To appear in "Machine Learning: Proceedings of the Twelfth International Conference" (ML95)

Text Categorization and Relational Learning

William W. Cohen
AT&T Bell Laboratories
600 Mountain Avenue Murray Hill, NJ 07974
wcohen@research.att.com

Abstract

We evaluate the first order learning system FOIL on a series of text categorization problems. It is shown that FOIL usually forms classifiers with lower error rates and higher rates of precision and recall with a relational encoding than with a propositional encoding. We show that FOIL's performance can be improved by relation selection, a first order analog of feature selection. Relation selection improves FOIL's performance as measured by any of recall, precision, F-measure, or error rate. With an appropriate level of relation selection, FOIL appears to be competitive with or superior to existing propositional techniques.

1 INTRODUCTION

There is increasing interest in using intelligent systems to perform tasks like e-mail filtering, news filtering, and automatic indexing of documents. Many of these applications require the ability to classify text into one of several predefined categories, and in many of these applications, it would be highly advantageous to automatically learn such classifiers from examples.

Many of the learning techniques traditionally used for text categorization problems use a propositional representation for text. These representations ignore such intuitively important properties as the order of words in a document. The first order learning techniques that have been recently developed in the machine learning community [Quinlan, 1990; Quinlan and Cameron-Jones, 1993; Muggleton and Feng, 1992] hold the potential for improved performance on this task, as they can (at least in principle) formulate classifiers that depend on word order.

In this paper, we will experimentally evaluate the performance of FOIL, an off-the-shelf first order learning system, on text categorization problems. The results of this experiment are encouraging, showing that using a first order representation rather than a propositional one can indeed lead to significantly better performance on real-world text categorization problems.

We also consider methods by which the performance of first order learners can be improved on text categorization tasks. In particular, we consider several techniques for performing *relation selection*. Relation selection is a first order analog of feature selection. We show that relation selection can often improve peformance as measured by any of recall, precision, F-measure, or error rate. With an appropriate level of relation selection, FOIL appears to be competitive with or superior to existing propositional techniques.

2 TEXT CATEGORIZATION

Text categorization is the classification of textual documents into one of several fixed classes or *categories*. For example, below is a list of IJCAI paper titles, which have been labelled with a "+" if they should be placed in a machine learning session:

d1	+	improving efficiency by learning intermediate concepts
d2	+	learning dnf by decision trees
d3	+	constructive induction on decision trees
d4	−	a comparison of atms and csp techniques
d5	−	the specialization and transformation of constructive existence proofs
⋮		

One way of building a text categorization system is to learn classification rules from pre-classified data; for instance, one might learn from the data above a general rule for when to schedule a paper in a machine learning session. A typical approach is to represent a document as a feature-vector $\vec{d}$, where the features are words in the vocabulary, and a feature w is true for an instance[1] $\vec{d}$ iff $\vec{d}$ contains the word w. In the example above, the features might be *atms*, *based*, *comparison*, *concepts*, ..., and the final rule might be something

[1] Another common approach is use numeric features that count the number of occurrences of the word w.

like

$$\textit{ML-SESSION} \equiv \textit{learning}=\textit{TRUE} \vee \textit{trees}=\textit{TRUE}$$

Thus the problem of building a text categorizer can be formulated as a traditional inductive learning problem.

Text categorization learning problems tend to have certain common properties. There are usually many features; however, in each example, only a small number of features are set to true. This can cause conventional attribute-value learning systems to be inefficient in terms of time and memory usage.

It is also frequently the case that for a fixed category, there will be many more negative examples than positive examples. The reason for this is that typically, there are many possible categories that can be assigned to an instance, and the categories are not mutually exclusive. (In the example above, for instance, there are many different areas of AI, and a number of papers would be relevant to two or more areas.) Because the categories are not mutually exclusive, it is convenient to learn binary classifier for each possible category, rather than to formulate the problem as a single multi-class learning problem. For most of these binary classifiers, positive instances will be rare.

The proponderance of negative examples can cause problems for some learning systems. It can also lead to methodological complications. For classes that have few positive examples, the error rate of the classifier that predicts the label "−" is usually quite low. Hence success in text categorization problems is often measured not only by error rate, but also by *recall* and *precision*. Recall is the fraction of the time that an actual positive example is predicted to be positive by the classifier, and precision is the fraction of the time that an example predicted to be positive is actually positive. More precisely:

$$\text{recall} \equiv \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false negatives}}$$

$$\text{precision} \equiv \frac{\#\text{true positives}}{\#\text{true positives} + \#\text{false positives}}$$

For convenience, we will define the precision of a classifier that always predicts negative as 1.00.

