Clustering-based Method for Positive and Unlabeled Text Categorization Enhanced by Improved TFIDF

LU LIU[1,2] AND TAO PENG[1,2]
[1]*Department of Computer Science*
University of Illinois at Urbana-Champaign
Urbana, 61801 USA
[2]*College of Computer Science and Technology*
Jilin University
Changchun, 130012 P.R. China

PU learning occurs frequently in Web pages classification and text retrieval applications because users may be interested in information on the same topic. Collecting reliable negative examples is a key step in PU (Positive and Unlabeled) text classification, which solves a key problem in machine learning when no labeled negative examples are available in the training set or negative examples are difficult to collect. Thus, this paper presents a novel clustering-based method for collecting reliable negative examples (C-CRNE). Different from traditional methods, we remove as many probable positive examples from unlabeled set as possible, which results that more reliable negative examples are found out. During the process of building classifier, a novel TFIDF-improved feature weighting approach, which reflects the importance of the term in the positive and negative training examples respectively, is presented to describe documents in the Vector Space Model. We also build a weighted voting classifier by iteratively applying the SVM algorithm and implement OCS (One-class SVM), PEBL (Positive Example Based Learning) and 1-DNFII (Constrained 1-DNF) methods used for comparison. Experimental results on three real-world datasets (Reuters Corpus Volume 1(RCV1), Reuters-21578 and 20 Newsgroups) show that our proposed C-CRNE extracts more reliable negative examples than the baseline algorithms with very low error rates. And our classifier outperforms other state-of-art classification methods from the perspective of traditional performance metrics.

Keywords: text classification, reliable negative examples, clustering, C-CRNE, WVC

1. INTRODUCTION

With the rapid growth of Web information and the advent of information retrieval from the WWW, text classification, which organizes and classifies a huge amount of information into some given topics, has become an important task in data mining [1]. Text classification can predict the category of a document and has many useful applications in real-world such as filtering spam [2,3], Web page categorization [4], personalized news [5], user's opinion analysis [6,7] and so on. Using a learning algorithm and a series of pre-labeled data which have been categorized into some given classes, is a common approach to obtain a classifier. One problem with classification is that it needs a number of labeled training data to build the accurate classifier. However,

Received March 29, 2013
Communicated by Tao Peng.
Email:taopeng@illinois.edu

manually labeling documents is quite difficult and time-consuming work. Therefore, this paper describes how to use a small set of labeled documents and a large set of unlabeled documents, that is, PU-learning, to build an accurate classifier. That is, the primary challenge of text classification problem is that no labeled reliable negative examples are available in the training data set because (1) negative training examples must uniformly represent the universal set excluding the positive class, and (2) manually collected negative training examples could be biased because of human's unintentional prejudice, which could be detrimental to classification accuracy. PU learning process can uses a small set of labeled examples of every class, and a large set of unlabeled examples to build an accurate classifier without labeling any negative examples. In addition, we mainly delve into how to collect as many reliable negative examples as possible and make PU classifier reach a high performance.

Generally, the method for PU text classification contains two steps. The first step is collecting reliable negative examples (also called *RN*) from unlabeled dataset (identified by *U*). The second step is iteratively training classifier using positive examples (identified by *P*), reliable negative examples and unlabeled examples. The documents, that users are interested in, are called positive examples. Likewise, the rest of the documents, that users are not interested in, are called negative examples, or reliable negative examples (*RN*) [21]. What we need to do is to build a classifier using *P* and *U* to identify the positive or negative documents in *U*. That is to say, the problem of accurately classifying the documents in *U* or test datasets is called PU learning. Many researchers delved into PU learning and proposed a number of approaches such as S-EM algorithm [8], PEBL [9], Roc-SVM [10], Biased-SVMs [11], Weighted Logistic Regression [12], one-class SVMs [13], PSOC [14], PE-PUC [15], SPUL [16] Weighted Voting Classifier (WVC) [14] and so on. Besides, some techniques, for example, active learning and transfer learning [17,18,19,20], have also been applied in PU-learning and achieved good performance. In the process of two-step building classifier, the number of extracted reliable negative examples and the error rate greatly impact the performance of classifier. So far, many methods for collecting reliable negative examples have also been proposed such as spy-technique [21], Cosine Rocchio [22], 1-DNF [9], 1-DNFII [14], NB technique [23] and so on.

During the process of building classifier, a TFIDF-improved approach is presented and adopted, also called TFIPNDF (Term Frequency Inverse Positive-Negative Document Frequency), for weighting the terms in the positive and negative training example set, respectively. According to TFIPNDF, during the process of training classifier, a term plays different roles in training set. That is, in positive training set, the more documents a positive feature appears, the more significant it is estimated in positive training set. Also, our method has higher requirements on the quality of the training set, especially the negative data set.

