FEATURE ENGINEERING FOR TEXT CLASSIFICATION

Sam Scott
Cognitive Science (IIS)
Carleton University
Ottawa, ON, K1S 5B6 (Canada)
sscott2@chat.carleton.ca

Stan Matwin
Computer Science Department
University of Ottawa
Ottawa, ON, K1N 6N5 (Canada)
stan@site.uottawa.ca

Abstract

Most research in text classification has used the "bag of words" representation of text. This paper examines some alternative ways to represent text based on syntactic and semantic relationships between words (phrases, synonyms and hypernyms). We describe the new representations and try to justify our suspicions that they could have improved the performance of a rule-based learner. The representations are evaluated using the RIPPER rule-based learner on the Reuters-21578 and DigiTrad test corpora, but on their own the new representations are not found to produce a significant performance improvement. Finally, we try combining classifiers based on different representations using a majority voting technique. This step does produce some performance improvement on both test collections. In general, our work supports the emerging consensus in the information retrieval community that more sophisticated Natural Language Processing techniques need to be developed before better text representations can be produced. We conclude that for now, research into new learning algorithms and methods for combining existing learners holds the most promise.

Keywords: text classification, feature engineering, text representations, phrases, synonyms, hypernyms, WordNet, RIPPER, semantic relationships

(From Extractor: representations, phrases, classification, feature, learning, learning algorithms, performance, machine learning, RIPPER, semantic relationships, test collections)

Introduction

Text classification is the task of automatically placing pre-defined index labels on previously unseen documents. Used in document indexing, e-mail filtering, web browsing, and personal information agents, text classification is an active and important area of research where machine learning and information retrieval research intersect. New feature selection techniques [Yang and Pederson, 1997; Ng et al., 1997] and learning algorithms [Joachims, 1998; Lam and Ho, 1998; McCallum et al., 1998] have produced good results on a number of standard test collections, but the vast majority of this research uses the "bag of words" representation of text, where each feature corresponds to a single word or stem. The aim of this paper is to examine as exhaustively as possible some alternative ways to represent text. We look at new representations based on syntactic and semantic relationships between words, and evaluate these new representations on two major document collections.

First, we discuss phrase-based representations. Although previous work failed to find performance improvements using phrases with statistical learning algorithms (see [Lewis, 1992b] for a review), we thought there was still reason to believe that phrases could improve the performance of a symbolic, rule-based learning algorithm. The recent development of RIPPER [Cohen, 1995a], a fast rule-based learner that performs well on bag of words representations, opens up the possibility of using phrase based representations for symbolic learning. Next, we look at new representations based on synonyms and hypernyms. Previous work using semantic relationships in text classification had not looked at a radical change of representation in the way that we had in mind, and usually used manual intervention to sort out semantic ambiguities (see [Rodríguez et al., 1997] for example.) In contrast, our work explores a fully-automatic change of representation for incorporating semantic relations.

In the sections that follow, we describe experiments on 8 automatically-generated representations based on words, and phrases, synonym and hypernym relationships. In no case do we find that the representations on their own produce a significant performance improvement, although combining classifiers based on different representations does produce some benefit. In general, this work supports the emerging consensus in the information retrieval community that more sophisticated Natural Language Processing techniques need to be developed before better text representations can be produced. In the short term, research into new algorithms based on combining classifiers probably holds the most promise.

1. A Quick Look at RIPPER

Before moving on to discuss the new text representations, we need to quickly introduce the learning algorithm we used. RIPPER was developed by William Cohen [1995a] based on repeated application of Furnkranz and Widmer's [1994] IREP algorithm. Like other rule-based learners, RIPPER grows rules in a greedy fashion guided by an information gain heuristic. RIPPER is comparable in accuracy to similar algorithms such as c4.5, but is significantly more efficient [Cohen 1995a]. This efficiency combined with RIPPER's implementation of set-valued features [Cohen, 1996b] allows learning in significantly larger feature spaces than would have been possible with c4.5.

RIPPER has already been applied to a number of standard problems in text classification with quite promising results [Cohen, 1996a; Cohen and Singer, 1996]. It is important to emphasize that RIPPER is a rule-based machine learning system that has made its mark in a field dominated by purely statistical algorithms such as Naive Bayes, Widrow-Hoff, or k-Nearest Neighbor. The high-dimensionality of most representations of text has in the past lead researchers away from rule- or tree-based learning systems. This makes RIPPER interesting since most conclusions about the effectiveness of various representations have been drawn in a context that may not apply to a rule-based learner.