3 AN EXPERIMENTAL TESTBED

In our experiments we will use the ten text categorization problems described described in [Lewis and Gale, 1994; Lewis and Catlett, 1994]. In each of these problems, the goal is to classify the AP newswire headlines as relevant or irrelevant to topics like "federal budget" and "nielson ratings". A large corpus of 371,454 pre-classified headlines were split into a training set of 319,463 titles and a test set of 51,991 titles. The headlines are an average of nine words long, and the total vocabulary is 67,331 words.

Since a corpus of this size is unwieldy (even for most propositional learning systems) all training was done on ten smaller subsamples, each containing 999 examples. The subsamples were choosen using a technique called *uncertainty sampling* [Lewis and Gale, 1994]. In brief, uncertainty sampling is a heuristic technique for extracting a sample that is small but informative with respect to particular category.

Lewis and Gale showed that training a probabilistic classifier on 999 uncertainty samples gave results competitive with training the classifier on the entire corpus. Lewis and Catlett showed that training C4.5 [Quinlan, 1994] on an uncertainty sample of 999 examples yields error rates comparable to training on a random sample of 10,000 examples.

One important property of uncertainty sampling is that the labels of examples outside the uncertainty sample are not used. Thus, uncertainty sampling is useful if unlabeled data is cheap but labeled data is expensive; this might be the case, for instance, if one was learning to adapt an information filtering system to a single user's needs.

4 THE LEARNING METHOD

The learning system we used in our experiments is FOIL6 [Quinlan, 1990; Quinlan and Cameron-Jones, 1993]. FOIL6 was chosen primarily because of its efficiency and availability.

FOIL6 learns function-free Prolog predicate definitions from examples and *background knowledge*. Both examples and background knowledge are encoded as ground facts. In our experiments, the examples of the target class C are facts of the form $+c(d)$ or $-c(d)$, where d is a constant that identifies a document. The background knowledge we used in our experiments consists of the following relations:

- $w_i(d,p)$ is true exactly when w_i is a word occuring in the document d, and p is the position at which w occurs. (Positions are integers $1,2,3,\ldots,n$, where n is the length of the longest document.) These *word relations* are a straightforward generalization of the normal propositional encoding for examples, which would be a set of unary relations of the form $w_i(d)$.
- $\textit{succ}(p_1,p_2)$ is true when $p_2 = p_1 + 1$.
- $\textit{near1}(p_1,p_2)$ is true when $|p_1 - p_2| \leq 1$.
- $\textit{near2}(p_1,p_2)$ is true when $|p_1 - p_2| \leq 2$.
- $\textit{near3}(p_1,p_2)$ is true when $|p_1 - p_2| \leq 3$.
- $\textit{after}(p_1,p_2)$ is true when $p_2 > p_1$.

Many modern IR systems support queries using the non-word relations *succ*, *after*, and *neark* [Harter, 1986, pages 81–94]. In our experiments, the non-word

Figure 1: FOIL's representation for the example dataset

Examples and word facts:

+mlsession(d1)	improving(d1,1)	efficiency(d1,2)	by(d1,3)	learning(d1,4)	...
+mlsession(d2)	learning(d2,1)	dnf(d2,2)	by(d2,3)	decision(d2,4)	trees(d2,5)
+mlsession(d3)	constructive(d3,1)	induction(d3,2)	on(d3,3)	decision(d3,4)	trees(d3,5)
−mlsession(d4)	a(d4,1)	comparison(d4,2)	of(d4,3)	atms(d4,4)	...
−mlsession(d5)	the(d5,1)	specialization(d5,2)	and(d5,3)	transformation(d5,4)	...
⋮	...				

Other background knowledge:

succ(0,1)	succ(1,2)	succ(3,2)	...	succ(n,$n+1$)
near1(0,1)	near1(1,0)	near1(1,2)	near1(2,1)	...
near2(0,1)	near2(0,2)	near2(1,0)	near2(2,0)	...
near3(0,1)	near3(0,2)	near2(0,3)	near3(1,0)	...
after(0,1)	after(0,2)	after(0,3)	...	after(n,$n+1$)

relations were defined over integers between 0 and $n+1$, whereas positions are integers between 1 and n. This was done deliberately to discourage FOIL6 from constructing clauses requiring specific words to appear at the beginning or end of a document.

As an illustration, Figure 1 shows how the dataset shown on page 1 would be encoded. From this input FOIL6 might learn a classifier such as

mlsession(S) ← learning(S,P1).
mlsession(S) ← decision(S,P1),trees(S,P2),succ(P1,P2).

