Building Text Classifiers Using Positive and Unlabeled Examples

Bing Liu
Department of Computer Science
University of Illinois at Chicago {liub@cs.uic.edu}
Yang Dai
Department of Bioengineering
University of Illinois at Chicago {yangdai@uic.edu}
Xiaoli Li, Wee Sun Lee
School of Computing, National University of Singapore/Singapore-MIT Alliance
{lixl, leews}@comp.nus.edu.sg
Philip S. Yu
IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

This paper studies the problem of building text classifiers using positive and unlabeled examples. The key feature of this problem is that there is no negative example for learning. Recently, a few techniques for solving this problem were proposed in the literature. These techniques are based on the same idea, which builds a classifier in two steps. Each existing technique uses a different method for each step. In this paper, we first introduce some new methods for the two steps, and perform a comprehensive evaluation of all possible combinations of methods of the two steps. We then propose a more principled approach to solving the problem based on a biased formulation of SVM, and show experimentally that it is more accurate than the existing techniques.

1. Introduction

Text classification is the process of assigning pre-defined category labels to new documents based on the classifier learnt from training examples. In traditional classification, training examples are labeled with the same set of pre-defined category or class labels and labeling is often done manually. Many text classification techniques have been proposed by researchers so far, e.g., the Rocchio algorithm [29], the naive Bayesian method (NB) [17][22], support vector machines (SVM) [32][12] and many others (see [33]).

The main problem with this classic approach is that a large number of labeled training examples of every class are needed for accurate learning. Since labeling is typically done manually, it is labor intensive and time consuming. In recent years, researchers investigated the idea of using a small labeled set of every class and a large unlabeled set to help learning [26]. This reduces the manual labeling effort.

This paper studies the problem of building two-class classifiers with only positive and unlabeled examples, but no negative examples. Recently, a few algorithms were proposed to solve the problem. One class of algorithms is based on a two-step strategy. These algorithms include S-EM [20], PEBL [34], and Roc-SVM [18].

Step 1: Identifying a set of reliable negative documents from the unlabeled set. In this step, S-EM uses a Spy technique, PEBL uses a technique called 1-DNF, and Roc-SVM uses the Rocchio algorithm [29].

Step 2: Building a set of classifiers by iteratively applying a classification algorithm and then selecting a good classifier from the set. In this step, S-EM uses the Expectation Maximization (EM) algorithm [7] with a NB classifier, while PEBL and Roc-SVM use SVM. Both S-EM and Roc-SVM have some methods for selecting the final classifier. PEBL simply uses the last classifier at convergence, which can be poor.

These two steps together can be seen as an iterative method of increasing the number of unlabeled examples that are classified as negative while maintaining the positive examples correctly classified. It was shown theoretically in [20] that if the sample size is large enough, maximizing the number of unlabeled examples classified as negative while constraining the positive examples to be correctly classified will give a good classifier.

In this paper, we first introduce another method for Step 1, i.e., the NB method, and another method for Step 2, i.e., SVM alone, and perform an evaluation of all 16 possible combinations of methods of Step 1 and Step 2. This also results in a benchmark system, called LPU (Learning from Positive and Unlabeled data), which is available on the first author's Web page. We then propose a more principled approach to solving this problem based on a biased formulation of SVM. Experimental results show that the new method is superior to all the existing two-step techniques.

* The work of Bing Liu is partially supported by the National Science Foundation under the NSF grant IIS-0307239.

2. Related Work

Traditional text classification techniques require labeled training examples of all classes to build a classifier [33]. They are thus not suitable for building classifiers using only positive and unlabeled examples.

A theoretical study of Probably Approximately Correct (PAC) learning from positive and unlabeled data was first conducted in [8]. [24] presents a theoretical study in the Bayesian framework. Sample complexity results for learning by maximizing the number of unlabeled examples labeled as negative while constraining the classifier to label all the positive examples correctly were presented in [20].

The S-EM technique is reported in [20]. The PEBL technique is reported in [34]. The Roc-SVM technique is reported in [18]. We will discuss them in detail later. Unlike these techniques which are based on the two-step strategy, [16] reports a logistic regression technique to solve the problem.

Besides maximizing the number of unlabeled examples labeled as negative, other methods for learning from positive and unlabeled examples are possible. A NB based method (called PNB) that tries to statistically remove the effect of positive data in the unlabeled set is proposed in [9]. The main shortcoming of the method is that it requires the user to give the positive class probability, which is hard for the user to provide in practice. It is also possible to discard the unlabeled data and learn only from the positive data. This was done in the one-class SVM [31][21], which tries to learn the support of the positive distribution. Our results show that its performance is poorer than learning methods that take advantage of the unlabeled data.

Finally, our work is related to learning using a small labeled set and a large unlabeled set [2][3][4][5][10] [11][25][26][28][35]. In these works, a small set of labeled examples of every class and a large unlabeled set are used for classifier building. It was shown that the unlabeled data helps learning. These works are different from ours as we have no negative example.