2. The Bag of Words

The representation that dominates the text classification literature is known as the "bag of words". For most bag of words representations, each feature corresponds to a single word found in the training corpus, usually with case information and punctuation removed. Often infrequent and frequent words are removed are removed from the original text. Sometimes a list of *stop words* (functional or connective words that are assumed to have no information content) is also removed.

Sometimes this word set is used with no further processing but more typically, there is some attempt to make the features more statistically independent. The most common way achieve independence is to remove suffixes from words using a *stemming* algorithm such as the one developed by Lovins [1968]. Stemming has the effect of mapping several morphological forms of words to a common feature. For example the words *learner*, *learning*, and *learned* would all map to the common root *learn*, and this latter string would be placed in the feature set rather than the former three.

Stemming and stop-word removal are so widely regarded as useful that they are almost universally used in text classification experiments. However, there is reason to doubt the universality of their effectiveness for retrieval [Riloff, 1995]. The chief advantage of infrequent word removal is reduction of feature set size, but since the number of word features did not pose a problem for RIPPER, we did not use this technique. The chief historical advantage of stop-word and frequent word removal is reduction of data set size, but since this was not a concern, we chose not to use this technique either. As for stemming, the advantages and disadvantages are less clear, so both stemmed and unstemmed representations were explored.

3. Phrase-based Representations

A very basic observation about bag of words representations is that a great deal of the information from the original document is discarded. Paragraph, sentence and word order is disrupted, and syntactic structures are broken. The end result is that the text is rendered incoherent to humans in order to make it coherent to a machine learning algorithm. The goal of using phrases as features is to attempt to preserve some of the information left out of the bag of words.

The observation that words alone do not always represent true atomic units of meaning is not new. David Lewis undertook a major study of the use of noun phrases for statistical classification as part of his Ph.D. thesis [Lewis, 1992b]. He reviewed the existing information retrieval literature on the use of syntactic phrases and found that most studies had not been able to demonstrate much improvement over word-based indexing. To try to explain these results, Lewis suggested that previous work fared poorly because phrase-based representations: a) contain a high number of terms, b) have an uneven distribution of feature values, c) contain many redundant features, and d) contain a lot of noise. All these characteristics pose theoretical problems for Bayesian classification. Lewis attempted to correct these problems by using a clustering algorithm to group phrase features into *meta-features*, but the new representation did not produce any improvement either [Lewis, 1992a; 1992b].

Despite the negative results obtained by this research, there was still reason to believe that in some cases phrase-based representations might improve the performance of rule-based learners. For example, the phrase "machine learning" has a highly specific meaning that is separate and distinct from the words "machine" and "learning". In a situation where the classes to be learned include *Artificial Intelligence*, *Machine Tools*, and *Education*, it is conceivable that the phrase "machine learning" will yield a high information gain for *Artificial Intelligence*, even though the individual words yield a low gain. In this case, a potentially powerful rule such as "machine & learning → *Artificial Intelligence*" might never be learned by RIPPER – the greedy search heuristic would prevent the addition of either individual word to the empty rule. But if the feature set included two-word phrases, then the rule: "'machine learning' → *Artificial Intelligence*" could be learned more easily. Statistical algorithms like naïve Bayes are immune to this sort of worry, as they always work with all available features.

The main problem in converting to a bag of phrases is the huge potential increase in the number of features (if there are n words, there are potentially n^k sequences of length k.) In order to keep the task tractable, some solution to this problem must be found. Cohen [1995b; 1995c] attempted to solve the problem using FLIPPER, a first-order version of RIPPER, to learn in a feature space that included phrase-defining predicates. This technique did not produce an improvement, but the result is not as strong as it could be because the argument about the greedy search heuristic employed by RIPPER also applies to FLIPPER. Only a representation that includes phrases as atomic features can overcome this problem.

The only way to avoid the ballooning feature space while still retaining phrases as atomic features is to employ a selection strategy. Luckily, not all sequences of words are equally meaningful. For example the sentence "The quick brown fox jumps over the lazy dogs," contains a number of potentially useful phrases such as "brown fox", "quick brown fox", and "lazy dogs" in addition to a larger number of word sequences that are probably not useful for classification: "The quick", "fox jumps", "over the", and so on. So the new phrase-based representation should be amenable to an aggressive selection strategy aimed at weeding out the word sequences that are least likely to be useful. We now consider two methods for performing this task.