The learning algorithm used in FOIL is described in detail elsewhere [Quinlan, 1990; Quinlan and Cameron-Jones, 1993]; for the convenience of the reader, however, we will summarize it briefly. FOIL's learning method is a greedy separate-and-conquer algorithm that starts with an empty predicate definition, and then extends the definition by adding one clause at a time. After each clause is constructed, the examples covered by that clause are deleted; this process continues until no more positive examples remain. Clauses are also built using a greedy method: one begins with a clause with an empty body, and repeatly specializes the clause body by adding literals. In determining which literal to add, one considers all possible literals (given the fixed set of background relations) and chooses the one which maximizes an information-theoretic measure of clause quality called *information gain*.

FOIL6 adds a number of refinements to this simple separate-and-conquer learning algorithm, some of which were important for these experiments. FOIL6 includes an method for handling noisy data by stopping construction of a clause (or predicate definition) before it is completely consistent with the data; this was valuable on the text categorization tasks we considered, since in general it is not possible to correctly assign a category to an instance from the text alone.

FOIL6 also includes a backtracking mechanism that extends the greedy search. In preliminary experiments, this feature seemed to cause unpredictable runtime performance. Because of this, and because the evidence that FOIL6's backtracking actually improves performance is weak [Cameron-Jones and Quinlan, 1993], we disabled backtracking in our experiments.[2]

In building clauses, FOIL6 uses two heuristics that allow literals that introduce new variables to be added to the clause, even if these literals have low gain. (The rationale for doing this is that the new variables may make it possible to later add other gainful literals to the clause.) One heuristic is to add, whenever possible, *determinate literals* that introduce new variables: a determinate literal is one that succeeds on every positive example and allows at most one binding of each new variable. Unnecessary determinate literals are removed in a later phase of the algorithm [Quinlan, 1991]. The second heuristic is to introduce non-gainful and non-determinate literals if there are no better alternatives. These "weak" literals also seemed to cause poor performance in preliminary experiments, and so we again disabled this behavior,[3] causing FOIL6 to simply stop extending a clause when no further gainful literals can be found.

5 EVALUATING THE RELATIONAL ENCODING

The goal of our first experiment was to see if there was any potential benefit from using the relational encoding, as opposed to a propositional one. To do this we ran FOIL twice on each of the ten problems, using two different sets of background relations. First we used only the word relations; as this encoding is isomorphic to the standard propositional one[4] these results are

[2]This was done with the command line option "-t0".

[3]This was done with the command line option "-w0".

[4]Notice that there are no relations defined over in-

Table 1: Propositional *vs.* relation encodings

category	default	propositional encoding			relational encoding		
	errors	errors	recall	precision	errors	recall	precision
bonds	60	133	0.517	0.230	132	0.5333	0.2353
burma	93	38	0.634	0.937	38	0.6344	0.9365
neilson	87	568	0.632	0.930	569	0.6207	0.915
tickertalk	40	41	0.000	0.000	43	0.250	0.2000
box-office	42	133	0.786	0.210	133	0.7857	0.2102
budget	197	290	0.315	0.286	290	0.3096	0.2837
dukakis	107	181	0.393	0.266	134	0.4019	0.3805
hostages	228	380	0.316	0.243	383	0.2982	0.2337
ireland	117	165	0.376	0.324	165	0.3675	0.3209
quayle	133	160	0.684	0.435	90	0.7218	0.6443
average	110.4	208.9	0.465	0.302	197.7	0.470	0.354

summarized under the heading "propositional encoding" in Table 1. We also ran FOIL using the full set of background relations listed above. These results are summarized under the heading "relational encoding". The columns of the table are mostly self-explanatory; however note that we record the absolute number of errors (on the entire test set of 51,991 titles) rather than error rate. Also, the column labeled "default errors" records the number of errors made by the *default rule*, which assigns the label "−" to every example.

To summarize, on one category (*tickertalk*) both the relational and propositional encodings gave very low precision and recall; inspection showed that these hypotheses classify almost everything as false. On one category (*nielson*) both encodings gave reasonable recall but low precision. On the other eight categories, both encodings gave reasonable levels of both recall and precision.[5]

The relational and propositional encodings usually have very similar performance, reflecting the fact (confirmed by inspection) that the additional background relations are only occasionally used by FOIL. The relational encoding does have noticibly higher precision on *dukakis* and *quayle*; this leads to a reduction in the number of errors for these categories of about 25% and 45%, respectively. Primarily as a consequence of this, the relational encoding perform a little better on average for all three measures of error rate, precision, and recall.