3. Techniques for Step 1

In this section, we first introduce the naïve Bayesian technique (NB) as a new method for Step 1 to identify a set *RN* of reliable negative documents from the unlabeled set *U* (we use *P* to denote the positive example set). We then describe the Rocchio technique used in Roc-SVM, the Spy technique used in S-EM, and the 1-DNF technique used in PEBL to facilitate our later evaluation.

3.1 The Naïve Bayesian classifier

The naïve Bayesian technique is a popular method for classification. Given a set of training documents *D*, each document is considered an ordered list of words. We use $x_{d_i,k}$ to denote the word x_t in position *k* of document d_i, where x_t is a word in the vocabulary $V = \{x_1, \ldots, x_{|V|}\}$. The vocabulary is the set of all words considered for classification. Let $C = \{c_1, c_2, \ldots, c_{|C|}\}$ be a set of pre-defined classes (in this paper we only consider two classes, thus $C = \{c_1, c_2\}$). To perform classification, we compute the posterior probability, $Pr(c_j|d_i)$, where c_j is a class and d_i is a document. Based on the Bayesian probability and the multinomial model [22][26], we have

$$\Pr(c_j) = \frac{\sum_{i=1}^{|D|} \Pr(c_j \mid d_i)}{|D|}, \tag{1}$$

and with additive (Lidstone) smoothing [19]

$$\Pr(x_t \mid c_j) = \frac{\lambda + \sum_{i=1}^{|D|} N(x_t, d_i) \Pr(c_j \mid d_i)}{\lambda |V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N(x_s, d_i) \Pr(c_j \mid d_i)}, \tag{2}$$

where λ is the smoothing factor, $N(x_t, d_i)$ is the number of times that word x_t occurs in document d_i and $Pr(c_j|d_i) \in \{0, 1\}$ depending on the class of the document. Experimental results in [1] show that $\lambda = 0.1$ performs well for text data. $\lambda = 1$ is the commonly used Laplacian smoothing. However, $\lambda = 1$ is significantly inferior to $\lambda = 0.1$ [1]. We used $\lambda = 0.1$ in all our experiments.

Finally, assuming that the probabilities of the words are independent given the class, we obtain the NB classifier:

$$\Pr(c_j \mid d_i) = \frac{\Pr(c_j) \prod_{k=1}^{|d_i|} \Pr(x_{d_i,k} \mid c_j)}{\sum_{r=1}^{|C|} \Pr(c_r) \prod_{k=1}^{|d_i|} \Pr(x_{d_i,k} \mid c_r)}. \tag{3}$$

In classifying a document d_i, the class with the highest $Pr(c_j|d_i)$ is assigned as the class of the document.

Identifying a set *RN* of reliable negative documents from the unlabeled set *U* is done as follows (Figure 1):

1. Assign each document in *P* the class label 1;
2. Assign each document in *U* the class label -1;
3. Build a NB classifier using *P* and *U*;
4. Use the classifier to classify *U*. Those documents in *U* that are classified as negative form the reliable negative set *RN*.

Figure 1: The NB method for Step 1

3.2 The Rocchio technique

Rocchio is an early text classification method [29]. In this method, each document is represented as a vector, and each feature value in the vector is computed using the classic *tf-idf* scheme [30]. Let *D* be the whole set of training documents, and C_j be the set of training documents in class c_j. Building a Rocchio classifier is achieved by constructing a prototype vector $\vec{c}_j$ for each class c_j.

$$\vec{c}_j = \alpha \frac{1}{|C_j|}\sum_{\vec{d}\in C_j}\frac{\vec{d}}{\|\vec{d}\|} - \beta \frac{1}{|D-C_j|}\sum_{\vec{d}\in D-C_j}\frac{\vec{d}}{\|\vec{d}\|}.$$

α and β are parameters that adjust the relative impact of relevant and irrelevant training examples. [6] recommends α = 16 and β = 4. In classification, for each test document *td*, it uses the cosine similarity measure [30] to compute the similarity of *td* with each prototype vector. The class whose prototype vector is more similar to *td* is assigned to *td*.

The algorithm that uses Rocchio to identify a set *RN* of reliable negative documents from *U* is the same as that in Figure 1 except that we replace NB with Rocchio.

3.3 The Spy technique in S-EM

The Spy technique in S-EM is given in Figure 2. It first randomly selects a set *S* of positive documents from *P* and put them in *U* (lines 2 and 3). The default value for *s*% is 15% in S-EM. Documents in *S* act as "spy" documents from the positive set to the unlabeled set *U*. The spies behave similarly to the unknown positive documents in *U*. Hence, they allow the algorithm to infer the behavior of the unknown positive documents in *U*. It then runs I-EM algorithm using the set *P*–*S* as positive and the set $U \cup S$ as negative (lines 3-7). I-EM basically runs NB twice (see the EM algorithm below). After I-EM completes, the resulting classifier uses the probabilities assigned to the documents in *S* to decide a probability threshold *th* to identify possible negative documents in *U* to produce the set *RN*. See [20] for details.

