Classifying Documents Without Labels

Daniel Barbará Carlotta Domeniconi Ning Kang
Information and Software Engineering Department
George Mason University
{dbarbara,cdomenic,nkang}@gmu.edu

September 18, 2003

Abstract
Automatic classification of documents is an important area of research with many applications in the fields of document searching, forensics and others. Methods to perform classification of text rely on the existence of a sample of documents whose class labels are known. However, in many situations, obtaining this sample may not be an easy (or even possible) task. Consider for instance, a set of documents that is returned as a result of a query. If we want to separate the documents that are truly relevant to the query from those that are not, it is unlikely that we will have at hand labelled documents to train classification models to perform this task. In this paper we focus on the classification of an unlabelled set of documents into two classes: relevant and irrelevant, given a topic of interest. By dividing the set of documents into buckets (for instance, answers returned by different search engines), and using association rule mining to find common sets of words among the buckets, we can efficiently obtain a sample of documents that has a large percentage of relevant ones. (I.e., a high "purity".) This sample can be used to train models to classify the entire set of documents. We try several methods of classification to separate the documents, including Two-class SVM, for which we develop a heuristic to identify a small sample of negative examples. We prove, via experimentation, that our method is capable of accurately classify a set of documents into relevant and irrelevant classes.

Keywords: Document classification, frequent itemsets, support vector machines, partially supervised classification.

1 Introduction

In information retrieval, such as content-based image retrieval, web-page classification, or document retrieval, we face an asymmetry between positive and negative examples [22, 4]. Suppose, for example, we submit a query to multiple search engines. Each engine retrieves a collection of documents in response to our query. Such collections include, in general, both relevant and irrelevant documents. Suppose we want to discriminate the relevant documents from the irrelevant ones. The set of all relevant documents in all retrieved collections represent a sample of the positive class, drawn from an underlying unknown distribution. On the other hand, the irrelevant documents may come from an unknown number of different "negative" classes. In general, we cannot approximate the distributions of the negative classes, as we may have too few representatives for each of them. Hence, we are facing a problem with an unknown number of classes, with the user interested in only one of them.

Modelling the above problem as a two-class problem, may impose misleading requirements, that can yield poor results. For example, lets assume for a moment that the positive and negative labels are available, and all negatives are "alike". We can apply Fisher discriminant analysis, and therefore project the data onto a subspace in which the ratio of the between-class scatter over the within-class scatter is maximized [7]. By doing so we require that the negative examples, as well as the positives, shall cluster in the discriminating subspace. This is an unnecessary requirement that can damage the accuracy of the resulting model. In fact, most likely negative examples belong to different classes, and the few examples available per class cannot be representative of the underlying distributions.

As such, a two-class model may not reflect the actual nature of the data. We are definitely better off focusing on the class of interest, as positive examples in this scenario have a more compact support, that reflects the correlations among their feature values.

Moreover, more often than not, the class labels of the data are unknown, either because the data is too large for an expert to label it, or because no such expert exists. In this work we eliminate the assumption of having even partially labelled data.

In this work we focus on document retrieval, and develop a technique to mining relevant text from unlabelled documents. Specifically, our objective is to identify a sample of positive documents, representative of the underlying class distribution. The scenario of a query submitted to multiple search engines will serve as running example throughout the paper, although the technique can be applied to a variety of scenarios and data. Our approach reflects the asymmetry between positive and negative data, and does not make any particular and unnecessary assumption on the negative examples. After identifying the sample of positive examples, we employ several classification methods to separate the documents in the data set. Specifically, we used One-class SVM, Two-class SVM, and the techniques of the system LPU [16] for the classification part. In order to be able to apply Two-class SVM, we developed a simple heuristic that finds a threshold capable of identifying a few negative examples from the data set, given the characteristics of the positive sample obtained.

The rest of the paper is organized as follows. Section 2 covers the related work. Section 3 presents the DocMine algorithm employed to identify the positive examples. Section 4 explains the details of the document classification step, including the way of obtaining a sample of negative examples. Section 5 shows the experimental results. Finally in Section 6 we present conclusions and future work.

2 Related Work

In [9] the authors discuss a hierarchical document clustering approach using frequent set of words. Their objective is to construct a hierarchy of documents for browsing at increasing levels of specificity of topics. The algorithm starts constructing, for each frequent itemset (i.e., set of words) in the whole document set, an initial cluster of all the documents that contain this itemset. Then, it proceeds making the clusters disjoint. To this extent a measure of goodness of a cluster for a document is used: a cluster is "good" for a document if there are many frequent items (with respect to the whole document set) in the document that are also frequent within the cluster. Hence, each document is removed from all the initial clusters but the one that maximizes this measure of goodness. This stage gives a disjoint set of clusters, that is used to construct a tree of groups of documents. The tree is built bottom-up by choosing for each cluster C_k at a given level the unique parent (cluster) with the largest similarity score. By merging all documents in C_k into a single conceptual document, the similarity score between C_k and its candidate parents is measured using a criterion similar to the measure of goodness of a cluster for a document.

[2] considers the problem of enhancing the performance of a learning algorithm allowing a set of unlabelled data augment a small set of labelled examples. The driving application is the classification of Web pages. Although similar to our scenario, the technique depends on the existence of labelled data to begin with. (This technique could be readily used after ours to learn a good classifier.)

Similarly, the work in [15] makes use of unlabelled documents to construct classifiers with enhanced performance. It is assumed that a set of (labelled) positive documents is given, and a (larger) set of unlabelled documents is available. The technique initially considers the unlabelled data as negatives. It then applies an iterative naive Bayes classifier, combined with the EM algorithm to re-estimate class posterior probabilities. Positive documents (called "spy") are introduced in the set of unlabeled data to estimate which documents are most likely the actual negatives.

The authors in [11] exploit semantic similarity between terms and documents in an unsupervised fashion. Documents that share terms that are different, but semantically related, will be considered as unrelated when text documents are represented as a *bag of words*. The purpose of the work in [11] is to overcome this limitation by learning a *semantic proximity matrix* [6] from a given corpus of documents by taking into consideration high order correlations. Two methods (both yielding to the definition of a kernel function) are discussed. In particular, in one model, documents with highly correlated words are considered as having similar content. Similarly, words contained in correlated documents are viewed as semantically related.

The work we present here serves a similar purpose, by using association rule mining. The search for frequent set of words, in a segmented corpus of examples, allows the selection of documents that share cooccurrent terms, thereby considered to have a similar content. Our method views documents as *bags of words*, for the purpose of mining frequent itemsets, and, at the same time, views itemsets (i.e., set of words) as *bags of documents* (i.e., the documents that contain them), for the purpose of retrieving the texts that have such words expressed within. An analog duality is also observed in Kandola et al. (2002).