4.1 Noun Phrases

Motivated by the observation that phrases indicating the objects and subjects of a text are likely to be the most useful, the first method selects only those word sequences that are recognized as noun phrases. For instance, if the system encounters our example sentence, it should recognize that "quick brown fox" is a noun phrase while "the quick" and "over the" are not. Importantly, it should also recognize that "brown fox" and "fox" are valid noun phrases as well – there is no a priori way to determine which of these phrases will yield the highest information gain. There is an

important difference here from [Lewis, 1992a; 1992b] where a bracketing technique was used that was biased towards the longest phrases, and specifically excluded single word phrases – "quick brown fox" would be extracted but not "brown fox" or "fox".

Extracting noun phrases from a document requires two separate algorithms. The first is a tagging algorithm to assign part of speech tags (*noun*, *verb*, *preposition*, etc.) to the individual words, and the second is an algorithm to group the tagged words into noun phrases. Eric Brill's rule-based part of speech tagger [Brill, 1992; 1994] was used in its default configuration to assign the part of speech tags, and grouping was done with a simple regular expression. The two algorithms together are referred to as the Noun Phrase Extractor (NoPE).[1] For our purposes a *noun phrase* was defined to be a sequence of nouns or adjectives terminating in a noun. In regular expression form this is represented as "Noun Phrase = {Adjective, Noun}* Noun." This definition is applied at every possible point in a text to look for matching phrases. Complex noun phrases such as "the quick brown fox that got away" will be missed, but NoPE does accomplish its stated task: the number of features is reduced by selecting many of those that represent the objects or subjects of discussion.

4.2 Key Phrases

NoPE uses syntactic information to select a useful subset of the all the available phrases in the text. This section describes a statistical method of achieving the same goal. The Extractor system developed by Peter Turney [1999] uses a statistical algorithm to try to extract the most *meaningful* phrases from a document. The algorithm used by Extractor is itself the result of learning phase, during which it was trained to assign keyphrases to a document so as to mimic the choices a human would make. Assuming the task is performed well, Extractor's keyphrases should be highly meaningful, unambiguous, and indicative of the content of the document – the very properties desired in the choice of phrase features for classification.

Turney compared the performance of Extractor to other keyphrase extraction algorithms and found that Extractor's choices matched human choices better than the other algorithms on a number of test corpora [Turney, 1999]. Of course, all that can be inferred from this is that Extractor was good at its stated task – matching the keyphrases chosen by humans. It does not necessarily mean that Extractor will choose good phrases for classification. Nevertheless, given the intuition that author-chosen keyphrases will be highly meaningful and unambiguous, and given that the Extractor is faster than the Brill tagger, it was worth exploring its use in addition to NoPE.

4.3 Constructing the Representations

Both NoPE and Extractor are used in the same way to generate a phrase-based representation. First phrases are extracted from every document in the corpus and assembled into a global list. Part of speech tags are removed from the words if necessary, and stemming and further feature selection can be performed on this list. (In the case of stemming, each word of the phrase would be replaced by its stem.) The result is used as the set of features for classification. A final pass is then required to assemble feature vectors based on this set.

4. Hypernym-based Representations

The bag of words representation also ignores semantic relationships between words. In particular, information about synonymy is not directly used, nor is the deeper semantic relationship of hypernymy (a linguistic term for the "is a" relationship – a knife is a weapon,

[1] Links to all the tools and data referred to in this work can be found on the National Research Council of Canada web site - http://ai.iit.nrc.ca/II_public/Classification/resources.html.

therefore "weapon" is a hypernym of "knife".) A rule-based learner could be aided by a feature engineering method that mapped words with low information gain to common hypernyms that might yield a higher information gain. We used this observation previously to build a new text representation that produced significant increases in accuracy on some small data sets [Scott and Matwin, 1998]. The current research builds on the preliminary work by applying a similar technique to much larger data sets.

