Partially Supervised Classification of Text Documents

Bing Liu LIUB@COMP.NUS.EDU.SG
Wee Sun Lee LEEWS@COMP.NUS.EDU.SG
School of Computing, National University of Singapore/Singapore-MIT Alliance, Singapore 117543

Philip S. Yu PSYU@US.IBM.COM
IBM T. J. Watson Research Center, New York, Yorktown Heights, NY 10598, USA

Xiaoli Li LIXL@COMP.NUS.EDU.SG
School of Computing, National University of Singapore, Singapore 117543

Abstract

We investigate the following problem: Given a set of documents of a particular topic or class P, and a large set M of mixed documents that contains documents from class P and other types of documents, identify the documents from class P in M. The key feature of this problem is that there is no labeled non-P document, which makes traditional machine learning techniques inapplicable, as they all need labeled documents of both classes. We call this problem *partially supervised classification*. In this paper, we show that this problem can be posed as a constrained optimization problem and that under appropriate conditions, solutions to the constrained optimization problem will give good solutions to the partially supervised classification problem. We present a novel technique to solve the problem and demonstrate the effectiveness of the technique through extensive experimentation.

1. Introduction

Text categorization or classification is the automated assigning of text documents to pre-defined classes. The common approach to building a text classifier is to manually label some set of documents to pre-defined categories or classes, and then use a learning algorithm to produce a classifier. This classifier can then assign classes to future documents based on the words they contain. This approach to building a text classifier is commonly called *supervised learning* because the training documents have been labeled (often manually) with pre-defined classes.

The main bottleneck of building such a classifier is that a large, often prohibitive, number of labeled training documents is needed to build accurate classifiers. Recently, it has been shown in (Nigam et al., 1998) that unlabeled data is helpful in classifier building. Their approach basically uses a small labeled set of documents of every class, and a large set of unlabeled documents to build classifiers. They show that using both labeled and unlabeled documents is better than using the small labeled set alone. This technique alleviates some labor-intensive effort.

In this paper, we study a different problem, which is also common in practice. It aims to identify a particular class of documents from a set of mixed documents, which contain this class of documents and also other kinds of documents. We call the class of documents that we are interested in the *positive documents*. We call the rest of the documents the *negative documents*. This problem can be seen as a classification problem involving two classes, positive and negative. However, we do not have labeled negative documents as they are mixed with positive documents in the mixed set. Traditional techniques require both labeled positive and labeled negative documents to build a classifier. In this research, we aim to build a classifier using only the positive set and the mixed set. Thus, we save on the labor-intensive effort of manual labeling of negative documents.

Finding targeted (positive) documents from a mixed set is an important problem. With the growing volume of online text documents available through the World Wide Web, Internet news feeds, and digital libraries, one often wants to find those documents that are related to one's work or one's interest. For example, one may want to build a repository of machine learning research papers. One can start with an initial set of papers from an ICML proceedings. One can then find those machine learning papers in related online journals or in conference series such as the AAAI/IJCAI conferences. Similarly, given one's the bookmarks (positive documents), one may want those documents that are of interest from Internet sources without labeling any negative documents. In applications, positive documents are usually available because if one has worked on a particular task for some time, one should have accumulated many related documents. Even if no positive document is available initially, finding some such documents from the Web or any other source is relatively simple (e.g., using Yahoo Web directory). One can then use this set to find the same class of documents from any other sources without manual labeling of negative documents from each source.

The key feature of these problems is that there is no labeled negative document, which makes traditional classification methods inapplicable, as they all need labeled documents of every class. We call this problem *partially supervised classification*, as there are only positive documents (which can be considered as labeled with the class positive), but not labeled negative documents. In this paper, we show theoretically that the partially supervised classification problem can be posed as a constrained optimization problem. In particular, we show that for the noiseless case, where there is a fixed target function from a known function class F that maps the features (document keywords and their frequences) to the label of the document, an algorithm that selects a function f from F that correctly classifies all positive documents and minimizes the number of mixed documents classified as positive will have an expected error of no more than ϵ with high probability if both the mixed documents and positive documents are of size $O(\frac{1}{\epsilon}\ln\frac{1}{\epsilon})$. We also generalize the result to the case where there may not be a fixed target function from F that maps from the features to the label of the document, i.e. the labels may be noisy or our function class F may not be powerful enough to contain the target function. In this case, we aim to find a function that performs close to the best possible, subject to a desired expected recall r. The corresponding optimization problem is to minimize the number of mixed documents classified as positive subject to a fraction r of the positive documents being corrected classified.

While the theory indicates that there is enough information available to solve the partially supervised classification problem, the constrained optimization problem required appears to be difficult to solve in practice. In this paper, we propose a novel heuristic technique for solving the partially supervised classification problem in the text domain. Our algorithm is built on the naive Bayesian classifier (McCallum & Nigam, 1998) in conjunction with the EM (Expectation Maximization) algorithm (Dempster et al., 1977). Our algorithm has two main novelties:

- After building an initial classifier (using naive Bayes and the EM algorithm), we select those documents that are most likely to be negative documents from the mixed set (using some "spies"). These documents are used together with the positive documents to reinitialize the EM algorithm in order to obtain a good local maximum of the likelihood function.
- The EM algorithm generates a sequence of solutions that increase the likelihood function. However, the classification error of this sequence of solutions may not necessarily be improving (in fact, performance quite often deteriorates as the likelihood increases). Motivated by the theoretical considerations, we give a heuristic estimate to the probability of error and use the estimate to select a good classifier out of the sequence of classifiers produced by the EM algorithm.

We carried out extensive experiments using 30 public domain document datasets. The results show that the proposed technique is effective and computationally efficient.