To compare the two encodings more carefully, we conducted the following test. We examined each of 51,991 test cases and classified it with both hypotheses. If both hypotheses agreed on its classification, the test case was ignored; otherwise, we recorded which hypothesis "won" (*i.e.* gave the correct classification.) We then used a binomial test to assess the likelihood of the null hypothesis that both hypotheses were equally likely to win a disagreement. Table 2 gives the results; for each category we list the won-loss record for the relational encoding; whether the null hypothesis can be rejected, and if so, with what confidence; and finally, the superior encoding, if one can be determined with high confidence.

Table 2: Comparing the encodings

category	relational wins-losses	reject null?		superior encoding
bonds	1-0	no		
burma	0-0	no		
neilson	0-1	no		
tickertalk	1-3	no	(0.683)	
box-office	0-0	no		
budget	2-2	no	(0.000)	
dukakis	48-1	yes	(0.999)	relational
hostages	5-8	no	(0.576)	
ireland	1-1	no	(0.000)	
quayle	76-6	yes	(0.999)	relational
total	134-22	yes	(0.999)	relational

The test confirms our informal remarks above. In terms of error rates, the relational encoding is clearly superior on two of the ten categories, and is not significantly inferior on any of the remaining eight. It is not obvious how to best aggregate behavior across categories; however, if one sums the number of wins and losses on all 10 categories, the relational encoding also comes out as superior with high confidence.

dices; hence the index arguments to the word relations cannot be used to discriminate between positive and negative examples.

[5] Of course, as measured by error rate, most of these classifiers are inferior to the default rule. However, if the loss associated with a false positive prediction was low compared to a false negative prediction (a reasonable assumption) the classifiers would be useful.

Table 3: Frequency-based relation selection

k	propositional encoding			relational encoding			better	reject	better
	errors	recall	precision	errors	recall	precision	by class	null?	overall
1	208.9	0.465	0.303	197.7	0.470	0.354	2r 0p	y (0.9999)	relational
3	145.1	0.510	0.433	131.5	0.506	0.467	3r 1p	y (0.9999)	relational
5	108.9	0.490	0.532	105.0	0.497	0.545	4r 1p	y (0.9949)	relational
7	99.3	0.444	0.560	106.0	0.463	0.536	1r 2p	y (0.9999)	propositional
10	97.4	0.424	0.579	100.6	0.424	0.574	2r 2p	n	
25	98.3	0.325	0.589	98.4	0.347	0.591	1r 2p	n	
50	91.4	0.265	0.690	92.4	0.268	0.717	0r 0p	n	
100	98.4	0.189	0.783	96.9	0.221	0.749	1r 1p	n	
200	104.3	0.111	0.946	111.7	0.122	0.815	1r 1p	y (0.9999)	propositional

Table 4: Frequency-based relation selection for monotone clauses

k	propositional encoding			relational encoding			better	reject	better
	errors	recall	precision	errors	recall	precision	by class	null?	overall
1	259.2	0.310	0.204	252.9	0.319	0.234	3r 0p	y (0.9999)	relational
3	138.3	0.403	0.446	132.0	0.416	0.427	2r 1p	y (0.9997)	relational
5	114.3	0.389	0.540	108.1	0.399	0.538	1r 2p	y (0.9999)	relational
7	102.4	0.371	0.571	104.2	0.378	0.547	2r 1p	n	
10	122.5	0.338	0.508	118.9	0.359	0.515	3r 0p	n	
25	100.6	0.250	0.585	97.0	0.289	0.641	1r 0p	y (0.9980)	relational
50	105.7	0.175	0.626	105.4	0.193	0.636	0r 0p	n	
100	100.9	0.146	0.800	102.4	0.162	0.674	1r 2p	n	
200	104.3	0.111	0.946	111.7	0.122	0.815	1p 1r	y (0.9999)	propositional

6 RELATION SELECTION

As formulated above, a text categorization problem leads to a learning problem in which every word appearing in any example is a feature (in the propositional case) or a background relation (in the first order case). For propositional learners, this huge number of features can often be problematic. Most implementations of propositional learners use a feature-vector representation for examples. For such a system the storage required is proportional to the product of the number of examples and the number of features. Using too many features thus can be impractical for large datasets; for instance, Lewis and Catlett were forced for efficiency reasons to use only a subset of the possible features in their experiments on these datasets with C4.5 [Lewis and Catlett, 1994].

The storage required by FOIL6 is only linear in the size of the original corpus, which makes it computationally feasible to use the full set of word relations. However, although it is feasible, it may be unwise. Certainly not all of the features are necessary to define the concept, and it is widely known that removing irrelevant features can lead to lower error rates [Almuallim and Dietterich, 1991]. We thus investigated several *relation selection* methods.