In this paper, we mainly present a novel method for the first step, that is, a clustering-based method for collecting reliable negative examples. Intuitively, reliable negative examples are not relevant to the given topic. In other words, they should share as few features with positive examples as possible. Clustering is a common technique used in data analysis. It can also cluster documents that are similar to each other. Motivated by this, we first cluster all documents in positive example set into some small clusters. Then, these small clusters are applied to cluster with unlabeled examples using a

predefined cluster-radius r_P. The goal of clustering documents in P is to improve the efficiency of clustering unlabeled examples. Finally, the documents, which are not clustered in U, are treated as the reliable negative examples. Experimental results show that the proposed clustering-based method for extracting reliable negative examples can extract more reliable negative examples than the existing methods. The classification results are better than the existing state-of-art PU classification methods.

The rest of this paper is organized as follows. We first briefly review the related work in Section 2. Then Section 3 describes our novel clustering-based method for collecting reliable negative examples in Section 3. Section 4 details how to build our classifier. Several comprehensive experiments are performed to evaluate the effectiveness and efficiency of our proposed method in Section 5 and results in which experiment settings, performance metrics, datasets, and results are all provided. Finally we draw the conclusions.

2. RELATED WORKS

Due to the two-step algorithm in PU learning, we introduce the required background and the related efforts on collecting reliable negative examples and building a classifier, respectively. So far, many PU methods have been proposed to tackle the problem in assigning text documents in their corresponding categories and lots of applications related to PU learning have been applied in the real world. The PU algorithms contain PAC learning [24], Bayes [25], S-EM [8], PEBL [9], Roc-SVM [10], Weighted Logistic Regression [12], Biased SVMs [11], one-class SVM [26], PSOC [14] and so on.

Before classifying text documents, some preprocessing steps such as removing stopwords, stemming, feature selection and term weighting need to be done first. Boolean weighting, word frequency weighting, *tf-idf* weighting, *tfc* weighting, *ltc* weighting and entropy weighting are some main term weighting methods [27]. The spy technique [21] used in S-EM picked a set of positive documents from P randomly and added them into U. NB technique [11] collected reliable negative examples from U using Naive Bayes classifier. The Rocchio-SVM method [22] identified reliable negative examples using *Rocchio* algorithm, which was traditional and did not perform well. It built classifier by constructing a prototype vector. 1-DNF [9] was another conventional method for collecting *RN*. The number of collected *RN* was always small so that the performance was usually poor. Compared with 1-DNF, 1-DNFII [14] improved the number of *RN* a lot. Zhang et al. [28] adopted KNN algorithm to rank the similarity of unlabeled examples from the k nearest positive examples. They also set a threshold to label examples in U. The unlabeled examples whose similarities were lower than the threshold were regarded as *RN*. Zhang et al. [29] also introduced a method for collecting *RN* based on clustering. They clustered positive and unlabeled examples (CPUE) at the same time to identify the reliable negative examples.

Liu [21] put forward S-EM algorithm which combined Bayes and EM algorithm together to build the classifier. Besides, Liu also summarized PU classification. During the process of building classifier, he tried to maximize the number of negative documents in unlabeled examples. However, meanwhile, the negative document set

would include a number of positive documents, which resulted in reducing the accuracy of classifier. Yu et al. [9] introduced PEBL framework for Web page classification. However, the number of generated reliable negative examples was too few to guarantee that the classifier was the best. Li et al. [10] combined the *Rocchio* method and SVMs for classifier building. Because *Rocchio* method did not reach a high accuracy, the accuracy of collecting reliable negative examples was not high, either. Liu et al. [11] also presented Biased-SVM which regarded positive examples as *P* and treated unlabeled examples as *RN*. The generated classifier did not reach a high precision. Manevitz et al. [13] implemented versions of SVM appropriate for one-class classification for information retrieval. Pan et al. [30] proposed a Dynamic Classifier Ensemble method for Positive and Unlabeled text stream (DCEPU) classification scenarios. They addressed the problem of classifying positive and unlabeled text stream with various concept drift by constructing an appropriate validation set and designing a novel dynamic weighting scheme in the classification phase.

Some researchers dealt with the problem that the amount of positive examples is low. Yu et al. [31] combined graph-based semi-supervised learning with the two-step method aiming at solving the PU-learning problem with a small set of positive examples. Nagy et al. [32] proposed a modified approach for text classification from positive and unlabeled examples. They applied Rocchio algorithm to extract reliable negative examples. They also improved the iterations when building the classifier by adding the positive examples identified from unlabeled example set into positive set *P*. Lu et al. [33] presented a refined method to do the PU text classification combining Rocchio and K-means algorithm. They considered the positive example set may be less than 5%. Li et al. [34] also proposed a novel technique LPLP (Learning from Probabilistically Labeled Positive examples) which aimed at dealing with the situation that positive example set was small. Qiu et al. [35] proposed a new text classification approach based on a keyword and Wikipedia knowledge in order to avoid labeling document manually.