4.1 WordNet

The feature engineering methods described here rely on the use of WordNet, a large on-line thesaurus that captures both synonym and hypernym relationships between words [Miller, 1990; Fellbaum, 1998]. Figure 1 shows an idealized piece of the WordNet hierarchy. The nodes in WordNet are referred to as "synsets". Each synset represents a particular semantic meaning shared by a group of words and phrases. Each word or phrase in the synset is a synonym of the others with respect to this particular meaning, and a word or phrase with more than one meaning will appear in more than one synset. The hypernym relation is modeled as arcs between the nodes.

We know of only one previously published attempt to make use of WordNet for text classification. Rodríguez et al. [1997] used WordNet with linear discriminant learning algorithms to produce better classification performance on the Reuters-22173data set.[2] They were able to take advantage of the fact that Reuters topic headings often correspond to words that appear in the text of the documents. They compiled a list of all the topic headings and looked up these words in WordNet to find synonyms. This list of synonyms was then used to bias the learning algorithm for each topic by increasing the initial weights of the features corresponding to the synonyms of that topic. This entire process was done by hand and took advantage of the researchers' a priori knowledge of the Reuters domain. No change of representation was involved, and the use of WordNet was limited to the synonymy relation only.

Some researches have also applied WordNet to information retrieval tasks. Sharon Flank [1998] reported good results using many of the WordNet semantic relations, including hypernymy, to expand short descriptions of pictures for retrieval. Ellen Voorhees looked at the possibility of using WordNet for document indexing [Voorhees, 1993] and query expansion [Voorhees, 1994]. The latter work found that when the queries were expanded or converted to synsets manually some retrieval tasks were improved significantly, but automatic expansion posed problems due the difficulty of choosing the correct sense for the word in a given context (word sense disambiguation).

WordNet's organization of synsets into parts of speech allows for some partial disambiguation, but each part of speech can still contain many different senses for each word. The problem of automatic disambiguation has been tackled by a number of researchers (see [Ide and Véronis, 1998] for a summary of recent work), but its difficulty has hampered past attempts to use WordNet in the information retrieval community. Often, the problem is solved by manually selecting the correct word sense – a method that was not practically possible in the current work. The changes of representation discussed in this section proceed on the idea that in the presence of a large amount of training data, the problem of disambiguation can be partially sidestepped.

4.2 Constructing the Representations

If every word could be correctly disambiguated, a function similar to stemming could be performed in which synonymous words are mapped to the same meta-feature. The difference is

[2] Reuters-22173 is the previously released version of Reuters-21578.

that the mapping would be semantically rather than morphologically based. To extend this idea, the hypernym links could also be followed through the semantic network to generalize by mapping to the hypernym senses of each word. For instance, in the preliminary work [Scott and Matwin, 1998] a class of documents was learned using both a word-based and a hypernym-based representation. The hypernym *weapon* often appeared in the rules learned using the hypernym-based representation, but the words mapping to the *weapon* synset never appeared in the rules learned using the word-based representation. It seems the individual words that were instances of the concept *weapon* did not yield enough information gain on their own to be selected, but when they were all mapped via WordNet to an appropriate meta-feature, RIPPER was able to make use of them, and the resulting hypotheses were significantly more accurate. This benefit was obtained without any automatic word sense disambiguation. Including all synsets without disambiguation produced a noisy feature space containing some highly informative features, and RIPPER was still able to find them among all the noise. So even though "knife" had hypernyms other than *weapon*, the *weapon* hypernym was still be included and available for classification.

Following on these observations, no disambiguation was attempted in the current work either. Instead all senses returned by WordNet were judged equally likely to be correct, and all of them were included in the feature set. Synsets were identified for the feature set by tagging the documents, then looking up all noun and verb hypernyms in WordNet and assembling a list of all synsets occurring in the output. A second pass converted documents to feature vectors based on this new representation. This process was influenced by the value of a parameter *h* that controls the *height of generalization*. This parameter was used to limit the number of steps upward through the hypernym hierarchy for each word. At height $h=0$ only the synsets that contained the words in the corpus were counted (i.e. synonyms). At height $h>0$ the same synsets were counted as well as all the hypernym synsets that appeared up to *h* steps above them in the hypernym hierarchy. The best value of *h* for a given text classification task depends on characteristics of the text such as use of terminology, similarity of topics, and breadth of topics. It also depends on the characteristics of WordNet itself.