2. Related Work

Text classification has been studied extensively in the past in information retrieval, machine learning and data mining. A number of techniques have been proposed, e.g., Rocchio algorithm (Rocchio, 1971), the naive Bayesian method (Lewis & Ringuette, 1994), K-nearest neighbour (Yang, 1999), and support vector machines (Joachims, 1997). These existing techniques, however, all require labeled data for all classes for building the classifier. They are not designed for solving the partially supervised classification. Note that we use the naive Bayesian method as the base in our technique because it has a strong foundation for EM and is more efficient. In addition, the focus of this paper is to demonstrate the potential of partially supervised classification; a similar approach could be applied to more complex classifiers.

A theoretical study of Probably Approximately Correct (PAC) learning from positive and unlabeled examples was done in (Denis, 1998). The study concentrates on the computational complexity of learning and shows that function classes learnable under the statistical queries model (Kearns, 1998) is also learnable from positive and unlabeled examples. (Letouzey et al., 2000) presents an algorithm for learning using a modified C4.5 (decision tree) algorithm based on statistical query model. Recently, learning from positive example was also studied theoretically in (Muggleton, 2001) within a Bayesian framework where the distribution of functions and examples are assumed known. The result obtained in (Muggleton, 2001) is similar to our theoretical result in the noiseless case. However, we also extend it to the more practical noisy case. From a practical point of view, we propose a novel technique to solve the problem in the text domain. This technique produces remarkably good results.

Another line of related work is learning using a small labeled set (Nigam et al., 1998) and (Shahshahani & Landgrebe, 1994). In both works, a small set of labeled data of every class and a large unlabeled set are used for classifier building. It was shown that the unlabeled data helps classification. These works are clearly different from ours as we do not have any labeled document of the negative class.

The proposed method is also different from traditional information retrieval (Salton, 1991). In information retrieval, given a query document and a large document collection, the system retrieves and ranks the documents in the collection according to their similarities to the query document. Web search uses a similar approach. It does not perform document classification, but only produces a ranking of documents according to some similarity measures.

3. Theoretical Foundations for Partially Supervised Classification

This section gives the theoretical foundations of partially supervised classification. We assume that our examples are generated independently according to some fixed distribution D over $\mathcal{X} \times \mathcal{Y}$ where $\mathcal{Y} = \{0, 1\}$. For the text domain, $\mathcal{X}$ is the set of possible documents, $\mathcal{Y}$ is the set of negative and positive classes and the term 'examples' denotes documents with their labels. Let $\Pr_D[A]$ denote the probability of event A when (X, Y) are chosen randomly according to D. For a finite sample T, $\Pr_T[A]$ denote the probability with respect to choosing an example uniformly at random from the set T. We are given two sets of examples, a positive sample P of size n_1 drawn independently from the conditional distribution $D_{X|Y=1}$ conditioned on $Y = 1$ and an unlabeled sample M of size n_2 drawn independently from the marginal distribution D_X on $\mathcal{X}$. A learning algorithm will select a function f from a class of functions $F : \mathcal{X} \mapsto \{0, 1\}$ to be used as a classifier.

To obtain insights into why learning is possible in the partially supervised case, we rewrite the probability of error

$$\begin{aligned}\Pr[f(X) \neq Y] &= \Pr[f(X) = 1 \wedge Y = 0] \\ &\quad + \Pr[f(X) = 0 \wedge Y = 1]. \quad (1)\end{aligned}$$

We have $\Pr[f(X) = 1 \wedge Y = 0] = \Pr[f(X) = 1] - \Pr[f(X) = 1 \wedge Y = 1] = \Pr[f(X) = 1] - (\Pr[Y = 1] - \Pr[f(X) = 0 \wedge Y = 1])$.

Substituting into equation 1, we obtain

$$\begin{aligned}\Pr[f(X) \neq Y] &= \Pr[f(X) = 1] - \Pr[Y = 1] \quad (2) \\ &\quad + 2\Pr[f(X) = 0 | Y = 1] \Pr[Y = 1].\end{aligned}$$

Note that $\Pr[Y = 1]$ is constant. If we are able to hold $\Pr[f(X) = 0|Y = 1]$ small, then learning is approximately the same as minimizing $\Pr[f(X) = 1]$. Holding $\Pr[f(X) = 0|Y = 1]$ small while minimizing $\Pr[f(X) = 1]$ is approximately the same as minimizing $\Pr_M[f(X) = 1]$ while holding $\Pr_P[f(X) = 1] \geq r$ (where r is recall) if the set of positive examples P and the set of unlabeled examples M are large enough.

We will measure the complexity of function classes using the VC-dimension (Vapnik & Chervonenkis, 1971) of the function class. The VC-dimension is a standard measure of complexity in computational learning theory (see e.g. (Anthony & Bartlett, 1999)). For a class of function $F : \mathcal{X} \mapsto \{0, 1\}$ and a finite set T, let $F_{|T}$ be the restriction of F to T (that is, the set of all possible $\{0, 1\}$-valued functions on the domain T that can be obtained from the class F). The VC-dimension of F, $VCdim(F)$ is the size of the largest set of points T in $\mathcal{X}$ such that $|F_{|T}| = 2^{|T|}$. In other words, if T is larger than the VC-dimension of F, there will be at least one of the $2^{|T|}$ possible functions on the domain T that is not in $F_{|T}$.

Obviously, $VCdim(F) \leq \lg |F|$ if F is a finite set of functions. The VC-dimension of the class of thresholded linear functions $\{f(x) = \sigma(w_0 + \sum_{i=1}^{d} w_i x_i) : w_i \in \Re^{d+1}\}$, where $\sigma(x) = 0$ if $x < 0$ and $\sigma(x) = 1$ otherwise, is $d + 1$ (see e.g. (Anthony & Bartlett, 1999)). The class of functions used in Naive Bayes is a subset of linear functions. Hence the VC-dimension of the class is no more than $2m + 1$ when m is the number of words used in the classifier. The bounds obtained in this paper are only meant to illustrate the rate at which the sample size should grow. The constants have not been optimized in any way[1].