A simple technique that has proven effective in other text categorization problems is to discard features corresponding to low frequency words: variants of this technique were used by Apté, Damerau and Weiss [1994] and also by Lewis and Catlett. We investigated the natural first order analog of this by discarding word relations for all words that occurred fewer than k times in the training corpus, for various values of k.

The results are shown in Table 3. Here, we indicate for each value of k the number of errors, recall, and precision obtained using each encoding; all of these numbers are averages across the ten categories. We also show (in the "better by class" column) the number of categories for which each encoding is superior at a 95% confidence level using the test applied in Table 2; for instance "4r 1p" indicates that the relational encoding is better on four categories and the propositional encoding is better on one. Finally, we indicate which encoding (if any) appears to be statistically superior if disagreements across all ten categories are aggregated as in Table 2.

The table can be summarized as follows. Frequency-based relation selection improves both recall and precision for very small values of k (up to around $k = 3$). After this point, increasing k continues to improve precision, but at the expense of recall. This appears to be a property of the FOIL6 learning algorithm, which will not include a clause in its hypothesis unless that clause is sufficiently accurate, as estimated on the training data. As relations are eliminated fewer apparently accurate clauses can be found, which leads to lower recall. However, the clauses that *are* found tend to cover more examples, and hence the estimates of their accuracy become more reliable. As k becomes very large FOIL6 has more and trouble finding clauses, and its output approximates the default classifier. Half of the hypotheses generated with $k = 200$ are empty rulesets.

Applying relation selection also changes the relative value of the relational and propositional encoding. When k is set to maximize recall, the relational encoding is better overall, and also better on four of the ten categories. However, this advantage gradually erodes as k is increased, and at very large values of k, the propositional encoding is better overall. The apparent reason is that for large values of k the relational encoding tends to hypothesize rules based on accidental patterns of frequently-occuring but contentless words. For instance, the poor performance of the relational encoding at $k = 200$ is due largely to the rule *hostages(S)* $\leftarrow$ *of(S,B), to(S,C), after(C,B), near2(B,C)* which comprises the whole of FOIL's hypothesis for the *hostages* class. In contrast, with propositional encoding outputs an empty hypothesis—which is surely optimal given that the only remaining features are the words *of*, *in*, and *to*.[6]

To conclude this discussion, we note that frequency-based relation selection is somewhat at odds with more recent work in feature selection [Almuallim and Dietterich, 1991; Kira and Rendell, 1992; Skalak, 1994; John *et al.*, 1994]; we should perhaps take a moment to justify its use. The ten problems described above contain between 2304 and 3077 word relations; this, together with the fact that the underlying learning system is relatively slow,[7] precludes for efficiency reasons approaches that search for a combination of relevant features [Almuallim and Dietterich, 1991; Skalak, 1994; John *et al.*, 1994]. Also (and leaving aside for a moment the potential conceptual difficulties associated with extensions to a first order context) the large number of features and the consequently large average distance between instances made it seem unlikely, at least to us, that a RELIEF-like approach [Kira and Rendell, 1992] would be successful.

We did perform some preliminary experiments with relation selection methods that select word relations based on their correlation with the category. However, the performance of the method was disappointing. This is consistent with previous experimental results: Apté, Damerau and Weiss [1994] also found a frequency-based measure to be preferable to an entropy-based measure in selecting text categorization features for SWAP1, a propositional rule-induction system.

[6] In text retrieval tasks, it is common practise to also discard frequently-occuring "function words" such as determiners and propositions; were this done, then the relational encoding would likely be competitive even for large values of k. We retain function words to facilitate comparison with previous results on these datasets.

[7] Even with backtracking and introduction of "weak" literals disables, FOIL6 takes around 30 CPU seconds for each problem on a MIPS Irix/5.

7 MONOTONICITY CONSTRAINTS

FOIL6 includes an option (-n) that will prevent it from considering negative literals in a clause. We used this feature to learn "monotone" versions of all of the learning problems considered above.

There were several motivations for this experiment. Like relational selection, constraining the format of a hypothesis restricts the number of possible hypotheses and hence reduces the search space of the learner. It is therefore natural to consider using these constraints to replace or supplement relation selection. Also, for most natural categories, it seems unlikely *a priori* that the absence of a word would constitute evidence in favor of a category; similarly, the background relations we introduced are, when negated, either linguistically implausible as features (*e.g.*, "*not(near1(X,Y))*") or largely redundant (*e.g.*, *not(after(X,Y))* is equivalent to *after(Y,X)* when X and Y are both bound). Hence one might expect that introducing this constraint would improve generalization performance.