3. CLUSTERING-BASED METHOD FOR COLLECTING RELIABLE NEGATIVE EXAMPLES (C-CRNE)

In this section, we present a novel method for collecting reliable negative examples from set *U*. Intuitively, reliable negative examples are not relevant to the given topic. In other words, they should share as few features with positive examples as possible. Clustering is a traditional technique to organize data and it can cluster documents into subsets that are very related. The data or documents in the same subset can be considered that they are highly relevant. So, clustering is implemented into our method to collect more reliable negative examples with high accuracy. The positive examples are clustered first using hierarchical agglomerative clustering. To cluster examples in set *P* and *U*, we define class center O_P, cluster-radius r_P in Eqs.1 and 2 as follows.

$$O_P = \frac{\sum_{i=1}^{m} x_i}{m} \tag{1}$$

$$r_P = \frac{r \times \varphi(m)}{\varphi(m)+1}, \quad r = \max_{x_k \in P} d(x_k, O_P) \tag{2}$$

where we assume the positive example set be $X=\{x_1, x_2, .., x_m\}$, $x_i \in P$. x_i is a vector of document i in P. m is the total number of documents that are applied to cluster. $\varphi(m) = \lg(m)+1$. $d(x_k,O_p)$ represents the distance between document k and the class center O_p. r is the maximum among those distances.

The whole algorithm contains two steps: clustering step and selecting step. In the process of clustering step, we first initialize class center and cluster radius according to Eqs. 1 and 2. As mentioned above, our algorithm proceeds using agglomerative clustering which starts with each document (an input) as a separate cluster. It takes m vectors $x_1, x_2,\ldots,x_m$ (use positive example set $X=\{x_1,x_2,\ldots x_m\}$ in algorithm 1 below), representing the input instances, and variable *numCluster* as the number of generated classes by now. Also, $C_1, C_2,\ldots, C_{numCluster}$ are the final classes. Therefore, the algorithm begins with $C_1=\{x_1\}$, $O_1=x_1$, *numCluster*=1 and the rest vectors are left in set Z as algorithm 1 describes below. For each document in P, we find out the closest class center O_j and calculate the distance between O_j and the document. If $d(x_i, O_j)< r_P$, then it means that document i (represented by vector x_i) in P can be added in class C_j and the class center of C_j also needs to be adjusted using Eq. 3 below.

$$O_j \leftarrow \frac{n_j \cdot O_j + x_i}{n_j + 1}, n_j \leftarrow n_j + 1 \tag{3}$$

where n_j is the number of elements in C_j.

If $d(x_i, O_j) \geq r_P$, then it means that document i should be added in a new class $C_{numCluster}$. According to the distance, the document is either categorized into the existing class or added into a new class. The clustering step proceeds until the positive example set is empty. In the process of selecting step, the distance between each document in U and each cluster of P is calculated. If the distance between a document and a class center is less than radius r, then the document is removed from U. The rest documents in U are regarded as reliable negative examples. Different from traditional methods, C-CRNE does not extract *RN* directly. It removes as many probable positive examples as possible. The algorithm is described as follows.

Algorithm of Clustering-based Method for Collecting Reliable Negative Examples
Input: P: positive document set, U: unlabeled document set
Output: *RN*: reliable negative document set

1. **procedure Clustering-based Method for Collecting Reliable Negative Examples**
2. Assume the positive example set be $X=\{x_1, x_2, .., x_m\}$, $x_i \in P$
3. $O_P \leftarrow \frac{\sum_{i=1}^{m} x_i}{m}$, $r \leftarrow \max_{x_k \in P} d(x_k, O_P)$, $r_P \leftarrow \frac{r \times \varphi(m)}{\varphi(m)+1}$
4. $C_1 \leftarrow \{x_1\}, O_1 \leftarrow x_1, numCluster \leftarrow 1, Z \leftarrow \{x_2, x_3, \ldots, x_m\}, n_j \leftarrow 0$
5. **repeat**
6. **select** one $x_i \in z$ and **find out** the closest O_j to x_i, $O_j \leftarrow \arg\min_{j}^{numCluster}{}_{k=1} d(x_i, O_k)$

7. **if** $d(x_i, O_j) < r_P$ **then**
8. **add** x_i to class c_j and adjust the center of class j, $O_j \leftarrow \frac{n_j \cdot O_j + x_i}{n_j + 1}$ and $n_j \leftarrow n_j + 1$
9. **else**
10. **add** a new class $C_{numCluster} \leftarrow \{x_i\}$, $numCluster \leftarrow numCluster + 1$, $O_{numCluster} \leftarrow x_i$
11. **end if**
12. **until** Z is empty
13. $RN \leftarrow U$
14. **for** $i \leftarrow 1$ to $numCluster$
15. **for** each $x_i \in RN$
16. **if** $d(x_i, O_i) < r$ **then**
17. $RN \leftarrow RN - \{x\}$
18. **end if**
19. **end for**
20. **end for**
21. **end procedure**

Algorithm 1. Clustering-based Method for Collecting Reliable Negative Examples.