3.0.1 Noiseless Case

In the noiseless case, there is a function $f_t \in F$ such that $Y = f_t(X)$ for every (X, Y) drawn from D. When a function correctly classify all the documents in the set of positive documents, we say that it achieves *total recall* on P.

In error minimization, we would like our learning algorithm to produce a function $\hat{f} \in F$ such that $\Pr[E[\hat{f}(X) \neq f_t(X)] > \epsilon] \leq \delta$. We want to show that under the appropriate conditions, selecting the function $\hat{f} \in F$ that minimizes $\sum_{i=1}^{n_2} f(X_i)$ on the unlabeled examples subject to $\hat{f}$ classifying all the positive examples correctly will give a function with small expected error.

In retrieval applications, we are often more concerned with precision and recall than with the error. Define expected recall of f, $\mathrm{ER}(f) = \Pr[f(X) = 1|Y = 1]$ and expected precision of f, $\mathrm{EP}(f) = \Pr[Y = 1|f(X) = 1]$.

The following theorem gives conditions under which the function that minimizes $\sum_{i=1}^{n_2} f(X_i)$ performs well.

Theorem 1 *Let F be a permissible[2] class of functions with VC-dimension d and let $f_t \in F$ be the target function. Let $\nu = E[f_t(X)]$. Let $P = X_1, \ldots, X_{n_1}$ be drawn from the distribution of positive examples $D_{X|Y=1}$ where*

$$n_1 > \frac{16}{\epsilon}\left(d \ln\left(\frac{16e}{\epsilon} \ln \frac{16e}{\epsilon}\right) + \ln \frac{24}{\delta}\right).$$

Let $M = Z_1, \ldots, Z_{n_2}$ be unlabeled examples drawn independently from D_X where

$$n_2 > \frac{16}{\epsilon}\left(d \ln\left(\frac{16e}{\epsilon} \ln \frac{16e}{\epsilon}\right) + \ln \frac{24}{\delta}\right).$$

Let $\hat{F}$ be the subset of F that achieves total recall on P and $\hat{f} = argmin_{f \in \hat{F}} \sum_{i=1}^{n_2} f(Z_i)$. Then, with probability at least $1 - \delta$, $ER(\hat{f}) > 1 - \epsilon$, $EP(\hat{f}) > \frac{(1-\epsilon)\nu}{(1+9\epsilon)\nu + 4\epsilon}$ and $E[\hat{f}(X) \neq f_t(X)] < (10\nu + 4)\epsilon$.

3.0.2 Noisy Case

Unlike the noiseless case, in the noisy case, Y need not be equal to $f_t(X)$ for any $f_t \in F$ when (X, Y) is independently drawn from D. This models the case where the labels are flipped with some probability and also the case where the target function f_t may not belong to F. To deal with these noisy cases, we assume that the user wants to use functions from some function class F to do the classification even though the user does not know whether F contains the target function or whether the labels are noisy. The user is required to specify a target expected recall r. The learning algorithm then tries to output a function $\hat{f} \in F$ such that with high probability the expected recall of $\hat{f}$ is close to r and the expected precision of $\hat{f}$ is close to the best possible among all functions in F that has expected recall close to r. To achieve this, the algorithm draws a set P of positive examples from $D_{X|Y=1}$ and a set M of unlabeled examples from $D_{|X}$ then finds the function that minimizes $\sum_{i=1}^{n_2} f(Z_i)$ on the unlabeled examples subject to the constraint that the fraction of errors on the positive examples P is no more than $1 - r$.

Theorem 2 *Let F be a permissible class of functions with VC-dimension d. Let $P = X_1, \ldots, X_{n_1}$ be drawn from the distribution of positive examples $D_{X|Y=1}$ where*

$$n_1 > \frac{C_1}{\epsilon^2}\left(d \ln\left(\frac{C_2}{\epsilon} \ln \frac{C_2}{\epsilon}\right) + \ln \frac{32}{\delta}\right),$$

$C_1 = 6400$ and $C_2 = 320e$. Let $M = Z_1, \ldots, Z_{n_2}$ be unlabeled examples drawn independently from D_X where

$$n_2 > \frac{C_3}{\epsilon^2}\left(d \ln\left(\frac{C_4}{\epsilon} \ln \frac{C_4}{\epsilon}\right) + \ln \frac{32}{\delta}\right),$$

$C_3 = 1600$ and $C_4 = 160e$. Let $\hat{F}$ be the subset of F that achieves a recall of at least $r - \epsilon/2$ on P, and let $\hat{f} = argmin_{f \in \hat{F}} \sum_{Z_i \in M} f(Z_i)$. Let F^* be the subset of F that achieves an expected recall of at least r, and let $e_0 = \inf_{f \in F^*} \Pr[f(X) = 1 \wedge Y = 0]$. Then with probability at least $1 - \delta$,

$$ER(\hat{f}) > r - \epsilon, \qquad EP(\hat{f}) > \frac{r\nu - \epsilon}{2\nu - r\nu + e_0 + \epsilon}$$

and $E[\hat{f}(X) \neq Y] < 2(1 - r)\nu + e_0 + 2\epsilon$.

[1] All proofs are omitted due to lack of space and can be found in the full version of the paper

[2] This is a measurability condition which need not concern us in practice. See (Haussler, 1992)

4. The Proposed Technique

The previous section showed theoretically that by using positive and mixed document sets, one can build accurate classifiers with high probablity when sufficient documents in P and M are available. However, the suggested theoretical method suffers from two serious practical drawbacks: (1) The constrained optimization problem may not be easy to solve for the function class that we are interested in and (2) Given a practical problem, it does not appear to be easy to choose a desired recall level that will give a good classifier using the function class that we are using. This section proposes a practical heuristic technique based on the naive Bayes classifier and the EM algorithm to perform the task.