5. Methodology

5.1 Test Collections

The two data sets used for this study were the Reuters-21578 corpus of newswire stories, and a corpus of folk song lyrics called The Digital Tradition (DigiTrad). Reuters is well established and has been used as the basis of much recent text classification work [Joachims, 1998; Lam and Ho, 1998]. Like other researchers, we used the "ModApte" testing/training split for these experiments, which results in 9603 training documents and 3299 testing documents, each assigned to 0 or more of the 90 topic labels. DigiTrad was first introduced for text classification research in [Scott and Matwin, 1998]. Each folk song in the DigiTrad collection has been assigned one or more keywords from a fixed list, thus defining a classification task. We used the "DT100" testing/training split [Scott, 1998] which results in 4333 training and 2166 testing documents, each assigned 0 or more of the 33 keyword labels. (Only those keywords represented by 100 or more examples in the entire collection were used.) Some statistics for the two data sets are shown in table 1.

Not all types of text are equally difficult to classify, and the collections chosen for this research represent two opposite extremes. Reuters consists of articles written purely as a source of factual information. The writing style tends to be direct and to the point, and uses a restricted vocabulary to aid quick comprehension. It has been observed that the topic headings in Reuters

tend to consist of words that appear frequently in the text, and this observation has been exploited to help improve classification accuracy [Rodríguez et al., 97]. DigiTrad, where the texts make heavy use of metaphoric, rhyming, unusual and archaic language, is a good example of the opposite pole. The texts are far less succinct and to the point, and the vocabulary is much less restricted (this latter difference shows up in the comparatively large number of unique words in the collection reported in table 1.) Consequently, the DigiTrad classes should be harder to automatically classify than Reuters.

6.2 Evaluation Methods

In order to compare the current work with that of other recent studies, we computed the micro-averaged breakeven point of precision and recall instead of the more traditional statistic of average cross-validated accuracy. The breakeven point is uncommon in the machine learning literature, but is the measurement of choice in the information retrieval community. For the benefit of readers unfamiliar with this literature, we briefly describe the technique here. Parallel discussions can be found in a number of places, such as [Lewis, 1992b].

Text classification tasks are unlike normal machine learning problems in two respects: examples can be given multiple class labels (meaning separate binary classifiers must be trained for each), and the positive examples of each class are usually in a very small minority (see table 1). These two characteristics combined mean that a plain accuracy statistic is not adequate to evaluate performance. To deal with the unbalanced nature of the classes, precision and recall are used instead of accuracy. Precision is the proportion of examples labeled positive by the system that were truly positive, and recall is the proportion of truly positive examples that were labeled positive by the system (see figure 2).

A process called micro-averaging is used to deal with the multiple classifiers in a given task. The confusion matrices for each class are added component-wise to yield an aggregate matrix for the entire task, and then the precision and recall statistics are computed from this matrix. This component-wise addition is made possible by treating each matrix as representing a binary classification into either a *positive* or *negative* class. The 2×2 matrix resulting from this addition represents the overall picture for all classes.

Most learning algorithms contain parameters that can be varied, often with the effect of trading off precision against recall. (In RIPPER the *loss ratio* parameter adjusts the relative weight of false positive and false negative classifications in a binary task.) To obtain a single number representing the performance of the system, the learning task is repeated for various values of these parameters. Each run yields micro-averaged precision and recall statistics that are interpolated (or extrapolated if necessary) to yield the hypothetical point at which precision and recall are equal. This is the micro-averaged breakeven point.

6.3 Experimental Design

Experiments were performed on each test collection using 8 different representations. Table 2 summarizes these representations and shows the number of features in each. (Note the acronyms for each representation that will be used from here on.) We used stemmed and unstemmed versions of each of the bag of words, noun phrase and keyphrase representations, and tried two versions of the hypernym representation corresponding to $h=0$ and $h=1$. Stemming was performed using the Lovins stemmer [Lovins, 1968]. Feature selection was used only for the noun phrase representations (NP and NP_S), because the number of features originally identified by NoPE was too high (well over 100 000 in both cases). In this case all phrases appearing fewer than

4 times in the corpus were discarded. Feature values were binary in all cases in order take advantage of the speed of RIPPER's set-valued features.

In the preliminary hypernym work where the test collections were much smaller, we used values for *h* ranging from 0 to 9 [Scott and Matwin, 1998]. Unfortunately, we were restricted to low *h* values for the current work because at higher values, RIPPER was simply too slow to be practical. This was probably due to the fact that the representations became less sparse at higher values for *h*. Since much of RIPPER's speed on text classification tasks derives from its sparse vector representation, this understandably caused a significant slowdown in performance.