3 The DocMine Algorithm

Given a document, it is possible to associate with it a *bag of words* (Joachims, 1998; Dumais et al., 1997; Leopold & Kindermann, 2002). Specifically, we represent a document as a binary vector $\mathbf{d} \in \Re^n$, in which each entry records if a particular word stem occurs in the text. The dimensionality n of $\mathbf{d}$ is determined by the number of different terms in the corpus of documents (size of the *dictionary*), and each entry is indexed by a specific term.

Going back to our example, suppose we submit a query to s different search engines. We obtain s

collections, or *buckets*, of documents

(3.1) $$B_j = \{\mathbf{d}_i\}_j, \quad j = 1, \ldots, s$$

While many documents retrieved by a specific search engine (a bad one) might be irrelevant, the relevant ones are expected to be more frequent in the majority of buckets. In addition, since we can assume that positive documents are drawn from a single underlying distribution, a compact support unifies them across all buckets. On the other hand, the negatives manifest a large variation. We make use of these characteristics to develop a technique that discriminates relevant documents from the irrelevant ones. In details, we proceed as follows.

We mine each bucket B_j to find the frequent itemsets that satisfy a given support level. Each resulting itemset is a set of words. The result of this process is a collection of sets of itemsets, one set for each bucket:

(3.2) $$F_j = \{W_i | W_i \ is \ a \ frequent \ itemset \ in \ bucket \ j\}$$

for $j = 1, \ldots, s$, where it is possible that $F_j = \emptyset$, for some j. Now we compute all itemsets that are frequent in m buckets

(3.3) $$I_m = \{W_i | W_i \in F_{j_1} \cap F_{j_2} \cap \ldots \cap F_{j_m}\}$$

for distinct $j_1, \ldots, j_m$. In general $m = \lfloor s/2 \rfloor + 1$. In our experiments we set $m = s$ since we consider a limited number of buckets ($s = 5$), driven by the number of available documents per topic. We wish now to retrieve the documents that *support* the itemsets that are frequent in m buckets. Then, for each $W_i \in I_m$, we select, in each of the m buckets that contain W_i as frequent itemset, the documents that has W_i expressed within. The resulting collection of documents P represent the presumed positive documents, relevant to our query.

The algorithm, which we call DocMine (**Doc**ument **Min**ing)([1] contains a description of DocMine and preliminary results on the purity of the sample of positives obtained) is summarized in the following. The algorithm takes as input the s buckets of documents, and the minimum support (Sup_{min}) for the computation of frequent itemsets.

Algorithm 3.1. (DocMine)

```
1. Input: $s$ buckets of documents $B_j = \{\mathbf{d}_i\}_j$, $j = 1, \ldots, s$, $Sup_{min}, m$.
2. Compute frequent itemsets in each bucket $B_j$:
   $F_j = \{W_i | W_i \ is \ a \ frequent \ itemset \ in \ bucket \ j\}$, $j = 1 \ldots, s$
3. Compute all itemsets that are frequent in $m$ buckets:
   $I_m = \{W_i | W_i \in F_{j_1} \cap F_{j_2} \cap \ldots \cap F_{j_m}\}$
4. Set $P = \emptyset$.
5. for each $W_i \in I_m$
     • for each $l = 1, \ldots, m$ such that $W_i \in \cap_{l=1}^{m} F_{j_l}$
         – for each $\mathbf{d} \in B_{j_l}$
             ∗ if $\mathbf{d}$ contains $W_i$
                 · $P = P \cup \{\mathbf{d}\}$
6. Output: the set $P$ (presumed positive documents)
```

Figure 1 shows a functional diagram of the algorithm.

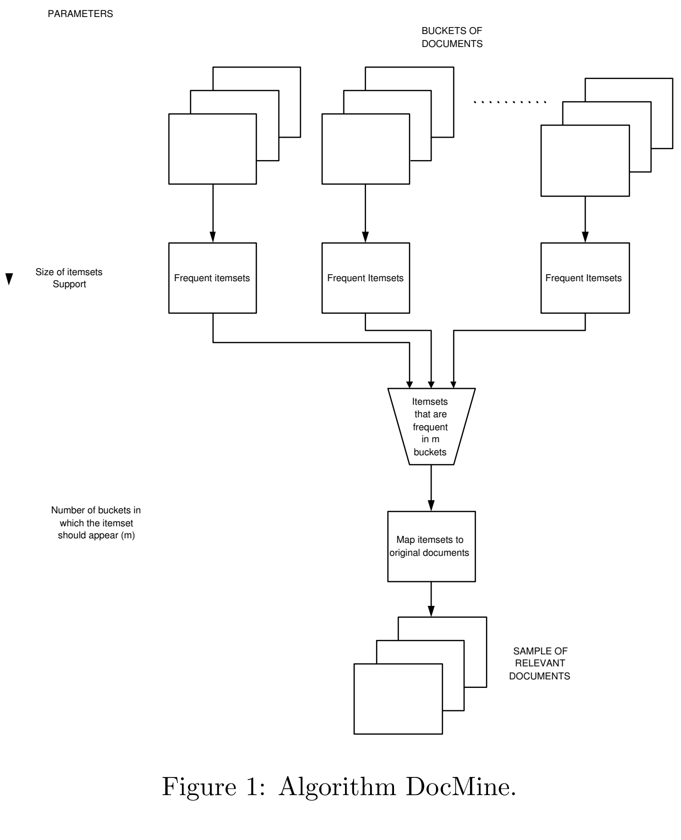
PARAMETERS
BUCKETS OF DOCUMENTS
....
Size of itemsets Support
Frequent itemsets
Frequent Itemsets
Frequent Itemsets
Itemsets that are frequent in m buckets
Number of buckets in which the itemset should appear (m)
Map itemsets to original documents
SAMPLE OF RELEVANT DOCUMENTS

Figure 1: Algorithm DocMine.

It is important to remark here that the DocMine algorithm can be tuned to ignore small size itemsets (i.e., sets of less than t items). The reason for this is that some words (or even small combinations of words) may be common across documents of many different topics (they would not discriminate). Our experience tells us that, for instance, combinations of two frequent words (itemsets of size two) are not sufficient to discriminate among different topics.

4 Document Classification

Given the set P of (presumed) positives computed by the DocMine algorithm, we can proceed towards the definition, and training of a classifier. We wish to make use not only of P, but also of the original whole set of documents, with the objective of bootstrapping negative documents as well.