Another motivation is that the final result of text categorization learning is a classifier which will eventually be sent to some query system to retrieve additional documents. Many different query systems exist, and most support only a limited variety of retrieval operations. In particular, some query system (*e.g.*, retrieval systems based on matching regular expressions) do not allow retrieval of documents based on the absence of words. In such a context, the ability to learn a monotone classifier would be of value.

Finally, we note that many first order learning systems learn pure Horn clauses, without negated literals (*e.g.*, [Muggleton and Feng, 1992]).

We thus re-ran the experiments of Table 3, constraining FOIL6 to output monotone clauses. Table 4 summarizes the results. Contrary to our expectations, enforcing monotonicity did not improve generalization performance; rather, there is a general degradation of error rate, recall and precision. Otherwise the results of the experiment were qualitatively similar to those for unconstrained clauses. Frequency-based relation selection improves performance for small k, and thereafter improves precision at the expense of recall.

However, when only monotone clauses are allowed, the edge that the relational encoding enjoys is somewhat

Table 5: Comparing learners by error rate

errors	default rule	999 uncertain FOIL(k=1) prop	rel	FOIL(k=50) prop	rel	C4.5 LR=5	Prob	3+9997 random C4.5 LR=1	Prob
bonds	60	133	132	37	37	33.4	36.2	39.6	75.5
burma	93	38	38	66	65	46.9	50.8	59.6	100.7
neilson	87	568	569	43	47	48.8	32.1	55.5	52.0
tickertalk	40	41	43	41	42	40.0	40.6	40.5	56.8
boxoffice	42	133	133	29	30	24.5	25.2	31.8	40.1
budget	197	290	290	176	176	174.8	187.5	181.8	203.6
dukakis	107	181	134	106	99	102.4	108.0	108.9	122.1
hostages	228	380	383	206	212	215.7	187.3	242.6	224.3
ireland	117	165	165	109	109	97.9	98.1	114.2	118.5
quayle	133	160	90	101	107	83.5	115.6	74.4	136.6
average	110.4	208.9	197.7	91.4	92.4	86.8	88.1	94.9	113.0

stronger. In particular, the relational encoding is superior both when k is set to maximize recall ($k = 3$) and when k is set to minimize error rate ($k = 25$).

8 COMPARISON TO OTHER METHODS

A number of other learning methods have been applied to these benchmark problems, and it is instructive to see how they compare with FOIL.

Lewis and Catlett [1994] reported results obtained with a Bayesian probabilistic classifier, and also with an extension of C4.5 that allows the user to specify a *loss ratio*.[8] Table 5 gives a category-by-category comparison of our results to those of Lewis and Catlett, using the familiar metric of error rate. We show results for FOIL without any relational selection ($k = 1$), and with relation selection for $k = 50$, the value at which error rate is approximately minimized. We also repeat the results reported by Lewis and Catlett obtained by training on samples of 10,000 randomly chosen examples. These results are not directly comparable to the results obtained with an uncertainty sample, but give some insight as to what performance can be expected on this task given larger amounts of labeled data. (Recall that uncertainty sampling is intended for situations in which unlabeled data is cheap, but determining the category of an instance is expensive.)

To summarize briefly, FOIL without relation selection seems to have higher error rates, in general, than the other methods. (Indeed, without relation selection FOIL is inferior to the default rule, which simply classifies every example as negative.) With relation selection FOIL is roughly comparable to the other methods. Note that all of the methods in the table make some use of feature selection.

Table 6 gives a similar comparison of results for recall and precision. In each table entry, the upper number is recall, and the lower is precision. The values for FOIL with relation selection are given for $k = 5$, which strikes a relatively good balance between precision and recall. To summarize, FOIL without relation selection is complementary to the other methods; while both the probabilistic classifier and C4.5 have relatively high precision and low recall, FOIL without relation selection has relatively high recall and low precision. Adding relation selection with $k = 5$ yields somewhat better recall and much higher precision.

Another commonly used metric in information retrieval is Van Rijsbergen's E-measure [Van Rijsbergen, 1979, pages 168–176], defined as

$$E_\beta \equiv 1 - \frac{(\beta^2+1)PR}{\beta^2 P + R}$$

where P is precision, R is recall, and β is a parameter controlling the importance given to precision relative to recall. A value of $\beta = 1$ corresponds to equal weighting of precision and recall. Following Lewis and Gale [1994], we give in Table 7 values for F_β (defined as $F_\beta \equiv 1 - E_\beta$) at $\beta = 1$. This is a single measure of effectiveness in which precision and recall are given equal importance, and higher scores correspond to better classifiers.

