Classifying Documents Without Labels *

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

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 prove, via experimentation, that our method is capable of filtering relevant documents even in adverse conditions where the percentage of irrelevant documents in the buckets is relatively high.

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 [23, 5]. 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 [8]. 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 separate a set of unlabelled documents in two classes: positive (relevant) and negative (irrelevant) documents. Our approach reflects the asymmetry between positive

*This work is partially supported by NSF Grant IIS-0208519

and negative data, and does not make any particular and unnecessary assumption on the negative examples. We do not assume that the positive examples form a majority.

2 Related Work

In [10] 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.

[3] 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 [16] 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 [12] 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 [12] is to overcome this limitation by learning a *semantic proximity matrix* [7] 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 co-occurrent 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.

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 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. Now we compute all itemsets that are frequent in m buckets

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 frequent itemset , we select the documents that have the itemset 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) is fully described in [1] along with preliminary results on the purity of the sample of positives obtained.

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 [14], omitting empty documents and those without labels. Common and rare words are removed, and the vocabulary is stemmed with the Porter Stemmer [18]. 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.

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. 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 this 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.

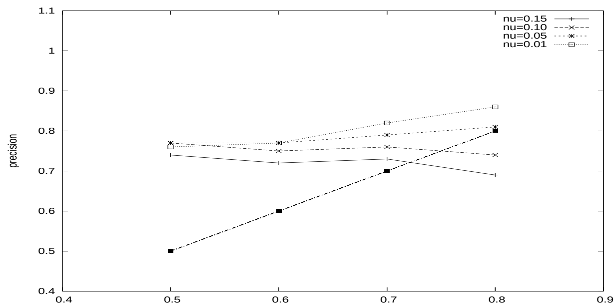

Figure 1: Classification performance for topic *earn* with One-class SVM. The x-axis corresponds to the R values.

We have observed that 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. For reasons of space, we only report here the precision values for one topic. Further results can be seen in [1] and in the longer version of this paper [2].

We compare the classification results of a number of different approaches (after applying DocMine), namely: One-class SVM [20, 21]; Two-class SVM [22, 6] (with bootstrapped negatives as described in Section 4); (SPY, SVM) [16, 17]; (SPY, EM) [16, 17]; (Rocchio, SVM) [19, 17]; (Rocchio, EM) [19, 17].

LIBSVM [4] is used to build the SVM classifiers. 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. The techniques SPY, SVM, Rocchio, and EM were tested using the classification system LPU [17]. 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 [15]. We emphasize that all the techniques in [15, 17] 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* for values of $Sup_{min} = 3\%$ and minimum cardinality 5, are shown in Figures 1-2. (Different settings were tested, and reported in [2].) 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. Figure 1 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. Figure 2 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. Although we cannot show all the results, for a variety of combinations of values of Sup_{min} and t, 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. Figure 3 shows 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 [17], 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 (Figure 3). As expected, different methods may be well suited for different 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.

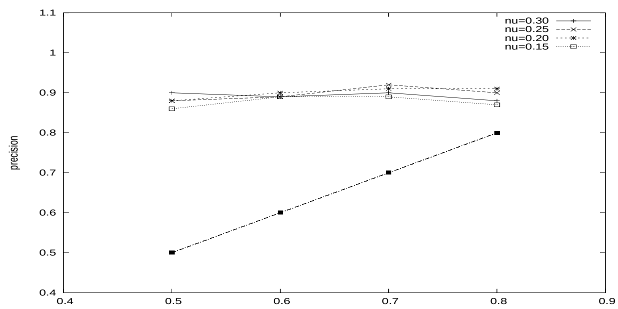

Figure 2: Classification performance for topic *earn* with Two-class SVM. The x-axis corresponds to the R values.

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 documents. 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.

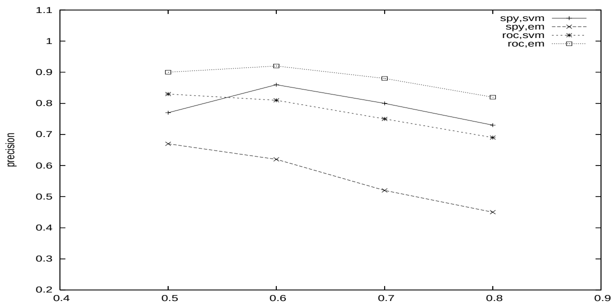

Figure 3: Classification performance for topic *earn* with (SPY, SVM), (SPY, EM), (Rocchio, SVM), (Rocchio, EM). The x-axis corresponds to the R values.

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*

[2] Barbará, D., Domeniconi, C., Kang, N. (2003).Classifying Documents Without Labels. Technical Report, Information and Software Engineering Department, George Mason University, January, 2004.

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

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

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

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

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

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

[9] 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.*

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

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

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

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

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

[15] 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.*

[16] 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.*

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

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

[19] 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.

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

[21] 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.

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

[23] 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.*