Default options were used for RIPPER with two exceptions. The first was the enabling of negative tests, allowing the algorithm to use both the "$\in$" and "$\notin$" operation on set valued features. This option was found to marginally improve performance in initial trials. Secondly, the *loss ratio* parameter (*L*) was varied to try to bracket the micro-averaged breakeven points on each task. The values used were {0.5, 1, 2, 4} for Reuters-21578 and {0.25, 0.5, 1, 2} for DigiTrad. Only those classes with 10 or more example documents (60 of the 90 Reuters classes and all of the DigiTrad classes) were classified using RIPPER. The rest were submitted to a default classifier that always returned a negative answer.

6. Results and Discussion

Table 3 shows the micro-averaged breakeven points obtained using RIPPER on each representation. At a first glance, it is clear that the prediction about the relative difficulty of DigiTrad and Reuters was correct. DigiTrad results are consistently less than half as good as Reuters. The results for Reuters can also be compared to recent work by Joachims [1998] to show that RIPPER outperforms Naïve Bayes, Rocchio, and c4.5 algorithms using bag of words, and does about as well as k-Nearest Neighbour. We also now know that new machine learning algorithms such as Combined Support Vector Machines [Joachims, 1998] and Generalized Instance Set [Lam and Ho, 1998] can improve this performance to a breakeven of approximately 0.86 on Reuters. But the main point of comparison for the current study is between the representations based on words (BW and BW_S) and the other representations. Unfortunately, none of the new representations performed significantly better than the bag of words.

The performance of the phrase based representations (NP, NP_S, KP, KP_S) supports the conclusion of past research – that phrases do not add any classification power. It seems that even in a learning paradigm based on greedy search, grouping words into phrases simply does not make the domain any more learnable. Recalling our discussion of the example phrase "machine learning", it seems that the situation in which a phrase is more informative than its component words simply does not arise very often. This being said, there were some small benefits from the use of the NP representation on Reuters. First, RIPPER obtained a slightly higher performance using less complex hypotheses than with BW – approximately 1 fewer rule for every second hypothesis, with the same number of clauses per rule (see [Scott, 1998] for more details). Secondly, the phrase representations reduced the number of features almost as much as stemming (see table 2) without the negative effects on performance (see table 3).

The results from the hypernym representations were the most disappointing. Preliminary work had shown significant improvements on some classes in the DigiTrad domain, and the hope was that this would generalize to the entire corpus [Scott and Matwin, 1998]. But the representations used in this study were considerably poorer than in the preliminary work: for reasons of tractability a low height of generalization had to be used, and the features were binary rather than real-valued density measurements. The latter factor in particular may have removed too much information for RIPPER to pick the meaningful senses out of all the noise.

7. Further Experiments

The final round of experiments evaluated combinations of classifiers based on the different representations. Dietterich [1997] pointed out that any set of high-quality classifiers that make at least partially uncorrelated errors can be expected to produce higher performance when combined with a simple majority vote. He also pointed out that due to the potential for the formation of differently shaped decision boundaries in the feature space, almost any pair of classifiers trained in slightly different conditions can be expected to make partially uncorrelated errors. We felt that the new text representations were probably producing enough variation in the feature space that the majority vote technique should improve RIPPER's predictions when combining hypotheses formed using the different representations. Furthermore, the majority vote technique can easily be applied to text classification, since regardless of the number of classes involved, every decision made by the system is binary.

We experimented with combinations of 3 and 5 classifiers in which a class label was assigned only if the majority of classifiers predicted that it should be. The best results for 3 and 5 classifiers on each test collection are reported in table 4. Note that the Reuters result is comparable to, though still not quite as high as the best reported results of [Joachims, 1998] and [Lam and Ho, 1998]. The fact that this technique improved the micro-averaged breakeven point validates the prediction that the various representations were making uncorrelated errors.