4.1 Naive Bayesian Text Classification

Naive Bayesian method is one of the popular techniques for text classification. It has been shown to perform extremely well in practice by many researchers, e.g., (McCallum & Nigam, 1998) (Lewis & Ringuette, 1994).

Given a set of training documents D, each document is considered an ordered list of words. We use $w_{d_i,k}$ to denote the word in position k of document d_i, where each word is from the vocabulary $V = < w_1, w_2, \ldots, w_{|v|} >$. The vocabulary is the set of all words we consider for classification. We also have a set of pre-defined classes, $C = \{c_1, \ldots, c_{|C|}\}$ (in our case, $C = \{c_1, c_2\}$). In order to perform classification, we need to 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, we have

$$\Pr[c_j] = \sum_i \Pr[c_j|d_i]/|D|. \tag{3}$$

and with Laplacian smoothing,

$$\Pr[w_t|c_j] = \frac{1 + \sum_{i=1}^{|D|} N(w_t, d_i) P(c_j|d_i)}{|V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N(w_s, d_i) P(c_j|d_i)}, \tag{4}$$

where $N(w_t, d_i)$ is the count of the number of times the word w_t occurs in document d_i and where $\Pr[c_j|d_i] \in \{0, 1\}$ depends on the class label of the document. Finally, assumming that the probabilities of the words are independent given the class, we obtain

$$\Pr[c_j|d_i] = \frac{\Pr[c_j] \prod_{k=1}^{|d_i|} \Pr[w_{d_i,k}|c_j]}{\sum_{r=1}^{|C|} \Pr[c_r] \prod_{k=1}^{|d_i|} P(w_{d_i,k}|c_r)}. \tag{5}$$

In the naive bayes classifier, the class with the highest $\Pr[c_j|d_i]$ is assigned as the class of the document.

4.2 The EM algorithm

The Expectation-Maximization (EM) algorithm (Dempster et al., 1977) is a popular class of iterative algorithms for maximum likelihood estimation in problems with incomplete data. It is often used to fill the missing values in the data using existing values by computing the expected value for each missing value. The EM algorithm consists of two steps, the *Expectation* step, and the *Maximization* step. The *Expectation* step basically fills in the missing data. The parameters are estimated in the *Maximization* step after the missing data are filled or reconstructed. This leads to the next iteration of the algorithm. For the naive Bayes classifier, the steps used by EM are identical to that used to build the classifier (equations (3) and (4) for the *Expectation* step, and equation (5) for the *Maximization* step). Note that the probability of the class given the document now takes the value in $[0, 1]$ instead of $\{0, 1\}$.

This ability of EM to work with missing data is exactly what we want. We regard all the positive documents to have the class value of c_1. We want to know the class value of each document in the mixed set. EM can help assign a probabilistic class label to each document d_j in the mixed set, i.e., $\Pr[c_1|d_j]$ and $\Pr[c_2|d_j]$. After a number of iterations, all the probabilities will converge. However, a good initialization is important in order to find a good maximum of the likelihood function. Since we have one class of documents, we can initialize the parameters for c_1 using this class. However, we cannot easily assign any probability in the negative class (c_2) because there are no c_2 documents to start with. Below, we propose a technique to deal with this problem by finding the most likely negative documents to be used in initializing EM. The proposed technique consists of two main steps (1) reinitialization and (2) building and selecting the final clasifier.

4.3 Step 1: Reinitialization

4.3.1 Applying the EM algorithm

The EM algorithm can be applied in our context as follows: Initially, we assign each positive document d_i in P the class label of c_1 (i.e., $\Pr[c_1|d_i] = 1$ and $\Pr[c_2|d_i] = 0$), and each document d_j in the mixed set M the class label of c_2 (i.e., $\Pr[c_2|d_j] = 1$ and $\Pr[c_1|d_j] = 0$). Using this initial labelling, a naive Bayesian classifier NB-C can be built. This classifier is then used to classify the documents in the mixed document set M. In particular, NB-C is used to compute a posterior probability of each document in the mixed set (using equation (5)), i.e., $\Pr[c_1|d_j]$, which is assigned to d_j as its new probabilistic class label. The class probability for each positive document remains the same, i.e., $\Pr[c_1|d_i] = 1$ throughout the process.

After every $\Pr[c_1|d_j]$ is revised, a new classifier NB-C is built based on the new $\Pr[c_1|d_j]$ values of the mixed documents, and the positive documents. The next iteration starts. This iterative process goes on until EM converges. The whole algorithm, called I-EM (initial EM), is given in Figure 1. Note that during the process of assigning probabilistic class label to each document d_j in M, we can also compute $\Pr[w_t|c_k]$ and $\Pr[c_k]$, which are sufficient to build a new NB-C as the information computed in the initial process for the positive set P remains the same.

I-EM(M, P)
1. Build an initial naive Bayesian classifier NB-C using the document sets M and P;
2. Loop while classifier parameters change
3. for each document $d_j \in M$
4. Compute $\Pr[c_1|d_j]$ using the current NB-C; // $\Pr[c_2|d_j] = 1 - \Pr[c_1|d_j]$
5. Update $\Pr[w_t|c_1]$ and $\Pr[c_1]$ given the probabilistically assigned class for d_j ($\Pr[c_1|d_j]$) and P (a new NB-C is being built in the process);

Figure 1. The I-EM algorithm with naive Bayesian classifier.