3.4 The 1-DNF technique in PEBL

The 1-DNF method (Figure 3) first builds a positive feature set *PF* which contains words that occur in the positive set *P* more frequently than in the unlabeled set *U* (lines 1-5). In lines 6-9, it tries to filter out possible positive documents from *U*. A document in *U* that does not have any positive feature in *PF* is regarded as a strong negative document.

```
1.  $RN = NULL$;
2.  $S = Sample(P, s\%)$;
3.  $Us = U \cup S$;
4.  $Ps = P - S$;
5.  Assign each document in $Ps$ the class label 1;
6.  Assign each document in $Us$ the class label -1;
7.  I-EM($Us$, $Ps$);          // This produces a NB classifier.
8.  Classify each document in $Us$ using the NB classifier;
9.  Determine a probability threshold $th$ using $S$;
10. for each document $d \in Us$
11.     if its probability $Pr(1|d) < th$ then
12.         $RN = RN \cup \{d\}$;
```

Figure 2: The Spy technique in S-EM.

```
1. Assume the word feature set be $\{x_1,\ldots, x_n\}$, $x_i \in U \cup P$;
2. Let positive feature set $PF$ = null;
3. for $i$ = 1 to $n$
4.     if ($freq(x_i, P)/|P| > freq(x_i, U)/|U|$) then
5.         $PF = PF \cup \{x_i\}$;
6. $RN = U$;
7. for each document $d \in U$
8.     if $\exists x_j\, freq(x_j, d) > 0$ and $x_j \in PF$ then
9.         $RN = RN - \{d\}$;
```

Figure 3: The 1-DNF technique in PEBL.

4. Techniques for Step 2

Four techniques are given here for the second step:

1. Run SVM only once using sets *P* and *RN* after Step 1. This method is not used before.
2. Run EM. This method is used in S-EM.
3. Run SVM iteratively. This method is used in PEBL.
4. Run SVM iteratively and then select a final classifier. This method is used in Roc-SVM.

We now discuss the 4 methods in turn.

4.1 Support Vector Machines (SVM)

Support vector machines (SVM) are linear functions of the form $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$, where $\mathbf{w}^T\mathbf{x}$ is the inner product between the weight vector **w** and the input vector **x**. SVM is used as a classifier by setting the class to 1 if $f(\mathbf{x}) > 0$ and to –1 otherwise. The main idea of SVM is to select a hyperplane that separates the positive and negative examples while maximizing the smallest margin. Let a set of training examples be $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i$ is an input vector and y_i is its class label, $y_i \in \{1, -1\}$. The problem of finding the hyperplane can be stated as the following optimization problem:

$$\text{Minimize: } \frac{1}{2}\mathbf{w}^T\mathbf{w}$$
$$\text{Subject to: } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, 2, ..., n$$

To deal with cases where there may be no separating hyperplane due to noisy labels of both positive and negative training examples, the soft margin SVM is proposed [12], which is formulated as:

$$\text{Minimize: } \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$
$$\text{Subject to: } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, ..., n$$

where $C \geq 0$ is a parameter that controls the amount of training errors allowed.

4.2 The EM algorithm in S-EM

The Expectation-Maximization (EM) algorithm is a popular iterative algorithm for maximum likelihood estimation in problems with missing data [7]. The EM algorithm consists of two steps, the *Expectation* step, and the *Maximization* step. The *Expectation* step basically fills in the missing data. In our case, it produces and revises the probabilistic labels of the documents in *U-RN* (see below). The parameters are estimated in the *Maximization* step after the missing data are filled. This leads to the next iteration of the algorithm. EM converges when its parameters stabilize. Using NB in each iteration, EM employs the same equations as those used in building a NB classifier (equations (1) and (2) for the *Expectation* step, and equation (3) for the *Maximization* step) [26] [20]. The class probability given to each document now takes the value in [1, 0] instead of {1, 0}. The algorithm is given in Figure 4.

```
1. Each document in $P$ is assigned the class label 1;
2. Each document in $RN$ is assigned the class label -1;
3. Each document $d \in Q$ (= $U - RN$) is not assigned
   any label initially. At the end of the first iteration
   of EM, it will be assigned a probabilistic label,
   $Pr(1|d)$. In subsequent iterations, the set $Q$ will
   participate in EM with its newly assigned
   probabilistic classes.
4. Run the EM algorithm using the document sets, $P$,
   $RN$ and $Q$ until it converges.
```

Figure 4: The EM algorithm with the NB classifier.