8. Conclusions and Future Directions

Lewis and Sparck Jones [1996] wrote that statistical techniques for classification and retrieval have "picked some of the low-hanging fruit off the tree." They believe that significant advances must be made before Natural Language Processing techniques can be used to improve text classification. The results of our work strengthen their general conclusions by showing that alternative representations do not improve symbolic learners either. In particular we believe that one door is now closed: it is probably not worth pursuing simple phrase-based representations any further. But another door is still partly open: it may be worth the effort, in domains like DigiTrad, to try to find a way to make hypernym representations more tractable for RIPPER. This will probably involve integrating a partial word-sense disambiguation algorithm in the feature formation step to produce a sparser representation and allow higher values for the *h* parameter.

We also found some evidence that alternative representations could serve as the basis for combining classifiers to produce better results. Indeed it is worth pointing out that the only published results that beat our own for Reuters ([Joachims, 1998] and [Lam and Ho, 1998]) are themselves based on combinations of classifiers. Future work in this area may be well worth the effort.

Finally, we have some concerns over the use of micro-averaged breakeven point in the information retrieval literature. Research could be conducted to try to define a more nuanced evaluation technique. For instance, Provost and Fawcett's [1997] ROC curves provide a slightly more sensitive measurement that can be adapted to show the convex hull of precision and recall, providing a more informative picture of performance than the breakeven point. The micro-averaging step of the analysis should also be critically evaluated. A lot of the subtle details of performance get lost in such an aggressive statistical move. Future work will look into defining new performance measurements that do not suffer these shortcomings.

References

Brill, Eric. 1992. A simple rule-based part of speech tagger. *Proceedings of the Third Conference on Applied Natural Language Processing*. ACL. 152-155.

Brill, Eric. 1994. Some Advances in Rule-Based Part of Speech Tagging. *AAAI-94*. 722-727.

Cohen, William W. 1995. Fast Effective Rule Induction. *ICML-95*. 115-123.

Cohen, William W. 1995. Learning to Classify English Text with ILP Methods. *Proceedings of the 5th International Workshop on Inductive Logic Programming*. 3-24.

Cohen, William W. 1995. Text Categorization and Relational Learning. *ICML-95*. 124-132.

Cohen, William W. 1996. Learning Rules that Classify E-Mail. *Proceedings of the 1996 AAAI Spring Symposium on Machine Learning in Information Access.* (www.parc.xerox.com/istl/projects/mlia)

Cohen, William W. 1996. Learning Trees and Rules with Set-valued Features. *AAAI-96*. 709-716.

Cohen, William W. and Singer, Yoram. 1996. Context-Sensitive Learning Methods for Text Categorization. *SIGIR-96*. 307-316.

Dietterich, Thomas G. 1997. Machine Learning Research: Four Current Directions. *Artificial Intelligence Magazine*. Winter, 1997. 97-136.

Fellbaum, Christiane (ed.). 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Furnkranz, Johannes and Widmer, Gerhard. 1994. Incremental Reduced Error Pruning. *ICML-94*. 70-77.

Flank, Sharon. 1998. A Layered Approach to NLP-Based Information Retrieval. *COLING-ACL'98*. 397-403.

Ide, Nancy and Véronis, Jean. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1). March, 1998. 1-40.

Joachims, Thorsten. 1998. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *ECML-98*. 137-142.

Lam, Wai and Ho, Chao Yang. 1998. Using a generalized instance set for automatic text categorization. *SIGIR-98*. 81-89.

Lewis, David D. and Sparck Jones, Karen. 1996. Natural Language Processing for Information Retrieval. *Communications of the ACM*, 39(1). January, 1996. 92-101.

Lewis, David D. 1992a. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task. *SIGIR-92*. 37-50.

Lewis, David D. 1992b. *Representation and Learning in Information Retrieval*. Ph.D. thesis, University of Massachusetts at Amherst. Technical Report 91-93. February, 1992.

Lovins, J. B. 1968. Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, 11. 22-31.

McCallum, Andrew; Rosenfeld, Ronald; Mitchell, Tom; and Ng, Andrew Y. 1998. Improving Text Classification by Shrinkage in a Hierarchy of Classes. *ICML-98*. 359-367.

Miller, George A. 1990. WordNet: an On-line Lexical Database. *International Journal of Lexicography*, 3(4). 235-244.

Ng, Hwee Tou; Goh, Wei Booh; and Low, Kok Leong. 1997. Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization. *SIGIR-97*. 67-73.