The final probabilistic class label for each document d_j in M can be used to classify the mixed set to identify those positive documents. Our experiment results show that this technique indeed is able to improve classification compared to the technique which simply applies the naive Bayesian technique to the original documents assuming that all the mixed documents have the negative (c_2) class.

This technique performs classification well on "easy" datasets, i.e., positive and negative documents can be easily separated. However, for difficult datasets, significant improvements can still be achieved. The reason that EM does not work well for hard datasets is that our initialization strongly biased towards positive documents. Below, we propose a novel technique to deal with this problem.

4.3.2 INTRODUCING SPY DOCUMENTS INTO THE MIXED SET

To solve the problem discussed above, we need an initialization that balance both the positive and negative documents. Since we do not know which documents are negative, we have to identify some very likely negative documents from the mixed set.

As discussed in Section 4.3.1, the I-EM algorithm helps to separate positive and negative documents. After EM, we are in a good position to identify those most likely negative documents from the mixed set. The issue is how to obtain reliable information for the identification. We do so by sending "spy" documents from the positive set P to the mixed set M. This idea was proven to be crucial. This approach randomly selects $s\%$ of the documents from the positive set P (in our experiment, we use 10%). These documents are the spies, denoted by S. S is added to the mixed set M. The spies behave identically to the unknown positive documents in M and hence allows us to reliably infer the behaviour of the unknown positive documents.

The I-EM algorithm is still utilized, but the mixed set now has some spy documents. After the EM algorithm completes, the probabilistic labels of the spies are used to decide which documents are most likely to be negative. A threshold t is employed to make the decision, which will be discussed below. Those documents in M with lower probabilities ($Pr[c_1|d_j]$) than t are the most likely negative documents, denoted by N. Those documents in M (spies are not included) that have higher probabilities than t become unlabeled documents denoted by U. The detailed algorithm for identifying those most likely negative documents N and also the unlabeled set is given in Figure 2.

Algorithm Step-1
1. $N = U = \phi$;
2. $S = sample(P, s\%)$;
3. $MS = M \cup S$;
4. $P = P - S$;
5. Assign every document d_i in P the class c_1;
6. Assign every document d_j in MS the class c_2;
7. Run I-EM(MS, P);
8. Classify each document d_j in MS;
9. Determine the probability threshold t using S;
10. for each document d_j in M
11.   if its probability $\Pr[c_1|d_j] < t$
12. $N = N \cup \{d_j\}$;
13.   else $U = U \cup \{d_j\}$;

Figure 2. Identifying likely negative documents.

We now discuss how to determine the threshold t. Let the set of spies S be $\{s_1, s_2, \ldots, s_k\}$, and the probabilistic label assigned to each s_i be $\Pr[c_1|s_i]$. Intuitively, we can use the minimum probability of S as the threshold value t, i.e., $t = \min\{\Pr[c_1|s_1], \Pr[c_1|s_2], \ldots, \Pr[c_1|s_k]\}$, which means that we want to retrieve all the spy documents. In a noiseless domain, using the minimum probability is acceptable. However, most real-life document collections have outliers and noise. Using the minimum probability is unreliable. The reason is that the posterior probability $\Pr[c_1|s_i]$ of a outlier document s_i in S could be 0 or much smaller than most (or even all) actual negative documents. However, we do not know the noise level of the data. We can estimate it by trying a few noise levels and selecting the best. We first sort the documents in S according to their $\Pr[c_1|s_i]$. We then use a selected noise level l to decide t: we select t such that $l\%$ of documents have probability less than t. We experimented with a few noise levels, e.g., $l = 5$, 10, 15 and 20. It turns out that it does not make much difference if we choose 5, 10, 15 or 20. The reason will become clear later after discussing Step 2. In our system we use 15%.

In summary, the objective of this first step of the proposed algorithm is to achieve the results given in Figure 3. The left-hand-side shows the initial situation. In the mixed set, we have both positive and negative documents. However, we do not know which documents are positive or negative. Spies from positive set are added to the mixed set. The right-hand-side shows the result that our technique achieves with the help of spies. We observe that most positive documents in the mixed set are put in the unlabeled set, and most negative documents are put in the likely negative set N. The purity of N is much higher than the mixed set.

4.4 Step 2: Building the final classifier using the document sets P, N and U

The step builds the final classifier. The EM algorithm is again employed, with the document sets, P, N, and U. This step is carried out as follows:

1. Put all the spy documents S back to the positive set P.
2. Assign every document in the positive set P the fixed class label c_1, $\Pr[c_1|d_i] = 1$, which will not change in each iteration of EM.
3. Assign each document d_j in the likely negative set N the initial class c_2, i.e., $\Pr[c_2|d_j] = 1$, which changes with each iteration of EM.
4. Each document d_k in the unlabeled set U is not assigned any label initially. At the end of the first iteration of EM, it will be assigned a probabilistic label, $\Pr[c_1|d_k]$. In subsequent iterations, the set U will participate in EM with its newly assigned probabilistic classes, e.g., ($\Pr[c_1|d_k]$).
5. Run the EM algorithm using the document sets P, N and U until it converges.

When EM stops, the final classifier is produced. We call the two-step EM procedure S-EM (spy EM).

We now turn to discuss why the percentage used for selecting the most likely negative documents N does not matter as long as it is in a reasonable range from $5\% - 20\%$. This is because in Step 2 (S-EM) of our algorithm, the probabilities of both document sets U and N are allowed to change. If there are too many positive documents in N, EM will

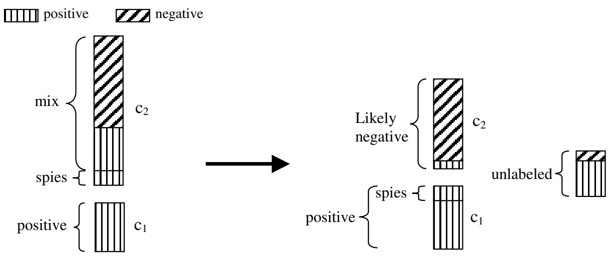

Figure 3. Before and after Step 1: Re-initialization

slowly correct the situation, i.e., moving them to the positive side. In our experiments, 15% is used, which worked very well. We also experimented with 5%, 10% and 20%. The classification results are similar.