Basically, EM iteratively runs NB to revise the probabilistic label of each document in set $Q = U - RN$. Since each iteration of EM produces a NB classifier, S-EM also has a mechanism to select a good classifier [20].

4.3 Iterative SVM in PEBL

PEBL uses *P*, *RN* and *U-RN* to run SVM iteratively (Figure 5). The basic idea is to use each iteration of SVM to extract more possible negative data from *U-RN* and put them in *RN* because PEBL's first step is only able to identify a very small set of negative documents. Let Q be the set of remaining unlabeled documents, $Q = U - RN$. The iteration converges when no document in Q is classified as negative. The final classifier is the result.

```
1.  Every document in $P$ is assigned the class label 1;
2.  Every document in $RN$ is assigned the class label –1;
3.  $i = 1$;
4.  Loop
5.      Use $P$ and $RN$ to train a SVM classifier $S_i$;
6.      Classify $Q$ using $S_i$;
7.      Let the set of documents in $Q$ that are classified as
            negative be $W$;
8.      if $W = \{\}$ then exit-loop
9.      else $Q = Q - W$;
10.         $RN = RN \cup W$;
11.         $i = i + 1$;
```

Figure 5: Running SVM iteratively.

4.4 Iterative SVM with Classifier Selection in Roc-SVM

This method is similar to the method in Section 4.3 except that it also decides which classifier to use after the algorithm in Figure 5 converges because each SVM iteration builds a different SVM classifier, and the last classifier may not be a good classifier. After iterative SVM converges, we add the following four lines:

```
1. Use the last SVM classifier $S_{last}$ to classify $P$;
2. if > 8% positive are classified as negative then
3.     use $S_1$ as the final classifier;
4. else use $S_{last}$ as the final classifier;
```

Figure 6: Classifier selection.

The reason for selecting a classifier is that there is a danger in running SVM repetitively. Since SVM is sensitive to noise, if some iteration of SVM extracts many positive documents from Q and put them in *RN*, then the last SVM classifier will be poor. This is the problem with PEBL. In this algorithm, we decide whether to use the first SVM classifier or the last one. Basically, we use the SVM classifier at convergence (called S_{last} in line 1) to classify *P*. If too many (> 8%) positive documents in *P* are classified as negative, it indicates that SVM has gone wrong. We then use the first classifier (S_1). Otherwise, we use S_{last} as the final classifier. 8% is used as the threshold because we want to be very conservative so that we will not select a very weak last SVM classifier at convergence.

The above method will not work if both the first and the last SVM classifiers are poor. This is often the case for PEBL because its Step 1 extracts too few negative documents from *U* and thus results in a weak first classifier. PEBL's last classifier may be weak also because one of the iterative SVMs may go wrong. If we use Spy or Rocchio in Step 1, the first SVM is often quite strong, although it may not be the best. Note that neither the first nor the last SVM may be the best classifier. In many cases, a SVM classifier somewhere in the middle is the best. However, it is hard to catch the best classifier.

5. The Proposed Biased SVM

We now present the proposed biased SVM formulation of the problem. Let the set of training examples be $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i$ is an input vector and y_i is its class label, $y_i \in \{1, -1\}$. Assume that the first k-1 examples are positive examples (labeled 1), while the rest are unlabeled examples, which we label negative (-1). It was shown in [20] that if the sample size is large enough, minimizing the number of unlabeled examples classified as positive while constraining the positive examples to be correctly classified will give a good classifier. In the noiseless case, this results in the following SVM formulation (no error for positive examples but only for unlabeled examples).

$$\text{Minimize: } \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=k}^{n} \xi_i$$

$$\text{Subject to: } \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \geq 1, \quad i = 1, 2, ..., k-1$$

$$-1(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = k, k+1, ..., n$$

$$\xi_i \geq 0, i = k, k+1 \ldots, n$$

To distinguish this formulation with the classic SVM we call it *Biased-SVM*. If we also allow noise (or error) in positive examples, we have the following soft margin version of the Biased-SVM formulation which uses two parameters C_+ and C_- to weight positive errors and negative errors differently.

$$\text{Minimize: } \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C_+\sum_{i=1}^{k-1} \xi_i + C_-\sum_{i=k}^{n} \xi_i$$

$$\text{Subject to: } y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, ..., n$$

$$\xi_i \geq 0, i = 1, 2, \ldots, n$$

We can vary C_+ and C_- to achieve our objective. Intuitively, we give a big value for C_+ and a small value for C_- because the unlabeled set, which is assumed to be negative, also contains positive data. Note that this asymmetric cost formulation has been used to solve unbalance data problem in [23] (one class of data is very small, while the other class is very large). Our formulation, however, is based on a different motivation.

To choose C_+ and C_-, the common practice is to use a separate validation set to verify the performance of the resulting classifier with the selected values for C_+ and C_-. Since the need to learn from positive and unlabeled examples often arises in retrieval situations, we employ the commonly used F score as the performance measure, F = $2pr/(p+r)$, where p is the *precision* and r is the *recall*.