We need to find a suitable representation of the documents for the purpose of classifying them. The set P contains the documents that support the collection of frequent itemsets I_m. Such itemsets contain the words of the dictionary that allowed the discrimination of P, and discriminant features is what is needed for classification. Hence, we can consider the union of the words (i.e., frequent items) in I_m, and represent a document in P as a *bag of frequent itemsets*. That is: each entry of $\mathbf{d} \in P$ is indexed by a word in I_m. The actual value of the entry is the frequency of the corresponing word in the document. This gives a suitable representation since it is *compact*, and captures the distinctive characteristic of the documents in P. In this respect, the DocMine algorithm can be seen as a *dimensionality reduction* and *feature extraction* procedure for the target class (or topic of interest). The dimensionality of $\mathbf{d}$ is determined by the number of frequent items in I_m. In our experiments it varies from 40 to 60.

To select a set of likely negative documents from the buckets we derive a data-driven threshold value. We make use of the fact that the average number of (distinct) frequent items is expected to be larger in positive documents rather than in negatives. Let P be the set of positives given in output by the DocMine algorithm. We denote with A the set of all the documents in the buckets. Note that all the documents share the same *bag of frequent itemsets* representation. For each document $\mathbf{d}$ in P, we count the number $N_\mathbf{d}$ of non-zero entries (i.e., number of distinct frequent items in $\mathbf{d}$). We then consider the inverse $1/N_\mathbf{d}$, and compute the average $Ave_P = 1/|P| \sum_P (1/N_\mathbf{d})$ for all documents in P. We proceed similarly for the documents in A, and obtain the corresponding average value Ave_A. Since we are averaging over the inverse of numbers of non zero entries in documents, Ave_A will be larger than Ave_P. We derive an estimate of the expected average value Ave_N for the negatives from the equation: $Ave_A = (Ave_P + Ave_N)/2$. (In reality this equation may not hold, as the distribution of the positives and negatives is not likely to be the same.) Thus we set $Ave_N = 2Ave_A - Ave_P$. We then consider as negatives the documents $\mathbf{d} \in (A - P)$ satisfying $1/N_\mathbf{d} > Ave_N$.

5 Experimental Results

To test the feasibility of our approach we use the Reuters-21578 text categorization collection [13], omitting empty documents and those without labels. Common and rare words are removed, and the vocabulary is stemmed with the Porter Stemmer [17]. After stemming, the vocabulary size is 12113. We also combine the result provided by the DocMine algorithm with several classification approaches, and compare the obtained accuracy levels.

5.1 Results with the DocMine algorithm In our experiments, we consider five buckets of documents ($s = 5$), and vary the percentage R of relevant documents (i.e., concerning the topic of interest) in each bucket from 50% to 80%. As topics of interest, we select the topics with the largest number of documents available in the data set. Once we have identified a topic, the non relevant documents are randomly selected from the remaining topics. We observe that some documents in the Reuters data have multiple topics associated (e.g., *grain* and *crops*). In our experiments, a document is considered positive if it has the topic of interest among its associated topics. For each topic examined, we test three different values of the minimum support (10%, 5%, 3%).

We have also investigated different threshold values (from 2 to 5) for the cardinality of the frequent itemsets ($|W_i|$). Only frequent itemsets of size above (or equal to) the threshold are considered for the retrieval of relevant documents. The rationale beyond this test is that if an item is too common across different documents, then it would have little discriminating power. The setting of a proper threshold for $|W_i|$ allows to discard frequently used words (not removed during preprocessing) that are not discriminating. Our experiments show that threshold values of 4 or 5 (depending on the value of the minimum support) give good results.

In the following tables we report, for each topic and each value of R, the number of (retrieved) documents in P ($|P|$), the number of positive (relevant) documents in P ($|P^+|$), the percentage of positive documents in P ($\%|P^+|$) –precision –, and the percentage of positive documents retrieved by P (r) – recall–. When a dash "-" is reported, instead of a numerical value, it means that there are no frequent itemsets, of corresponding size, common to the five buckets. Each caption has (in parenthesis) the total number of positive documents versus the total number of documents in the five buckets.

We observe that, although we report the recall values for all the experiments, the most important measure of the effectiveness of this step is the precision obtained in the sample. The reason is that precision quantifies the "purity" of the sample, whose documents we intend to label as relevant to the topic at hand.

We have considered three different topics in our experiments: *earn*, *acq*, and *grain*. The data set contains 3776 documents of topic *earn*, 2210 of topic *acq*, and 570 of topic *grain*. In each experiment, we distribute all the available positives among the buckets, and adjust the number of negatives accordingly to the R value considered.

Tables 1-4 show the results for the topic *earn*. Figures 2-4 plot the precision values for the topic *earn*, for increasing threshold t on the itemset size $|W_i|$. Each line corresponds to a value of R (percentage of positive documents in each bucket). The plots show that, in each case, the setting of $t = 5$ allows the achievement of a precision value very close to 1. For larger support values (5% and 10%), $t = 4$ suffices for the selection of an almost "pure" sample of documents. Even in the adverse condition of 50% of irrelevent documents in the buckets, the DocMine algorithm is able to achieve a very high precision. Similar results are shown in Tables 5-8 and Figures 5-7 for the topic *acq*.

Tables 9-12 show the results for the topic *grain*. Since in this case we have available a small number of positives (570), we have considered only the two larger support values (10% and 5%) ($Sup_{min} = 3\%$ generated an intractable large number of frequent itemsets for our current implementation of the algorithm). Moreover, for larger values of $|W_i|$, often the five buckets had no frequent itemsets in common, due to the limited number of positives available. In figure 8, we summarize the precision values as a function of t, corresponding to $Sup_{min} = 5\%$. For $R = 70\%$ and $R = 80\%$, and $t = 4$, a precision value above 90% is achieved (97% and 92%, respectively) also in this case, for which a limited sample of positives is available.

5.2 Classification Results We compare the classification results of a number of different approaches:

1. DocMine and One-class SVM [19, 20];
2. DocMine and Two-class SVM [21, 5] (with bootstrapped negatives as described in Section 4)
3. DocMine and (SPY, SVM) [15, 16];
4. DocMine and (SPY, EM) [15, 16];
5. DocMine and (Rocchio, SVM) [18, 16];
6. DocMine and (Rocchio, EM) [18, 16];

LIBSVM [3] is used to build the SVM classifier in 1 and 2 above. We used a Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}) = e^{-\gamma\|\mathbf{x}_i - \mathbf{x}\|^2}$, with γ set to the inverse of the number of input features. We test the sensitivity of the classification accuracy for different values of the soft-margin parameter.