4.5 Selecting a classifier

The EM algorithm works well when the local maximum that the classifier is approaching separates the positive and negative documents. This may not always be true. For example, in many situations, the positive and negative documents each consists of many clusters and the clusters may not be separable. In such a situation, it may be better to stop at the naive Bayesian classifier instead of iterating with EM. In general, each iteration of the S-EM algorithm gives a classifier that may potentially be a better classifier than the classifier produced at convergence. So, if we run EM for n iterations, we have a set of n classifier from which we can choose one. We try to estimate the classification error by using an approximation to equation (2).

We note from equation (2) that, on an unlabeled sample M, if we knew $\Pr_M[Y = c_1]$ and $\Pr_M[f(X) = c_2|Y = c_1]$, we will be able to calculate the probability of error since we can measure $\Pr_M[f(X) = c_1]$. Furthermore, we can estimate $\Pr_M[f(X) = c_2|Y = c_1]$ from the corresponding measurement on the positive set P. To get an estimate of the probability of error, we only require a good estimate of $\Pr_M[Y = c_1]$.

In this paper, we choose to use an estimate of the change in the probability of error in order to decide which iteration of EM to choose as the final classifier. From equation (2), the change in the probability of error from iteration i to $i+1$ is $\Pr_M[f_{i+1}(X) = c_1] - \Pr_M[f_i(X) = c_1] + 2(\Pr_M[f_{i+1}(X) = c_2|Y = c_1] - \Pr_M[f_i(X) = c_2|Y = c_1])\Pr_M[Y = c_1]$. We use $\Pr_P[f_j(X) = c_2|Y = c_1]$ as an approximation to $\Pr_M[f_j(X) = c_2|Y = c_1]$ for $j = i$ and $j = i+1$, and use $\Pr_M[f_i(X) = c_1]$ as an approximation to $\Pr_M[Y = c_1]$ to obtain

$$\begin{aligned}\Delta_i \quad = \quad & \Pr_M[f_{i+1}(X) = c_1] - \Pr_M[f_i(X) = c_1] \\ & +2(\Pr_P[f_{i+1}(X) = c_2|Y = c_1] \\ & -\Pr_P[f_i(X) = c_2|Y = c_1])\Pr_M[f_i(X) = c_1].\end{aligned}$$

We select iteration i as our final classifier the first time that Δ_i becomes positive. Here, Δ_i is an approximation for the error difference.

For this estimate to work well, we require $\Pr_M[f_i(X) = c_1]$ to be a reasonably good approximation to $\Pr_M[Y = c_1]$. Our reinitialization of the EM algorithm is usually able to provide us with a starting point near a good local minimum where the estimate is not too far wrong. Without reinitializing, we find that using Δ_i as a selection criteria is not very effective.

5. Empirical Evaluation

5.1 Evaluation measures

Since our task is to identify or retrieve positive documents from the mixed set, it is appropriate to use information retrieval measures for our purpose. Two popular measures are the F score and breakeven point. F score is defined as, $F = 2pr/(p + r)$, where p is the precision and r is the recall. F score measures the performance of a system on a particular class (see (Bollmann & Cherniavsky, 1981) (Shaw, 1986) for its theoretical bases and practical advantages). The breakeven point is the value at which recall and precision are equal (Lewis & Ringuette, 1994). However, the breakeven point measure is not suitable for our task as it only evaluates the sorting order of class probabilities of documents. It does not give a good indication of classification performance. The F value on the other hand reflects an average effect of both precision and recall. When either of them is small, the F value will be small. Only when both of them are large, will the F value be large. This is suitable for our purpose, as we want to identify positive documents, and it is undesirable to have either too small a precision or too small a recall. In our experiment results, we will also report the accuracy results.

5.2 Experiment datasets

Our experiments used two large document corpora, from which we created 30 datasets. The first one is the 20 Newsgroups (Lang, 1995). It contains 20 different UseNet discussion groups, which are also categorized into 4 main categories, computer, recreation, science, and talk. We remove all the UseNet headers (thereby discarding the subject line) in our experiments. Since our task is to identify positive documents from a set of mixed documents, we choose many categories as the positive classes, and then using various individual categories, or combinations of categories to form the negative class documents. Another collection that we used is WebKB (Nigam et al., 1998). WebKB consists of Web pages from a number of university computer science departments. The pages are divided into 7 categories: student, faculty, course, project, staff, dept and other. In this work, we used the non-other categories: student, faculty, cours, project, staff and dept. For our experiment purposes, all the html tags are removed.

Our objective is to recover those positive documents put into the mixed set M. Note that we do not need separate test sets as in normal classification. M can be seen as the test set. For each experiment, we divide the full positive set into two subsets, P and R, where P is the positive document set used in our algorithm, which has $a\%$ of the full positive set, and R is the set of remaining documents. $b\%$ of the documents in R is then put into the negative set to form the mixed set M. We do not put all the documents in R into M as we wish to create the realistic situations of skewed datasets. We believe that in most realistic situations, the mixed set M can be very large, but the number of positive documents in M is often small. We vary a and b to create different settings for our experiments.