Unfortunately it is not clear how to estimate the F score without negative examples. In [16], a performance criteria for comparing different classifiers is proposed, which can be estimated directly from the validation set without the need of negative examples.

Let X be the random variable representing the input vector, Y be the actual label. The criteria is $pr/\Pr[Y = 1]$, where $\Pr[Y = 1]$ is the probability of actual positive documents. In [16], it is shown that $pr/\Pr[Y = 1] = r^2/\Pr[f(X) = 1]$, where $\Pr[f(X) = 1]$ is the probability that a document is classified as positive. r can be estimated using the positive examples in the validation set and $\Pr[f(X) = 1)$ can be estimated from the whole validation set. This criteria works because it behaves similarly to the F score in the sense that it is large when both p and r are large and is small if either p or r is small.

6. Empirical Evaluation

We now evaluate all the techniques of the two-step approach and the new biased-SVM problem formulation.

6.1 Experimental Setup

Datasets: We used two popular text collections in our experiments. The first one is the Reuters-21578[1], which has 21578 documents collected from the Reuters newswire. Of the 135 categories, only the most populous 10 are used. Each category is employed as the positive class, and the rest as the negative class. This gives us 10 datasets. The second collection is the Usenet articles[2] collected by Lang [15] from 20 different newsgroups. Each group has approximately 1000 articles. We use each newsgroup as the positive set and the rest of the 19 groups as the negative set, which creates 20 datasets. In data pre-processing, we applied *stopword* removal, but no feature selection or *stemming* were done. For Rocchio and SVM, *tf-idf* values are used in the feature vectors.

For each dataset, 30% of the documents are randomly selected as test documents. The rest (70%) are used to create training sets as follows: γ percent of the documents from the positive class is first selected as the positive set P. The rest of the positive documents and negative documents are used as unlabeled set U. We range γ from 10%-90% (0.1-0.9) to create a wide range of scenarios.

Experimental systems: Experiments on S-EM and Roc-SVM were conducted using our own systems. Since PEBL is not publicly available, we implemented it based on [34]. For SVM, we used the SVMlight system with linear kernel [14]. All the other methods are implemented by us.

Evaluation measure: In our experiments, we use the popular F score on the positive class as the evaluation measure. F score takes into account of both recall (r) and precision (p), F = $2pr/(p+r)$. We also have accuracy results. However, due to space limitations, we do not list them here. Accuracies behave similarly to F scores.

6.2 Results of the two-step strategy

Below we first summarize all the methods studied in this paper for the two-step approach.

Step 1:

1. Spy: This is the method used in S-EM.
2. 1-DNF: This is the method used in PEBL.
3. Rocchio: This method used in Roc-SVM.

[1] http://www.research.att.com/~lewis/reuters21578.html

[2] http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes/20_newsgroups.tar.gz

4. NB: This method is proposed in this paper.

Step 2:

1. EM: This is the method used S-EM
2. SVM: This method is proposed in this paper. It runs SVM only once after Step 1.
3. SVM-I: This method is used in PEBL. It runs SVM iteratively. The last classifier at convergence is the final classifier evaluated on the test data.
4. SVM-IS: This method is used in Roc-SVM. It runs SVM iteratively with classifier selection, i.e., after iterative SVM converges, it selects either the first or the last classifier as the final classifier.

Clearly, each technique for Step 1 can be combined with each technique for Step 2. We will empirically evaluate all the 16 possible combinations[3].

Table 1 shows the macro-averaged F score of the 10 Reuters datasets for each γ setting. Due to space limitations, we are unable to list all the detailed results. All the F scores are obtained from unseen test sets.

Columns 1 to 16 show the F scores of all 16 combinations of methods in Steps 1 and 2. PEBL is 1-DNF combined with SVM-I, S-EM is Spy combined with EM, and Roc-SVM is Rochio+SVM-IS. Column 17 gives the results of NB alone for each γ setting. In this case, NB simply treats all the documents in the unlabeled set as negative examples. Its results allow us to see whether all the sophisticated techniques work (more on this later).

Table 2 shows the macro-averaged F scores of the 20 20Newsgroup datasets for each γ setting. From the results in Tables 1 and 2, we draw the following conclusions.