In 3, 4, 5, and 6 above, the techniques SPY, SVM, Rocchio, and EM were tested using the classification system LPU [16]. The method SPY or Rocchio is used to identifying a set of negative documents from the unlabeled set (documents in our buckets). SVM or EM is then used to build and select a classifier [14]. We emphasize that all the techniques in [14, 16] assume the existence of a set of (labeled) positive documents. Thus, we first apply our DocMine algorithm to obtain such a collection of positives.

Classification results for the topics *earn* and *acq*, for different values of Sup_{min} and minimum cardinality t, are shown in Figures 9-14. Accuracy is tested on the whole set of documents in our buckets. Labels of documents are used only to test accuracy, and never during training. The top plot of each figure shows the precision values of a one-class SVM trained with the set P computed by DocMine. The results for different tested values of the soft margin parameter "nu" are reported. The middle plot of each figure shows the precision values of a two-class SVM trained with the set P computed by DocMine, and the set of negatives obtained via the technique described in Section 4. Here also, we show the results for different tested values of the soft margin parameter "nu". The two-class SVM, in general, is able to achieve higher levels of accuracy, indicating that our technique to bootstrapping negatives has the potential of enhancing classification performance. For a variety of combinations of values of Sup_{min} and t, (Middle plots of Figures 10, 11, 12), the two-class SVM gives excellent performance, especially in adverse conditions where the percentage of negatives in the buckets is high. Robustness across different "nu" values is also high. The bottom plots show the precision values of different combinations of techniques ((SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM)) applied using the set P computed by DocMine. As also pointed out in [16], the performance achieved by these methods at convergence may often be considerably worst than the one achieved at an intermediate step. Here we report the best precision value (hand picked) achieved by any of the steps (intermediate or at convergence). For the topic *earn* the combination (Rocchio, EM) gives the best performance across all conditions (Bottom plots of

Figures 9-12). For the topic *acq* the combination (SPY, SVM) gives the best performance (a monotonically increasing accuracy for larger values of R). As expected, different methods may be well suited for differet data sets. Nevertheless, our techniques to bootstrap positives and negatives, combined with two-class SVMs, show a robust behavior across the conditions and data tested (for values of "nu" in the range 0.15-0.20). This is a very promising and encouraging result for the construction of accurate classifiers without using *any* labels for training.

Table 1: Results for topic *earn*. $R = 50\%$ (3776/7552).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	5323	2824	0.53	0.74
	≥ 3	2538	2204	0.87	0.58
	≥ 4	1848	1848	1.00	0.49
	≥ 5	1103	1103	1.00	0.29
5%	≥ 2	6441	3012	0.47	0.80
	≥ 3	4653	2597	0.56	0.69
	≥ 4	1972	1913	0.97	0.51
	≥ 5	1284	1284	1.00	0.34
3%	≥ 2	7246	3597	0.50	0.95
	≥ 3	5789	2943	0.51	0.78
	≥ 4	3671	2408	0.66	0.64
	≥ 5	1642	1628	0.99	0.43

Table 2: Results for topic *earn*. $R = 60\%$ (3776/6294).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	4453	2932	0.66	0.78
	≥ 3	2725	2250	0.83	0.60
	≥ 4	1842	1841	0.99	0.49
	≥ 5	1403	1403	1.00	0.37
5%	≥ 2	5684	3507	0.62	0.93
	≥ 3	3985	2668	0.67	0.71
	≥ 4	2045	1999	0.98	0.53
	≥ 5	1381	1376	0.99	0.36
3%	≥ 2	5859	3561	0.61	0.94
	≥ 3	4636	2928	0.63	0.78
	≥ 4	3311	2490	0.75	0.66
	≥ 5	1879	1875	0.99	0.50

6 Conclusions

We have introduced a new algorithm, based on association rule mining, to select a representative sample of positive examples from a given set of unlabelled docu-

Table 3: Results for topic *earn*. $R = 70\%$ (3776/5394).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	3592	2940	0.82	0.78
	≥ 3	2515	2274	0.90	0.60
	≥ 4	1849	1842	0.99	0.49
	≥ 5	1674	1674	1.00	0.44
5%	≥ 2	4784	3467	0.72	0.92
	≥ 3	3253	2747	0.84	0.73
	≥ 4	2027	1989	0.98	0.53
	≥ 5	1644	1642	0.99	0.43
3%	≥ 2	4982	3555	0.71	0.94
	≥ 3	4422	3447	0.78	0.91
	≥ 4	3550	3079	0.87	0.81
	≥ 5	1807	1803	0.99	0.48

Table 4: Results for topic *earn*. $R = 80\%$ (3776/4720).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	3192	2810	0.88	0.74
	≥ 3	2398	2279	0.95	0.60
	≥ 4	1394	1393	0.99	0.37
	≥ 5	1205	1205	1.00	0.32
5%	≥ 2	4151	3483	0.84	0.92
	≥ 3	3003	2763	0.92	0.73
	≥ 4	2126	2111	0.99	0.56
	≥ 5	1589	1587	0.99	0.42
3%	≥ 2	4294	3493	0.81	0.93
	≥ 3	3854	3275	0.85	0.87
	≥ 4	3059	2780	0.91	0.74
	≥ 5	2447	2377	0.97	0.63

Table 5: Results for topic *acq*. $R = 50\%$ (2210/4420).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	3413	1896	0.56	0.86
	≥ 3	2015	1414	0.70	0.64
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-
5%	≥ 2	4014	2157	0.54	0.98
	≥ 3	3131	818	0.58	0.37
	≥ 4	1654	1072	0.65	0.49
	≥ 5	-	-	-	-
3%	≥ 2	3756	1971	0.52	0.89
	≥ 3	2495	1510	0.61	0.68
	≥ 4	1383	892	0.64	0.40
	≥ 5	-	-	-	-

Table 6: Results for topic *acq*. $R = 60\%$ (2210/3685).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	2728	1927	0.71	0.89
	≥ 3	1874	1456	0.78	0.66
	≥ 4	400	369	0.92	0.17
	≥ 5	-	-	-	-
5%	≥ 2	3345	2173	0.65	0.98
	≥ 3	2598	1823	0.70	0.82
	≥ 4	1582	1312	0.83	0.59
	≥ 5	451	415	0.92	0.19
3%	≥ 2	3609	2193	0.61	0.99
	≥ 3	3196	2002	0.63	0.91
	≥ 4	2321	1593	0.69	0.72
	≥ 5	1153	1041	0.90	0.47

