the Junker approach very seldom added knowledge to the rules was that when a thesaurus term was added to a rule it weakened the rule's information gain as compared to the unaltered rule. This is because the thesaurus term probably appeared few times if at all in documents in the training set. Perhaps thesaurus altered rules should be added along with the original rule and not be held to as high a standard as information gain.

In addition, as mentioned earlier, C-KANT made use of common dimensionality reduction techniques such as stemming and stop word removal. However, Taghva has recommended that more extensive reductionality techniques should be applied when categorizing OCR text.[3] Those results were derived from tests using the statistical categorizer BOW.[25] It would be interesting to know if rule-based categorizers would also perform better with more aggressive dimensionality reduction.

Finally a more ambitious approach to rule-building may have better results for text categorization as well. CLIPS allows for a much more sophisticated representation of documents using its frame structure.[12] In our study, documents were only represented as a "bag of terms" occurring in the "body" of the documents. Some studies have incorporated the relative position of terms into rules, but with little improvement in performance.[26] Perhaps describing a document in a more complex fashion using information from the OCR process itself could lead to a better categorizer.

REFERENCES

1. F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys* , 2002. Accepted for publication.
2. M. Junker and A. Abecker, "Exploiting thesaurus knowledge in rule induction for text classification," in *Proceedings of RANLP-97, 2nd International Conference on Recent Advances in Natural Language Processing*, R. Milkov, N. Nicolov, and N. Nikolov, eds., pp. 202–207, (Tzigov Chark, BL), 1997.
3. K. Taghva, T. A. Nartker, and J. Borsack, "Recognize, categorize, and retrieve," in *Proc. of the Symposium on Document Image Understanding Technology*, pp. 227–232, Laboratory for Language and Media Processing, University of Maryland, (Columbia, MD), April 2001.
4. C. Apte, F. Damerau, and S. M. Weiss, "Automated learning of decision rules for text categorization," *ACM Transactions on Information Systems 1994* **12**(3), pp. 233–251, 1994.
5. W. W. Cohen and Y. Singer, "Context-sensitive learning methods for text categorization," *ACM Transactions on Information Systems* , pp. 141–173, 1999.
6. H. Li and K. Yamanishi, "Text classification using ESC-based stochastic decision lists," in *Proceedings of CIKM-99, 8th ACM International Conference on Information and Knowledge Management*, pp. 122–130, (Kansas City, MO), 1999.
7. I. Moulinier and J. Ganascia, "Applying an existing machine learning algorithm to text categorization," in *Connectionist, statistical, and symbolic approaches to learning for natural language processing*, pp. 343–354, Springer Verlag, 1996.
8. I. Moulinier, G. Raškinis, and J. Ganascia, "Text categorization: a symbolic approach," in *Proceedings of SDAIR-96, 5th Annual Symposium on Document Analysis and Information Retrieval*, pp. 87–99, Information Science and Research Institute, (Las Vegas, NV), 1996.
9. C. Wenzel and R. Hoch, "Text categorization of scanned documents applying a rule-based approach," in *Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR)*, pp. 333–346, Information Science and Research Institute (ISRI), (Las Vegas, NV), 1995.
10. E. Dimitrova, "Retrieval effectiveness for OCR text using thesauri," Master's thesis, University of Nevada, Las Vegas, 1999.

11. P. Srinivasan, "Thesaurus construction," in *Information Retrieval: Data Structures and Algorithms*, W. B. Frakes and R. Baeza-Yates, eds., pp. 161–218, Prentice-Hall, 1992.
12. J. Giarratano and G. Riley, *Expert Systems: Principles and Programming*, ITP, 3rd ed., 1998.
13. R. Savely and C. Culbert, *CLIPS Reference Manual, Version 6.10*, 1998. URL: `<http://www.ghgcorp.com/clips/download/documentation/>` (viewed Jan. 28, 2002).
14. J. Fürnkranz and G. Widmer, "Incremental reduced error pruning," in *Proceedings of the 11th Annual Conference on Machine Learning*, pp. 70–77, 1994.
15. J. R. Quinlan, "Learning logical definitions from relations," *Machine Learning* , 1990. Introduction to FOIL.
16. W. W. Cohen, "Fast effective rule induction," in *International Conference on Machine Learning*, pp. 115–123, 1995.
17. J. Quinlan, "Mdl and categorical theories (continued)," in *Proceedings of the 12th International Conference on Machine Learning*, pp. 464–470, (Lake Tahoe, CA), 1995.
18. W. B. Frakes, "Introduction to information storage and retrieval systems," in *Information Retrieval: Data Structures and Algorithms*, W. B. Frakes and R. Baeza-Yates, eds., pp. 1–12, Prentice-Hall, 1992.
19. D. D. Lewis, "Evaluating text categorization," in *Proceedings of the Speech and Language Workshop*, pp. 312–318, 1991.
20. Y. Yang and X. Liu, "A re-examination of text categorization methods," in *22nd Annual International SIGIR*, pp. 42–49, 1999.
21. K. Taghva, T. Nartker, J. Borsack, and A. Condit, "Unlv-isri document collection for research in OCR and information retrieval," in *Proc. IS&T/SPIE 2000 Intl. Symp. on Electronic Imaging Science and Technology*, (San Jose, CA), January 2000.
22. NRC, "Regulatory guide 3.69: Topical guidelines for the licensing support system." URL: `<http://www.nrc.gov/reading-rm/doc-collections/reg-guides/fuels-materia%ls/active/03-069>`(viewed Oct. 25, 2002), 1996.
23. J. Aitchison, A. Gilchrist, and D. Bawden, *Thesaurus construction and use: A practical manual*, Fitzroy Dearborn, 4th ed., 2000.
24. K. Taghva, J. Borsack, and A. Condit, "The effectiveness of thesauri-aided retrieval," in *Proc. IS&T/SPIE 1999 Intl. Symp. on Electronic Imaging Science and Technology*, (San Jose, CA), January 1999.
25. A. McCallum, "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering." `http://www.cs.cmu.edu/~mccallum/bow`, 1996.
26. W. W. Cohen, "Text categorization and relational learning," in *Proceedings of International Conference on Machine Learning ICML-95*, pp. 124–132, 1995.