Provost, Foster and Fawcett, Tom. 1997. Analysis and Visualization of Classifier Performance: Comparison Under Imprecise Class and Cost Distributions. *KDD-97*. (www.croftj.net/~fawcett/ROCCH)

Riloff, Ellen. 1995. Little Words Can Make a Big Difference for Text Classification. *SIGIR-95*. 130-136.

Rodríguez, Manuel de Buenaga; Gômez-Hidalgo, José María; and Díaz-Agudo, Belén. 1997. Using WordNet to Complement Training Information in Text Categorization. *RANLP-97*. 150-157.

Scott, Sam. 1998. *Feature Engineering for a Symbolic Approach to Text Classification*. Masters Thesis, University of Ottawa Computer Science Department. Technical report TR-98-09. (http://ai.iit.nrc.ca/II_public/Classification/abstract.html)

Scott, Sam and Matwin, Stan. 1998. Text Classification Using WordNet Hypernyms. *Usage of WordNet in Natural Language Processing Systems: Proceedings of the Workshop* (COLING-ACL'98). 45-51.

Turney, Peter. 1999. Learning to Extract Keyphrases from Text. Submitted to *Journal of Artificial Intelligence Research*. (E-mail the author at Peter.Turney@iit.nrc.ca for a copy.)

Voorhees, Ellen M. 1993. Using WordNet to Disambiguate Word Senses for Text Retrieval. *SIGIR-93*. 171-180.

Voorhees, Ellen M. 1994. Query Expansion Using Lexical-Semantic Relations. *SIGIR-94*. 61-69.

Yang, Yiming and Pederson, Jan O. 1997. A Comparative Study on Feature Selection in Text Categorization. *ICML-97*. 412-420.

	Reuters-21578		**DigiTrad**	
	train	**test**	**train**	**test**
documents	9603	3299	4333	2166
classes	90	90	33	33
balance	1.1	1.3	4.1	4.2
total words	1257993	399179	1161244	576156
unique words	27940	16907	43301	29415

Table 1: Some statistics for the two test collections. "Balance" shows the average percentage of positive classifications over all classes.

Name	Description	Features	
		Reuter	**DigiTrad**
BW	bag of words	27940	43301
BW_S	stemmed words	18590	28996
NP	noun phrases	21427	14204
NP_S	stemmed NP	20061	12457
KP	keyphrases	23110	15287
KP_S	stemmed KP	22122	14264
H_0	hypernyms (h=0)	13385	18646
H_1	hypernyms (h=1)	15393	21044

Table 2: Name, description, and number of features for each representation and each test collection.

#	Reuters		DigiTrad	
	reps.	b.e.	reps.	b.e.
1	NP	.827	BW_S	.360
3	BW, NP, NP_S	.845	BW, NP, KP	.384
5	BW, NP, NP_S, KP, KP_S	.849		

Table 4: Best results for 1, 3, and 5 classifiers combined with majority voting for each test collection

representation	micro-averaged breakeven	
	Reuters	**DigiTrad**
BW	.821	.359
BW_S	.810	.360
NP	.827	.357
NP_S	.819	.356
KP	.817	.288[e]
KP_S	.816	.297[e]
H_0	.741[e]	.283
H_1	.734[e]	.281

Table 3: Micro-averaged breakeven points for each representation on each test collection using RIPPER. A superscript "e" indicates an extrapolated rather than interpolated point.

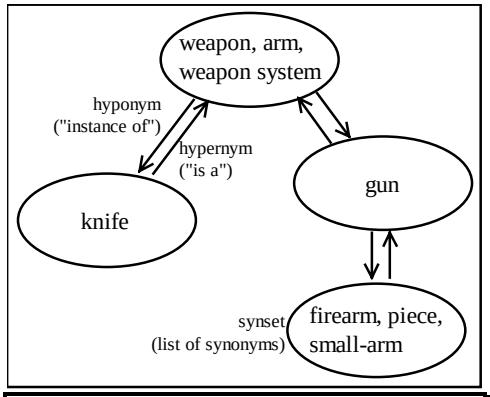

Figure 1: A piece of the WordNet semantic hierarchy (idealized model).

		Hypothesis	
		+ve	-ve
Correct Class	+ve	a	b
	-ve	c	d

$$\text{Accuracy} = \frac{a}{a+b+c+d}$$

$$\text{Precision} = \frac{a}{a+c} \qquad \text{Recall} = \frac{a}{a+b}$$

Figure 2:Conversion of confusion matrix to precision and recall statistics.