Table 7: Results for topic *acq*. $R = 70\%$ (2210/3160).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	2515	1993	0.79	0.90
	≥ 3	1753	1475	0.84	0.67
	≥ 4	680	627	0.92	0.28
	≥ 5	-	-	-	-
5%	≥ 2	2964	2186	0.74	0.99
	≥ 3	2359	1879	0.80	0.85
	≥ 4	1560	1397	0.90	0.63
	≥ 5	709	665	0.94	0.30
3%	≥ 2	3105	2194	0.71	0.99
	≥ 3	2768	2074	0.75	0.94
	≥ 4	1793	1380	0.77	0.62
	≥ 5	1052	950	0.90	0.43

Table 8: Results for topic *acq*. $R = 80\%$ (2210/2763).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	2370	2067	0.87	0.94
	≥ 3	1712	1554	0.91	0.70
	≥ 4	850	815	0.96	0.37
	≥ 5	-	-	-	-
5%	≥ 2	2579	2187	0.85	0.99
	≥ 3	2187	1912	0.87	0.87
	≥ 4	1553	1456	0.94	0.66
	≥ 5	861	830	0.96	0.38
3%	≥ 2	2705	2186	0.81	0.99
	≥ 3	2275	1931	0.85	0.87
	≥ 4	1999	1719	0.86	0.78
	≥ 5	940	902	0.96	0.41

Table 9: Results for topic *grain*. $R = 50\%$ (570/1140).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	895	517	0.58	0.91
	≥ 3	-	-	-	-
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-
5%	≥ 2	1016	554	0.55	0.97
	≥ 3	659	399	0.61	0.70
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-

Table 10: Results for topic *grain*. $R = 60\%$ (570/950).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	687	507	0.74	0.89
	≥ 3	-	-	-	-
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-
5%	≥ 2	831	555	0.67	0.97
	≥ 3	515	421	0.82	0.74
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-

Table 11: Results for topic *grain*. $R = 70\%$ (570/814).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	688	522	0.76	0.92
	≥ 3	110	110	1.00	0.19
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-
5%	≥ 2	762	561	0.74	0.98
	≥ 3	626	503	0.80	0.88
	≥ 4	174	168	0.97	0.29
	≥ 5	-	-	-	-

Table 12: Results for topic *grain*. $R = 80\%$ (570/713).

Sup_{min}	$\|W_i\|$	$\|P\|$	$\|P^+\|$	$\%\|P^+\|$	r
10%	≥ 2	627	529	0.84	0.93
	≥ 3	237	232	0.98	0.41
	≥ 4	-	-	-	-
	≥ 5	-	-	-	-
5%	≥ 2	677	562	0.83	0.99
	≥ 3	579	510	0.88	0.89
	≥ 4	296	273	0.92	0.48
	≥ 5	-	-	-	-

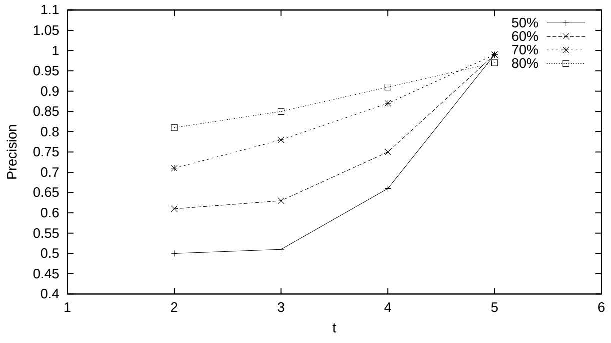

Figure 2: Precision values for topic *earn* and $Sup_{min} = 3\%$. The x-axis is the minimum cardinality of common itemsets (t).

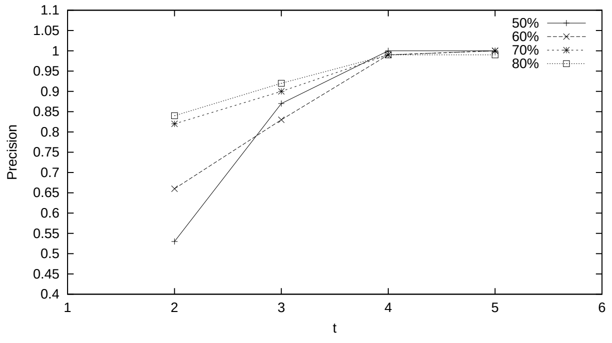

Figure 4: Precision values for topic *earn* and $Sup_{min} = 10\%$.The x-axis is the minimum cardinality of common itemsets (t).

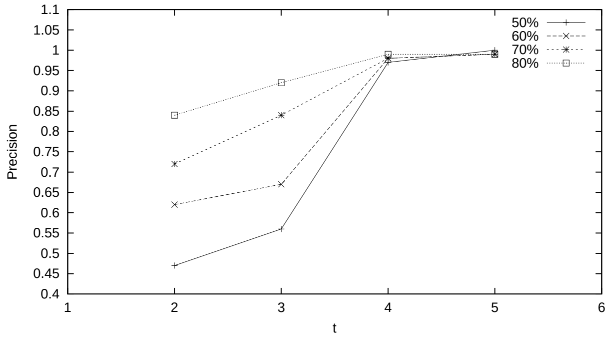

Figure 3: Precision values for topic *earn* and $Sup_{min} = 5\%$. The x-axis is the minimum cardinality of common itemsets (t).

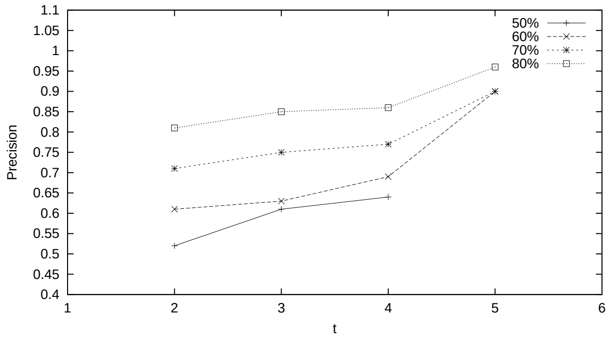

Figure 5: Precision values for topic *acq* and $Sup_{min} = 3\%$. The x-axis is the minimum cardinality of common itemsets (t).

ments. Our experiments show that our method is capable of selecting sets of documents with precision above 90% in most cases, when frequent itemsets of cardinality 4 or 5 are considered. We emphasize that, in all cases, the precision tends to reach high levels, as the cardinality of the common itemsets grows, regardless of the value of the support, or the percentage of relevant documents in the original buckets.