1. S-EM: Its performance is stable over a wide range of conditions. However, it is only comparable with others when the positive set is very small (γ = 0.1-0.3). When the positive set is large, it is worse than some other combinations. These observations are also true for columns 9 and 13. The reasons are that EM uses NB, which is a weaker classifier than SVM [13][33], and that these data sets are not suitable for NB and EM. See the detailed discussion for point 7 below.
2. PEBL: Its performance is poor when the number of positive examples is small. The reason is that PEBL's second step can go wrong without a large number of positive documents. When the positive set is large, it becomes more stable. Its results are comparable with others for Reuters datasets. However, for 20Newsgroup datasets, it is worse than many others e.g., Spy+SVM, Roc-SVM, and NB+SVM.
3. Spy+SVM: It underperforms a few other combinations when the positive set is very small (γ = 0.1-0.2). The reason is that when the positive set is small the number of spies put in the unlabeled set will be too small and thus the resulting *RN* set will not be very reliable. However, it is the best method as long as the positive set is not too small. NB+SVM also gives good results in such cases, although it is slightly inferior to Spy+SVM. We believe that in practice most positive sets are reasonably large because the user is aware that without a sufficiently large and/or representative positive set, he/she will not obtain a good result. These two methods are also very efficient because they run SVM only once.
4. Roc-SVM and Rocchio+SVM-I: They are also good techniques for large positive sets.
5. 1-DNF, Spy, Rocchio, and NB for Step 1: 1-DNF is weaker for Step 1. It is only good for Reuters data with large positive sets. Spy and Rocchio are more robust. NB is slightly weaker than them.
6. SVM vs EM (NB) for Step 2: It is clear that SVM-based methods for Step 2 (SVM, SVM-I and SVM-IS) significantly outperform EM-based methods when the positive set is reasonably large. It is well-known that SVM is a stronger classifier than NB (EM uses NB).
7. NB alone: It is interesting to observe that a single NB (column 17 of both tables) slightly outperforms S-EM (column 5) and Rocchio+EM (column 9) and NB+EM (column 13). It is known that NB is able to tolerate some noise. Running EM actually makes the results worse. The reason is that EM has a weakness due to the assumptions of NB (our EM runs NB multiple times). In devising the NB classifier, two assumptions are made [22][26]: (1) words are independent given a class, and (2) text documents are generated by a mixture model and there is a one-to-one mapping between mixture components and classes (which means that each class contains only documents of one topic or category). This is not true for our situations because each of our negative class contains documents from many diverse topic categories. Thus, the more we run NB, the worse the results get. This phenomenon is also mentioned in [26]. In [20], it is shown that S-EM outperforms NB because most of its datasets contain only documents of two topics. Then, the above assumption (2) is satisfied. However, we believe that in practice, the negative class typically contains documents from many topics. We noticed in our experiments that the results become worse and worse with each iteration of EM. However, EM in S-EM has a classifier selection mechanism which is able to select the first classifier almost all the time. This means that EM is simply NB for Step 2.

Note that there are a few other methods that can be used alone just like NB, e.g., Rocchio, SVM, and one-

[3] The PNB system in [9] is not compared as it requires the user to input the positive class probability, which is hard for the user to supply in practice. Furthermore, as we will see, NB based techniques (PNB is based on NB) are inferior to SVM based techniques. We plan to compare with the logistic regression based approach in [16] in the near future.

Table 1: Average F scores on Reuters collection

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Step1	1-DNF	1-DNF		1-DNF		Spy	Spy	Spy	Rocchio	Rocchio	Rocchio		NB	NB	NB	NB	
Step2	EM	SVM	PEBL	SVM-IS	S-EM	SVM	SVM-I	SVM-IS	EM	SVM	SVM-I	Roc-SVM	EM	SVM	SVM-I	SVM-IS	NB
0.1	0.187	0.423	0.001	0.423	0.547	0.329	0.006	0.328	0.644	0.589	0.001	0.589	0.547	0.115	0.006	0.115	0.514
0.2	0.177	0.242	0.071	0.242	0.674	0.507	0.047	0.507	0.631	0.737	0.124	0.737	0.693	0.428	0.077	0.428	0.681
0.3	0.182	0.269	0.250	0.268	0.659	0.733	0.235	0.733	0.623	0.780	0.242	0.780	0.695	0.664	0.235	0.664	0.699
0.4	0.178	0.190	0.582	0.228	0.661	0.782	0.549	0.780	0.617	0.805	0.561	0.784	0.693	0.784	0.557	0.782	0.708
0.5	0.179	0.196	0.742	0.358	0.673	0.807	0.715	0.799	0.614	0.790	0.737	0.799	0.685	0.797	0.721	0.789	0.707
0.6	0.180	0.211	0.810	0.573	0.669	0.833	0.804	0.820	0.597	0.793	0.813	0.811	0.670	0.832	0.808	0.824	0.694
0.7	0.175	0.179	0.824	0.425	0.667	0.843	0.821	0.842	0.585	0.793	0.823	0.834	0.664	0.845	0.822	0.843	0.687
0.8	0.175	0.178	0.868	0.650	0.649	0.861	0.865	0.858	0.575	0.787	0.867	0.864	0.651	0.859	0.865	0.858	0.677
0.9	0.172	0.190	0.860	0.716	0.658	0.859	0.859	0.853	0.580	0.776	0.861	0.861	0.651	0.846	0.858	0.845	0.674