To summarize the F-measure results, FOIL without relation selection does relatively poorly compared to the other systems. With relation selection, however, FOIL does relatively well; in fact with $k = 5$ FOIL outperforms both C4.5 and the probabilistic classifier with either encoding. The relational encoding is again slightly better than the propositional encoding according to this measure.

[8] A *loss ratio* indicates the ratio of the cost of a false negative to the cost of a false positive; the goal of learning is to minimize misclassification cost on unseen data. In machine learning. In training C4.5 on uncertainty samples, better results were obtained with a loss ratio of five.

Table 6: Comparing learners by recall and precision

		999 uncertain						3+9997 random	
		FOIL(k=1)		FOIL(k=5)		C4.5	Prob	C4.5	Prob
		prop	rel	prop	rel	LR=5		LR=1	
bonds	*recall*	0.5167	0.5333	0.5500	0.5667	0.5033	0.5283	0.4183	0.3017
	precision	0.2296	0.2353	0.6111	0.6939	0.8935	0.8005	0.8423	0.3501
burma	*recall*	0.6344	0.6344	0.5054	0.5591	0.5280	0.5183	0.4129	0.6880
	precision	0.9365	0.9365	0.8246	0.8667	0.9424	0.8893	0.8848	0.3122
neilson	*recall*	0.6322	0.6207	0.7586	0.7356	0.5081	0.7448	0.4943	0.5241
	precision	0.9310	0.9150	0.7253	0.6957	0.8805	0.8675	0.7890	0.8114
tickertalk	*recall*	0.0000	0.2500	0.5000	0.5000	0.0000	0.1750	0.0750	0.3750
	precision	0.0000	0.2000	0.2222	0.2222	1.0000	0.3500	0.2727	0.7580
boxoffice	*recall*	0.7857	0.7857	0.7381	0.7381	0.5476	0.7000	0.3619	0.3024
	precision	0.2102	0.2102	0.3163	0.3163	0.8070	0.7000	0.7525	0.5404
budget	*recall*	0.3147	0.3096	0.3807	0.3706	0.1665	0.1954	0.3670	0.9700
	precision	0.2857	0.2837	0.4545	0.5935	0.7558	0.5704	0.5587	0.4263
dukakis	*recall*	0.3925	0.4019	0.3925	0.4206	0.1776	0.7940	0.6252	0.5510
	precision	0.2658	0.3805	0.5385	0.5488	0.5689	0.4722	0.4930	0.2193
hostages	*recall*	0.3158	0.2982	0.3991	0.3991	0.1860	0.3746	0.2794	0.1272
	precision	0.2432	0.2337	0.3333	0.3434	0.5848	0.6564	0.4486	0.5341
ireland	*recall*	0.3761	0.3675	0.4103	0.4188	0.2043	0.3000	0.1299	0.1051
	precision	0.3235	0.3209	0.5926	0.4949	0.8328	0.6842	0.5507	0.4713
quayle	*recall*	0.6842	0.7218	0.7143	0.7143	0.5474	0.2737	0.7587	0.1023
	precision	0.4354	0.6443	0.6835	0.6738	0.7575	0.6570	0.7046	0.4416
average	*recall*	0.4652	0.4698	0.4899	0.4973	0.3367	0.3732	0.3855	0.1721
	precision	0.3023	0.3537	0.5302	0.5449	0.8023	0.6648	0.6297	0.4182

9 CONCLUSIONS

In this paper, we have evaluated first order learning techniques (and more specifically, FOIL6) on text categorization problems. Such first order learning techniques are of interest here because, unlike purely propositional methods, they can learn classifiers that depend on word order.

To summarize one set of results, we found that on a set of ten benchmark problems FOIL using a propositional encoding obtains (generally speaking) higher rates of recall than other propositional systems, but much lower precision, and much higher error rates. FOIL's performance can be improved dramatically by relation selection, a natural first order analog to feature selection. Low levels of relation selection raise recall modestly, and raise precision substantially, leading to a version of FOIL that outperforms existing systems according to the widely-used F_β metric. Higher levels of relation selection lead to a version of FOIL that is competitive with existing propositional systems with respect to error rate.

We also showed that FOIL often forms classifiers with lower error rates and higher rates of precision and recall with a relational encoding than with a propositional encoding. These differences are small on average; in fact, inspection of the hypotheses that FOIL forms shows that FOIL only infrequently makes use of the relational aspects of the encoding. However, the reduction in error rate is statistically significant, and is quite substantial for some categories.

The interaction between relation selection and use of a relational encoding is rather complex. Under certain circumstances, relation selection can eliminate the advantage obtained by adopting a relational encoding; if taken to extremes, relation selection can even cause the relational encoding to be inferior. However, using a relational encoding is generally advantageous in a number of important circumstances: notably when no relation selection is used, when relation selection is used to maximize recall, when relation selection is used to maximize F_β, and when relation selection is used to minimize error rate for monotone clauses.