We have used the sample P of sifted documents to train classification models. We conducted experiments using One-Class SVM, Two-class SVM (for this we use an estimated threshold to find potentially negative examples), and the techniques SPY, Rocchio, and SVM using the classification system LPU [16].As expected, the results obtained depend on the data set used, but in general all methods performed exceedingly well. This was specially true for our technique of bootstrapping positives (by DocMine) and negatives (by automatic thresholding), combined with a Two-class SVM approach, which showed robustness across all the scenarios tested.

In the future we will also consider an unsupervised learning approach, in which we take the resulting sample of documents given by DocMine, and use a clustering algorithm to find clusters of positives. Those clusters will be considered a good model of relevant documents, and used to filter possible outliers among the original documents. By measuring the fitness of each original document with respect to these clusters, we can prioritize the original set. We also plan to conduct more extensive experiments, including the real scenario of documents returned by several search engines.

References

[1] Barbará, D., Domeniconi, C., Kang, N. (2003). Mining Relevant Text from Unlabeled Documents. *Proceedings of the Third IEEE International Conference on Data Mining*

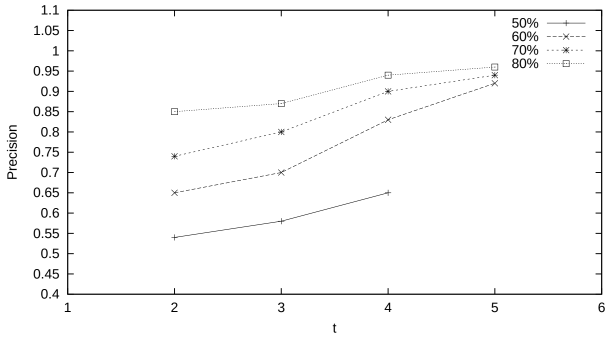

Figure 6: Precision values for topic *acq* and $Sup_{min} = 5\%$.The x-axis is the minimum cardinality of common itemsets (t).

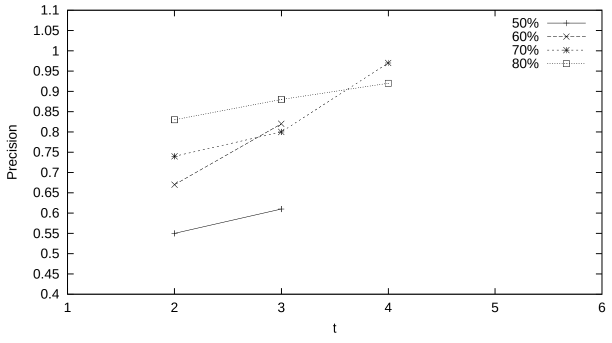

Figure 8: Precision values for topic *grain* and $Sup_{min} = 5\%$. The x-axis is the minimum cardinality of common itemsets (t).

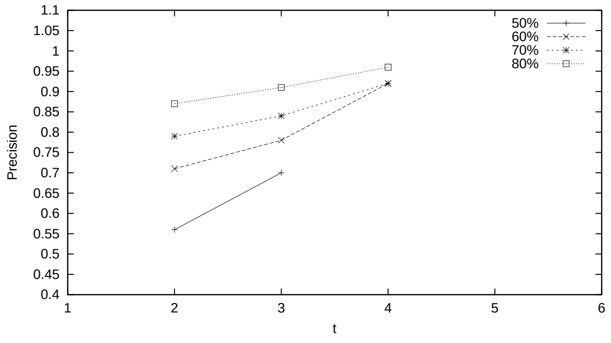

Figure 7: Precision values for topic *acq* and $Sup_{min} = 10\%$.The x-axis is the minimum cardinality of common itemsets (t).

[2] Blum, A., Mitchell T. (1998). Combining Labelled and Unlabelled Data with Co-Training. *Proceedings of the 1998 Conference on Computational Learning Theory.*

[3] Chang, C. C., Lin, C. J. (2003). LIBSVM – A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

[4] Chen, Y., Zhou, X. S., Huang, T. S. (2001). One-class SVM for learning in image retrieval. *Proceedings of International Conference on Image Processing.*

[5] Cristianini, N., Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines (and other kernel-based learning methods.* Cambridge University Press.

[6] Cristianini, N., Shawe-Taylor, J., Lodhi, L. (2002). Latent Semantic Kernels. *Journal of Intelligent Information Systems*, **18**(2):127–152.

[7] Duda, R. O., & Hart, P. E. (1973). *Pattern Classification and Scene Analysis*. John Wiley & Sons, Inc., New York.

[8] Dumais, S. T., Letsche, T. A., Littman, M. L., & Landauer, T. K. (1997). Automatic cross-language retrieval using latent semantic indexing. *AAAI Spring Symposium on Cross-Language Text and Speech Retrieval.*

[9] Fung, B. C. M., Wang, K., & Ester M. (2003). Hierarchical Document Clustering Using Frequent Itemsets. *Proceedings of the SIAM International Conference on Data Mining.*

[10] Joachims, T. (1998). Text categorization with support vector machines. *Proceedings of European Conference on Machine Learning.*

[11] Kandola, J., Shawe-Taylor, J., & Cristianini, N. (2002). Learning Semantic Similarity. *Neural Information Processing Systems (NIPS).*

[12] Leopold, E., & Kindermann, J. (2002). Text categorization with support vector machines, how to represent texts in input space? *Machine Learning*,**46**,423-444.

[13] Lewis, D., Reuters-21578 Text Categorization Test Collection Distribution 1.0. http://kdd.ics.uci.edu/databases/reuters21578/ reuters21578.html

[14] Liu B., Dai, Y., Li, Xiaoli, Lee W.S., and Yu, P. (2003). Building Text Classifiers Using Positive and Unlabeled Examples. *Proceedings of the Third IEEE International Conference on Data Mining.*

[15] Liu, B., Lee, W.S., Yu, P.S., Li, X. (2003). Partially Supervised Classification of Text Documents. *Proceedings of the International Conference on Machine Learning.*

[16] Liu, B., Li, X. (2003). LPU: Learning from Positive and Unlabeled Text Documents. http://www.cs.uic.edu/∼liub/LPU/LPU-download.html

[17] Porter, M. (1980). An algorithm for suffix stripping, Program, **14**(3): 130-137 http://www.tartarus.org/∼martin/PorterStemmer

[18] Rocchio, J. (1971). Relevance Feedback in Information Retrieval. In Salton: *The SMART Retrieval System: Experiments in Automatic Document processing*, Chapter 14, pp. 313-323, Prentice-Hall.