	positive	negative	pos size	M size	pos in M	NB(F)	NB(A)	I-EM8(F)	I-EM8(A)	S-EM(F)	S-EM(A)
1	hardpc	hardmac	1000	14000	2000	52.78	90.78	82.68	95.41	83.16	95.67
2	graphic	hardmac	200	1400	400	14.87	73.56	41.11	78.47	75.00	85.99
3	stud	course	200	1400	400	21.32	74.78	53.65	81.89	75.91	88.94
4	proj	course	328	1586	656	64.84	78.46	96.51	97.16	95.73	96.61
5	facu	course	100	1132	202	41.01	86.67	90.83	96.92	85.03	95.36
6	rec	compu	224	1380	450	55.02	80.90	82.88	91.86	83.99	91.51
7	hockey	baseball	800	6600	1600	89.60	95.43	96.82	98.44	96.39	98.30
8	os-win	wind.x	200	1400	400	45.64	79.92	92.20	95.82	91.19	95.39
9	project	student	200	1400	400	12.53	73.30	49.62	82.70	67.68	86.00
10	faculty	student	100	1843	202	4.26	89.30	12.73	89.79	41.09	91.96
11	electr	space	224	2091	450	15.00	80.16	56.21	86.08	62.71	85.95
12	med	electr	200	1400	400	47.24	80.27	92.41	95.86	91.25	95.40
13	rec	sci	200	1400	400	39.02	78.37	87.27	93.50	88.94	94.30
14	compu	sci	800	5600	1600	78.76	89.95	92.66	96.05	91.93	95.72
15	politics	rec	1000	6000	2000	28.82	72.59	29.65	73.11	77.39	87.77
16	sci	talk	800	5600	1600	68.89	86.45	96.90	98.26	97.44	98.55
17	hardmac	os-win	800	5600	1600	74.16	88.17	94.37	96.81	94.32	96.86
18	atheism	rel.misc	200	1400	400	83.61	90.63	44.82	29.90	85.60	90.57
19	rel.misc	pol.misc	200	1400	400	30.76	75.74	65.43	82.67	66.05	82.83
20	pol.guns	pol.misc	200	1400	400	48.26	79.86	75.38	87.30	73.67	86.54
21	comp	noncomp	200	1400	400	26.19	75.41	70.39	86.17	76.12	86.63
22	windows	nonwin	200	4400	400	42.31	93.25	73.43	94.10	63.67	94.73
23	sci	nonsci	800	14600	1600	50.17	92.58	82.85	95.90	77.16	95.27
24	rec	nonrec	800	14600	1600	79.24	96.20	91.40	98.16	88.70	97.75
25	talk	nontalk	800	14600	1600	80.88	96.43	78.11	94.01	93.44	98.58
26	hdibm.pc	nonibm	200	4400	400	20.31	91.72	35.96	70.04	49.21	85.64
27	os	nonos	200	4400	400	4.93	91.09	9.50	91.23	22.31	91.91
28	hdmac.pc	nonmac	200	4400	400	25.22	92.09	50.24	81.38	62.16	94.38
29	graphic	nongrap	200	4400	400	32.46	92.47	57.19	89.60	61.16	94.32
30	stufac	nonstufac	579	2911	1159	39.81	69.15	74.09	77.64	79.86	85.31
	Average		405	4471	811	43.93	84.52	68.58	87.54	76.61	92.16

Table 1. Results for datasets with $a = 20\%$ and $b = 50\%$.

5.3 Experiment results

Three techniques are tested in our experiments:

Naive Bayesian classifier (NB): The Bayesian technique is directly applied to P (as c_1) and M (as c_2) to produce a classifier, which is then used to classify documents in M.

I-EM: This applies the EM algorithm to P and M until it converges (no spies in M). The final classifier is then used to classify M to identify the positive documents in M.

S-EM: Spies are used to re-initialize EM to build the final classifier. We also use the error estimate to select a good classifier from the sequence of classifiers produced by EM after reinitialization. We use 10% of the positive documents as spies in all experiments. The threshold t for choosing likely negative documents is 15%.

The experiment results for 4 different settings of a and b are shown in Tables 1 and 2. Table 1 shows the full results of the 30 datasets with $a = 20\%$ and $b = 50\%$. Descriptions of the datasets are given in the full version of the paper. From row 1 to 18, each negative document set consists of documents from a single category. From row 19 to row 30, each negative set contains documents of multiple categories. In the table, Column 1 gives the dataset number. Columns 2 and 3 give the names of each positive and negative document sets respectively. Column 4 gives the total number of positive documents in P. Column 5 gives the total number of documents in M. Column 6 gives the number of positive documents in M. Columns 7 and 8 give the F value and accuracy (A) of NB on the mixed set of each dataset. Note that the F value only measures retrieval results of the positive documents in M, while the accuracy measures the whole set M. Columns 9 and 10 show the F value and the accuracy of I-EM. We run I-EM with 8 iterations, denoted I-EM8 (I-EM does not improve after that). Columns 11 and 12 give the corresponding results of S-EM. In our experiments, we found that it is not necessary to run EM (with spies) to convergence before extracting likely negative documents N for reinitializing. Only 2 iterations of EM are sufficient. We also found that running more than 4 iteration of S-EM after reinitializing does not significantly improve the final results. Thus, we use 2 iteration of EM before reinitialization and 4 iterations of EM after reinitialization to save computation (our method is thus efficient as it only scans the data 6 times). The final row of the table gives the average results of each column. All the results are the average values of 5 random runs. Due to space limitations, Table 2 summarize the average results over the 30 datasets for the other three settings of a and b.

From the 2 tables, we observe that S-EM outperforms the other two methods dramatically with F values. The accuracy is also much better than I-EM8, and is slightly better than NB. However, since our datasets are highly skewed (the positive sets are very small), accuracy does not reflect the classification performance well. We note that NB performs quite well when the fraction of positive documents in M is small (row 2 of table 2). This is not surprising because when we assume the documents in M are all negative we are mostly correct and the naive Bayes method is able to tolerate a small amount of noise in the training data. However, we note that in many practical situations, items from the positive class can form a significant fraction of the items in the environment. For example, if the positive class is the class of movies we may enjoy watching, we would expect these movies to form a significant fraction of all movies.