Acknowledgements

The author is extremely grateful to David Lewis, who provided help and advice at almost every stage of this research. In particular, David provided the data used in the experiments, made suggestions concerning a draft of the paper, and also contributed materially to the ideas in the paper with a series of helpful discussions. Haym Hirsh and two anonymous reviewers also provided useful feedback on the presentation.

Table 7: Comparing learners by F-measure ($\beta = 1$)

$F_{\beta=1}$	999 uncertain						3+9997 random		full corpus
	FOIL(k=1)		FOIL(k=5)		C4.5	Prob	C4.5	Prob	Prob
	prop	rel	prop	rel	LR=5		LR=1		
bonds	0.318	0.327	0.579	0.624	0.644	0.637	0.559	0.324	0.509
burma	0.756	0.756	0.627	0.680	0.677	0.655	0.563	0.113	0.464
nielsons	0.162	0.159	0.742	0.715	0.644	0.801	0.608	0.637	0.741
tickertalk	0.000	0.044	0.082	0.082	0.000	0.033	0.015	0.050	0.047
boxoffice	0.332	0.332	0.443	0.443	0.652	0.700	0.489	0.388	0.647
budget	0.299	0.296	0.414	0.456	0.273	0.291	0.443	0.158	0.235
dukakis	0.317	0.391	0.454	0.476	0.271	0.136	0.551	0.088	0.163
hostages	0.275	0.262	0.363	0.369	0.282	0.477	0.344	0.205	0.498
ireland	0.348	0.343	0.485	0.454	0.328	0.417	0.210	0.172	0.288
quayle	0.532	0.681	0.699	0.693	0.636	0.386	0.731	0.166	0.493
average	0.3339	0.3591	0.4888	0.4992	0.4407	0.4533	0.4513	0.2301	0.4085

References

(Almuallim and Dietterich, 1991) H. Almuallim and T. Dietterich. Learning with many irrelevant features. In *Proceedings of the Tenth National Conference on Artificial Intelligence*. AAAI Press, 1991.

(Apté *et al.*, 1994) Chidanand Apté, Fred Damerau, and Sholom M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994.

(Cameron-Jones and Quinlan, 1993) R. M. Cameron-Jones and J. R. Quinlan. First-order learning, zeroth order data. In *Proceedings of the Sixth Australian Joint Conference on Artificial Intelligence*. World Scientific, 1993.

(Harter, 1986) Stepher P. Harter. *Online Information Retrieval*. Academic Press, San Diego, 1986.

(John *et al.*, 1994) George John, Ron Kohavi, , and Karl Pfeger. Irrelevant features and the subset selection problem. In *Machine Learning: Proceedings of the Eleventh Annual Conference*, New Brunswick, New Jersey, 1994. Morgan Kaufmann.

(Kira and Rendell, 1992) Kinji Kira and Larry Rendell. A practical approach to feature selection. In *Proceedings of the Tenth International Conference on Machine Learning*, Amherst, Massachusetts, 1992. Morgan Kaufmann.

(Lewis and Catlett, 1994) David Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning: Proceedings of the Eleventh Annual Conference*, New Brunswick, New Jersey, 1994. Morgan Kaufmann.

(Lewis and Gale, 1994) David Lewis and William Gale. Training text classifiers by uncertainty sampling. In *Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.

(Muggleton and Feng, 1992) Stephen Muggleton and Cao Feng. Efficient induction of logic programs. In *Inductive Logic Programming*. Academic Press, 1992.

(Quinlan and Cameron-Jones, 1993) J. R. Quinlan and R. M. Cameron-Jones. FOIL: A midterm report. In Pavel B. Brazdil, editor, *Machine Learning: ECML-93*, Vienna, Austria, 1993. Springer-Verlag. Lecture notes in Computer Science # 667.

(Quinlan, 1990) J. Ross Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3), 1990.

(Quinlan, 1991) J. Ross Quinlan. Determinate literals in inductive logic programming. In *Proceedings of the Eighth International Workshop on Machine Learning*, Ithaca, New York, 1991. Morgan Kaufmann.

(Quinlan, 1994) J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann, 1994.

(Skalak, 1994) D. B. Skalak. Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *Machine Learning: Proceedings of the Eleventh Annual Conference*, New Brunswick, New Jersey, 1994. Morgan Kaufmann.

(Van Rijsbergen, 1979) C. J. Van Rijsbergen. *Information Retrieval*. Butterworth, London, second edition, 1979.