[19] Scholkopf, B., Smola, A., Williamson, R., Bartlett, P.L. (2000). New support vector algorithms. *Neural Computation*, **12**:1207-1245

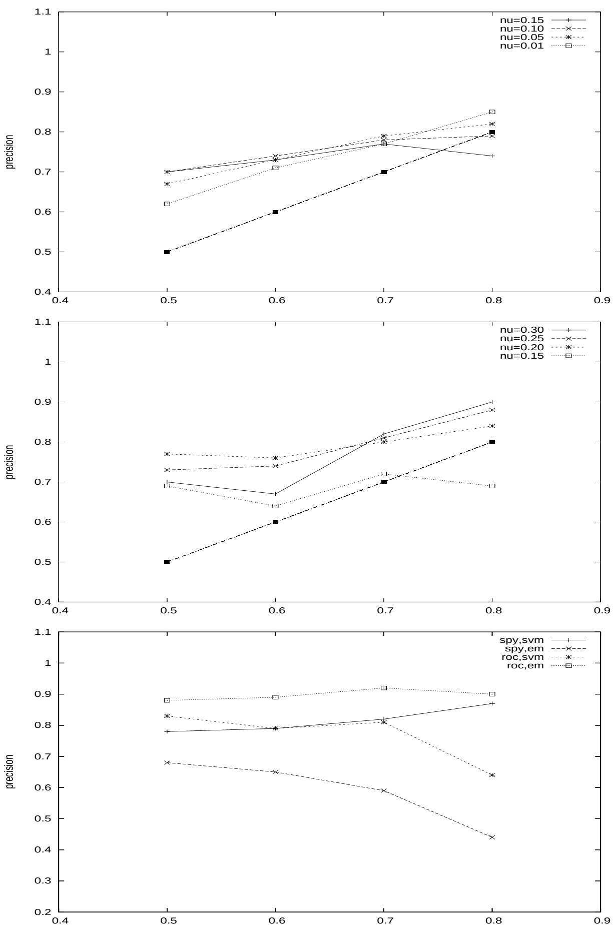

Figure 9: Classification performance for topic *earn* with (Top): One-class SVM, (Middle): Two-class SVM, (Bottom): (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). ($Sup_{min} = 3\%$, and minimum cardinality $t = 4$.) The x-axis corresponds to the R values.

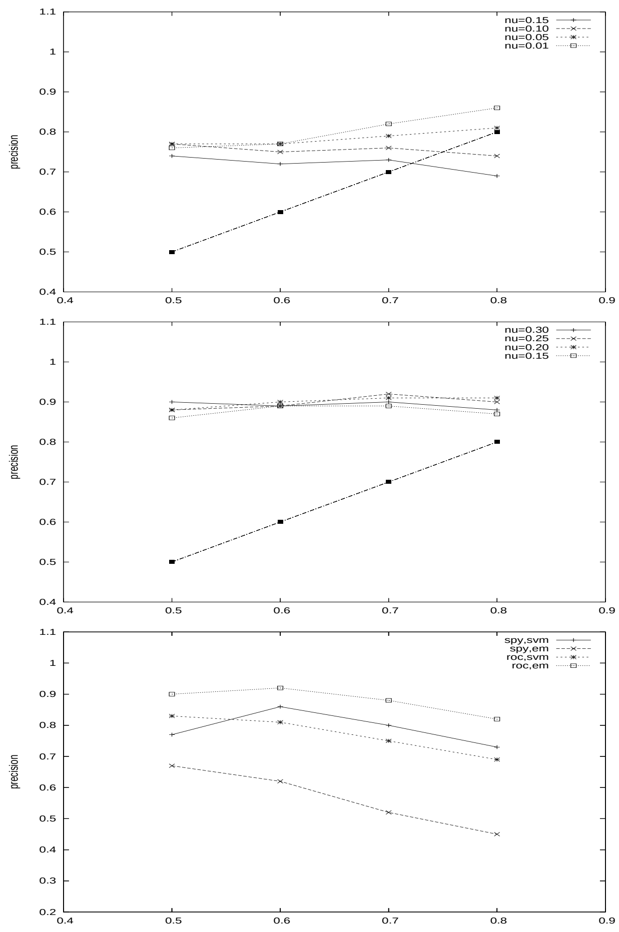

Figure 10: Classification performance for topic *earn* with (Top): One-class SVM, (Middle): Two-class SVM, (Bottom): (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). ($Sup_{min} = 3\%$, and minimum cardinality $t = 5$.) The x-axis corresponds to the R values.

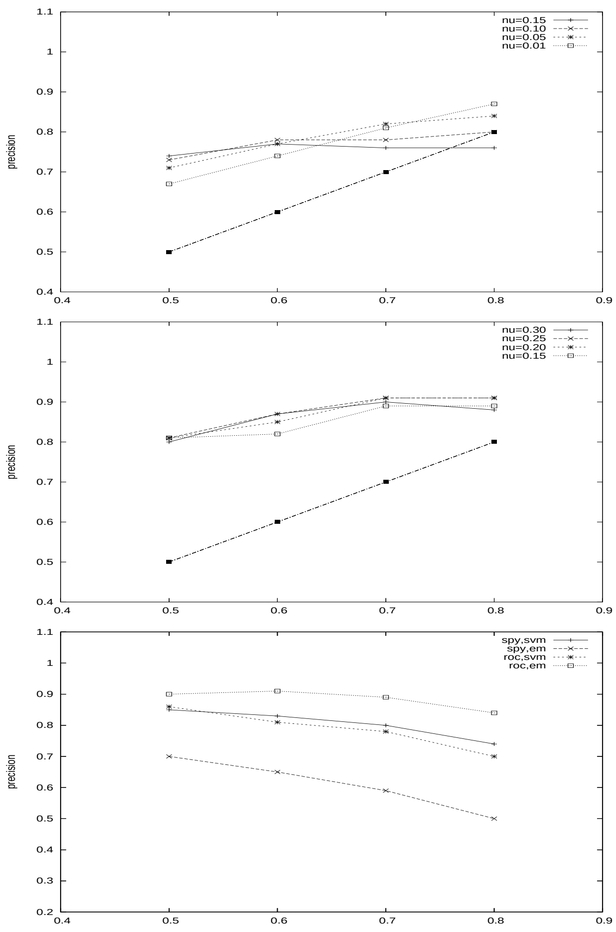

Figure 11: Classification performance for topic *earn* with (Top): One-class SVM, (Middle): Two-class SVM, (Bottom): (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). ($Sup_{min} = 5\%$, and minimum cardinality $t = 4$.) The x-axis corresponds to the R values.