A natural question to ask is whether the steps: (1) reinitial-

settings	pos size	M size	pos in M	NB(F)	NB(A)	I-EM8(F)	I-EM8(A)	S-EM(F)	S-EM(A)
$a = 20\%$ and $b = 20\%$	405	3985	324	60.66	94.41	68.08	91.96	76.93	95.96
$a = 50\%$ and $b = 20\%$	1013	3863	203	72.09	95.94	63.63	86.81	73.61	95.28
$a = 50\%$ and $b = 50\%$	1013	4167	507	73.81	93.12	71.25	85.79	81.85	94.32

Table 2. Summary of the other results.

	20-20%(F)	20-20%(A)	20-50%(F)	20-50%(A)	50-20%(F)	50-20%(A)	50-50%(F)	50-50%(A)
I-EMbest	72.35	95.57	71.41	90.85	70.77	93.70	77.62	92.64
I-EM8	68.08	91.96	68.58	87.54	63.63	86.81	71.25	85.79
S-EM4	72.74	91.89	76.37	89.17	65.86	88.74	75.04	88.32
S-EM	76.93	95.96	76.61	92.16	73.61	95.28	81.85	94.32

Table 3. The F score and accuracy (A) results for different datasets indicating the necessity of both reinitialization and selection of classifier. The column descriptor a-b corresponds to the method of constructing the datasets described in Section 5.2.

izing and (2) selecting a good model instead of using the converged model are both necessary for good performance. For comparison, we obtained the best results of I-EM without reinitialization from all 8 iterations where the F value is used to decide the best result. This result is not achievable in practice as there is no way to decide the best result without actual negative data. As we list in Table 3, S-EM outperforms even this omniscient version of I-EM showing that the reinitialization is essential for the improved performance. We also show that reinitialization does not account for all the improvement by calculating the performance of the S-EM without selecting a good model (S-EM4, which is the 4th iteration of EM after reinitialization). As shown in Table 3, selecting a good model significantly outperforms simply taking the final iteration of the reinitialized version of EM. Note that our classifier selection method could not be applied to I-EM because the estimates can be very inaccurate since we initially assign all documents in M the class c_2 (negative).

Each iteration of the proposed technique in Step 1 of the proposed technique is linear in the number of documents in $M \cup P$. Since we find that 2 EM iterations are sufficient for Step 1, the complexity of this step is only $O(|M \cup P|)$. The second step has the same complexity as only 4 aditional EM iterations are needed. Thus, the whole technique is linear to the number of documents in $M \cup P$. Only 6 scans of the two document sets are required to build the final classifier.

6. Conclusions

This paper studied the problem of classification with only partial information, one class of labeled (positive) documents, and a set of mixed documents. We show theoretically that there is enough information in the positive and unlabelled data to build accurate classifiers. We proposed a novel technique to solve the problem in the text domain. Our algorithm utilizes the EM algorithm with the naive Bayesian classification method. We reinitialize the EM algorithm after a few runs by using the positive documents and the most likely negative documents from the mixed set. We then use an estimate of the classification error in order to select a good classifier from the classifiers produced by the iterations of the EM algorithm. Extensive experiments show that the proposed technique produces extremely accurate classifiers given that only the positive class is known.

7. Acknowledgements

Bing Liu, Xiaoli Li and Wee Sun Lee acknowledge the support of A-STAR and NUS Academic Research Fund grants R252–000–041–112, R252–000–041–303 and R252–000–070–107.

References

Anthony, M., & Bartlett, P. (1999). *Neural network learning: Theoretical foundations*. Cambridge University Press.

Bollmann, P., & Cherniavsky, V. (1981). Measurement-theoretical investigation of the mz-metric. *Information Retrieval Research* (pp. 256–267).

Dempster, A., Laird, N. M., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, *39*, 1–38.

Denis, F. (1998). PAC learning from positive statistical queries. *Proc. 9th International Conference on Algorithmic Learning Theory - ALT '98* (pp. 112–126). Springer-Verlag.

Haussler, D. (1992). Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Inform. Comput.*, *100*, 78–150.

Joachims, T. (1997). Text categorization with support vector machines: learning with many relevant features. *Technical Report LS8-Report*.

Kearns, M. (1998). Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, *45*, 983–1006.

Lang, K. (1995). Newsweeder: Learning to filter netnews. *International Conference on Machine Learning* (pp. 331–339).

Letouzey, F., Denis, F., Gilleron, R., & Grappa, E. (2000). Learning from positive and unlabeled examples. *ALT-2000*.

Lewis, D., & Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. *Third annual symposium on document analysis and information retrieval* (pp. 81–93).

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive bayes text classification.

Muggleton, S. (2001). Learning from positive data. *Machine learning, to appear*.

Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. (1998). Learning to classify text from labeled and unlabeled documents. *AAAI-98* (pp. 792–799). Madison, US: AAAI Press, Menlo Park, US.

Rocchio, J. (1971). *Relevance feedback in information retrieval. in g. salton (ed.). the smart retrieval system - experiments in automatic document processing*. Englewood Cliffs, NJ.

Salton, G. (1991). Developments in automatic relevance feedback in information retrieval. *Science*, *253*, 974–979.

Shahshahani, B., & Landdgrebe, D. (1994). The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon. *IEEE Trans. on Geoscience and Remote Sensing*, *52*, 1087–1095.

Shaw, J. (1986). On the foundation of evaluation. *American society for information science*, *37*, 346–348.

Vapnik, V. N., & Chervonenkis, A. Y. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probabability and its Applications*, *16*, 264–280.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*, *1*, 67–88.