Table 2: Average F scores on 20Newsgroup collection

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Step1	1-DNF	1-DNF		1-DNF		Spy	Spy	Spy	Rocchio	Rocchio	Rocchio		NB	NB	NB	NB	
Step2	EM	SVM	PEBL	SVM-IS	S-EM	SVM	SVM-I	SVM-IS	EM	SVM	SVM-I	Roc-SVM	EM	SVM	SVM-I	SVM-IS	NB
0.1	0.145	0.545	0.039	0.545	0.460	0.097	0.003	0.097	0.557	0.295	0.003	0.295	0.368	0.020	0.003	0.020	0.333
0.2	0.125	0.371	0.074	0.371	0.640	0.408	0.014	0.408	0.670	0.546	0.014	0.546	0.649	0.232	0.013	0.232	0.611
0.3	0.123	0.288	0.201	0.288	0.665	0.625	0.154	0.625	0.673	0.644	0.121	0.644	0.689	0.469	0.120	0.469	0.674
0.4	0.122	0.260	0.342	0.258	0.683	0.684	0.354	0.684	0.671	0.690	0.385	0.682	0.705	0.610	0.354	0.603	0.704
0.5	0.121	0.248	0.563	0.306	0.685	0.715	0.560	0.707	0.663	0.716	0.565	0.708	0.702	0.680	0.554	0.672	0.707
0.6	0.123	0.209	0.646	0.419	0.689	0.758	0.674	0.746	0.663	0.747	0.683	0.738	0.701	0.737	0.670	0.724	0.715
0.7	0.119	0.196	0.715	0.563	0.681	0.774	0.731	0.757	0.660	0.754	0.731	0.746	0.699	0.763	0.728	0.749	0.717
0.8	0.124	0.189	0.689	0.508	0.680	0.789	0.760	0.783	0.654	0.761	0.763	0.766	0.688	0.780	0.758	0.774	0.707
0.9	0.123	0.177	0.716	0.577	0.684	0.807	0.797	0.798	0.654	0.775	0.798	0.790	0.691	0.806	0.797	0.798	0.714

class SVM. In fact, we experimented each of them. However, their results are poor and thus are not listed here. For example, the F score for one-class SVM is around 0.3-0.5 for most datasets.

8. Pure NB and pure SVM: We also obtain the F scores (see the table below) in the pure case with original (70%) training data, i.e., no positive examples are added to the negative set to form the unlabeled set.

	Reuters (F)	20Newsgroup (F)
NB	0.670	0.709
SVM	0.870	0.792

Comparing the results here with the results in Tables 1 and 2, we observe that when the positive set is large, the F scores of a few SVM based methods are very close to the pure case (sometime better than the pure case). However, when the number of positive documents is not that large, there is still room for further improvements.

6.3 Results of Biased-SVM

In this set of experiments, we again used the SVMlight package, which allows control of C_+ and C_- through the parameters *c* and *j*, where *c* is C_- and $j = C_+/C_-$. In our experiments, we varied *c* from 0.01, 0.03, 0.05, …, 0.61 and *j* from 10, 20, 30, …, 200. We use 30% of the training documents as the validation set in each experiment. The classifier selection criterion $pr/\Pr[Y = 1]$ is used to select the best *c* and *j* parameters based on the validation set. The final test results are obtained by using the original training set after the parameter *c* and *j* have been selected. Note that we need to run SVM a large number of times. However, some heuristics have been designed to reduce the number of runs, which will be described in the full version of the paper.

We have performed experiments with $\gamma = 0.3$ and 0.7. The averaged results given in Table 3 show that Biased-SVM performs better than the (previous) best of all methods in Tables 1 and 2. We observe that when the positive set is small, the improvement is more significant. This is especially true for the 20Newsgroup collection, which is a harder collection.

Table 3: Average F scores on the two collections

	γ	Average F score of Biased-SVM	Previous best F score
Reuters	0.3	0.785	0.78
	0.7	0.856	0.845
20Newsgroup	0.3	0.742	0.689
	0.7	0.805	0.774

7. Conclusions

In this paper, we discussed the two-step strategy for learning a classifier from positive and unlabeled data. Two new methods were added to the existing techniques. A comprehensive evaluation of all the combinations of methods was conducted to compare their performances, which enables us to draw some important conclusions. We also proposed a more principled approach to solving the problem based on a biased formulation of SVM. Our results show that in general Biased-SVM outperforms all the existing two-step techniques.

References

[1]. Agrawal, R., Bayardo Jr., R. & Srikant, R. (2000) "Athena: Mining-based interactive management of text databases." *EDBT*-00.

[2]. Basu, S., Banerjee, A., & Mooney, R. (2002) "Semi-supervised clustering by seeding." *ICML-02*.