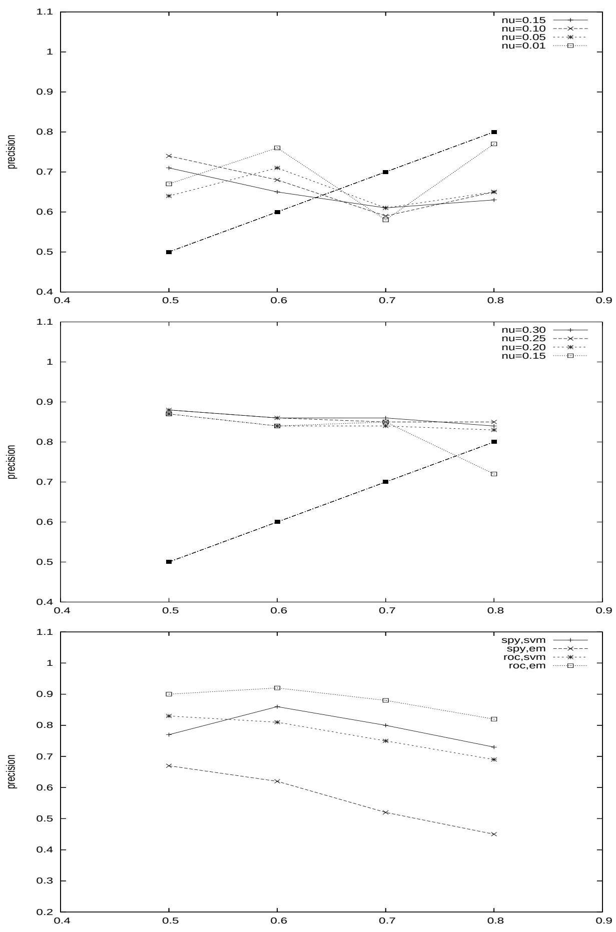

Figure 12: Classification performance for topic *earn* with (Top): One-class SVM, (Middle): Two-class SVM, (Bottom): (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). ($Sup_{min} = 5\%$, and minimum cardinality $t = 5$.) The x-axis corresponds to the R values.

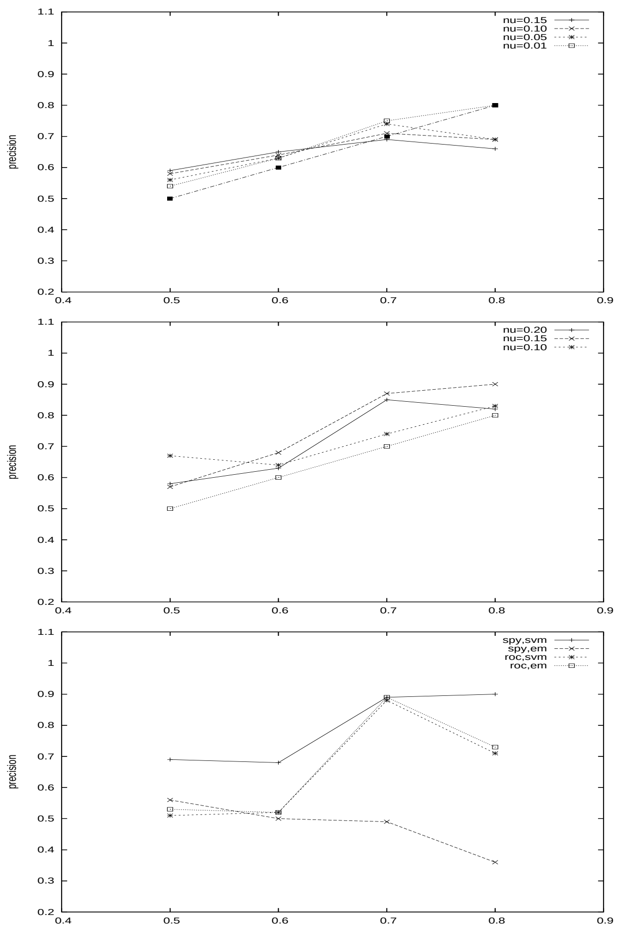

Figure 13: Classification performance for topic *acq* with (Top): One-class SVM, (Middle): Two-class SVM, (Bottom): (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). ($Sup_{min} = 3\%$, and minimum cardinality $t = 4$.) The x-axis corresponds to the R values.

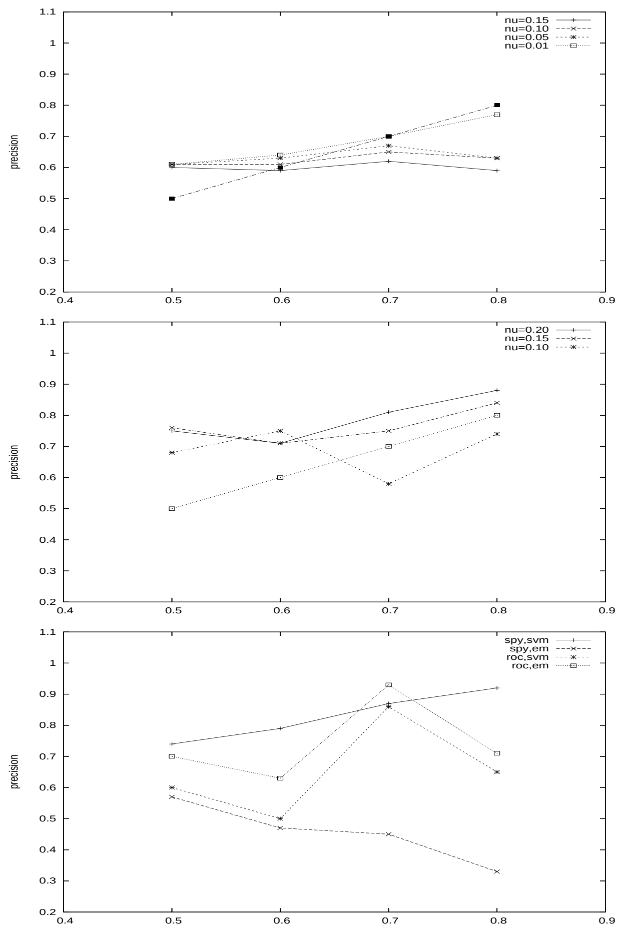

Figure 14: Classification performance for topic *acq* with (Top): One-class SVM, (Middle): Two-class SVM, (Bottom): (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). ($Sup_{min} = 5\%$, and minimum cardinality $t = 4$.) The x-axis corresponds to the R values.

[20] Scholkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, **13**:1443-1471.

[21] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

[22] Zhou, X. S., & Huang, T. S. (2001). Small sample learning during multimedia retrieval using BiasMap. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.