[3]. Bennett, K., and Demiriz. (1998) "A Semi-supervised support vector machines." *Advances in Neural information processing systems* 11.

[4]. Blum, A., Mitchell, T. (1998) "Combining labeled and unlabeled data with co-training," *COLT-98*.

[5]. Bockhorst, J., and Craven, M. (2002) "Exploiting relations among concepts to acquire weakly labeled training data." *ICML-02*.

[6]. Buckley, C., Salton G., and Allan J. (1994) "The effect of adding relevance information in a relevance feedback environment," *SIGIR-94*.

[7]. Dempster, A., Laird, N. M. & Rubin. D. (1997) "Maximum likelihood from incomplete data via the EM algorithm." *Journal of the Royal Statistical Society*, B:39, 1-38.

[8]. Denis, F. "PAC learning from positive statistical queries." *ALT-98*.

[9]. Denis, F. Gilleron, R and Tommasi, M. (2002). "Text classification from positive and unlabeled examples." *IPMU*, 2002.

[10]. Ghani, R. (2002) "Combining labeled and unlabeled data for multiclass text categorization." *ICML-2002*.

[11]. Goldman, S. and Zhou, Y. (2000) "Enhancing supervised learning with unlabeled data." *ICML-00*.

[12]. Guyon, I., Boser, B. and Vapnik, V. (1993). "Automatic capacity tuning of very large VC-dimension classifiers." *Advances in Neural Information Processing Systems*, Vol. 5.

[13]. Joachims, T. (1998) "Text categorization with support vector machines: Learning with many relevant features." *ECML-98*.

[14]. Joachims, T. (1999). "Making large-scale SVM learning practical." *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola (ed.).

[15]. Lang, K. (1995). "Newsweeder: Learning to filter netnews." *ICML-95*.

[16]. Lee, W. S, and Liu, B. (2003) "Learning with positive and unlabeled examples using weighted logistic regression." *ICML-2003*.

[17]. Lewis, D., and Gale, W. (1994). "A sequential algorithm for training text classifiers." *SIGIR-94*.

[18]. Li, X., and Liu, B. (2003). "Learning to classify text using positive and unlabeled data." *IJCAI-03*.

[19]. Lidstone, G. (1920). "Note on the general case of the Bayes-Laplace formula for inductive or a posteriori probabilities." *Transactions of the Faculty of Actuaries,* 8:182-192.

[20]. Liu, B., Lee, W. S., Yu, P., and Li, X. (2002). "Partially supervised classification of text documents." *ICML-02*.

[21]. Manevitz, L & Yousef, M. (2001). "One-class SVMs for document classification." *J. of Machine Learning research*, 2.

[22]. McCallum, A., Nigam, K. (1998) "A comparison of event models for naïve Bayes text classification." *AAAI-98 Workshop on Learning for Text Categorization.* 1998.

[23]. Morik, K., Brockhausen, P. and Joachims, T. (1999) "*Combining statistical learning with a knowledge-based approach - A case study in intensive care monitoring*." *ICML-99*, 1999.

[24]. Muggleton, S. (2001). "Learning from the positive data." *Machine Learning*, Accepted.

[25]. Muslea, I., Minton, S., and Knoblock, C. A. (2002). "Active + semi-supervised learning = robust multi-view learning." *ICML-02*.

[26]. Nigam, K., McCallum, A., Thrun, S. and & Mitchell, T. (2000). "Text classification from labeled and unlabeled documents using EM." *Machine Learning*, 39.

[27]. Osuna, E., R. Freund, and F. Girosi (1997). Support vector machines: Training and applications. AI Memo 1602, Massachusetts Institute of Technology.

[28]. Rakutti, B. Ferra, H. Kowalczyk, A. (2002). "Using unlabeled data for text classification through addition of cluster parameters." *ICML-02*.

[29]. Rocchio, J. (1971). "Relevant feedback in information retrieval." In G. Salton (ed.). *The smart retrieval system- experiments in automatic document processing*, Englewood Cliffs, NJ.

[30]. Salton, G. and McGill, M. (1983). *Introduction to Modern Information Retrieval*. McGraw-Hill.

[31]. Scholkopf, S. Platt, J. Shawe, J. Smola, A. & Williamson, R. (1999). "Estimating the support of a high-dimensional distribution." *Technical Report MSR-TR-99-87*, Microsoft Research.

[32]. Vapnik, V. (1995). *The nature of statistical learning theory*, Springer-Verlag, NY, USA, 1995

[33]. Yang, Y. and Liu, X. (1999). "A re-examination of text categorization methods." *SIGIR-99*.

[34]. Yu, H., Han, J. & Chang, K. (2002). "PEBL: Positive example based learning for Web page classification using SVM." *KDD-02*.

[35]. Zhang, T. (2000). "The value of unlabeled data for classification problems," *ICML-00*.