$\varphi(m) = \lg(m) + 1$. *numCluster* represents the number of generated classes by now. m is the total number of documents that are applied to cluster. C_1, C_2, …, $C_{numCluster}$ are the final classes. O_j is the center of class C_j. n_j is the number of elements in C_j.

4. BUILDING CLASSIFIER BY ITERATIVELY APPLYING SVM

4.1 Feature Selection: TFIPNDF

Text classifier embeds the documents into some feature space, which may be extremely large, especially for very large vocabularies. And, the size of feature space affects the efficiency and effectiveness of text classifiers. Therefore, pruning the feature space is necessary and significant. In this paper, we prune the feature space according to the term frequency, which is based on the hypothesis that those very rare words are unimportant for classifying, and they will not affect the performance of the overall situation.

Thus, when representing the weight of term t_j in document d_i, x_{ij} is equal to 0 if term t_j does not occur in document d_i. So, the feature vector for document d_i is written as $d_i = (t_1 : x_{i1}, ..., t_j : x_{ij}, ..., t_{im} : x_{im})$. And an example is also illustrated in Fig.1 after feature weighting.

In this paper, we adopt an improved TFIDF term weighting method, TFIPNDF (Term Frequency Inverse Positive-Negative Document Frequency), which is based on statistical term frequency and inverse document frequency components from positive and negative training examples, respectively. For example, given a training set for text classifier, it contains 20 training documents (10 positive and 10 negative training documents), in which term t occurs in ten documents. According to TFIDF, that is, all

the inverse document frequency (IDF) weights of term *t* are $\log\left(20/10\right)$ in the collection. But, consider two cases, term *t* occurs in positive example set 5 times and in negative example set 5 times, and term *t* occurs in positive example set 9 times and in negative example set once. Obviously, term *t* reflects different importance in positive and negative examples in the two cases. Consequently, TFIDF does not take into account the difference of term IDF weighting in the positive and negative example sets separately. Compared with TFIDF, term frequency component of TFIPNDF is the same with TFIDF. However, when weighting inverse document frequency component, TFIPNDF method calculates the IPDF (Inverse Positive Document Frequency) and INDF (Inverse Negative Document Frequency) weight values in the positive and negative training examples, according to the distribution of the terms, respectively. In other words, IPNDF (or, IPDF and INDF) reflects the importance of the term in the positive and negative training examples, respectively. Therefore, TFIPNDF is composed of two parts:

$$TFIPNDF = \begin{cases} f_{ki} \times \dfrac{P_i}{S_P} \times \log\left(\dfrac{N}{n_i}\right), & (document_k \in P) \\ f_{ki} \times \dfrac{N_i}{S_N} \times \log\left(\dfrac{N}{n_i}\right), & (document_k \in N) \end{cases} \tag{4}$$

where f_{ki} is the frequency of word *i* in document *k*, *N* is the number of documents in the collection, n_i is the number of documents where word *i* occurs in the collection, P_i is the number of positive documents where word *i* occurs, N_i is the number of negative documents where word *i* occurs, S_P and S_N are the numbers of positive and negative documents in the collection, respectively.

```
1 1:0.052522168 2:0.040241696 3:0.0730656 4:0.030477624
1 1:0.046460036 25:0.042615794 48:0.049508102 50:0.0501
1 1:0.024018338 2:0.018402489 4:0.055749554 41:0.062027
1 2:0.10570139 3:0.063972905 16:0.046730578 37:0.058411
1 2:0.16337353 4:0.061866637 33:0.04347121 42:0.1377343
1 2:0.22854537 6:0.18543679 55:0.61050755 57:0.13617758
1 1:0.03545934 2:0.02173473 4:0.016461108 7:0.018080348
1 55:0.25021157 57:0.25115076 206:0.31192142 520:0.3825
1 1:0.010848313 3:0.020122005 8:0.06236598 18:0.0448506
1 1:0.03696323 2:0.07282459 4:0.024513219 7:0.013462263
```

Fig. 1. Illustration of a snippet of data matrix that stores the feature vectors (each row represents a feature vector).

A document collection may contain documents of many different lengths. It is useful to use normalized weight assignments. A vector length normalization of TFIPNDF is defined as:

$$x_{ki} = \begin{cases} \dfrac{f_{ki} \times \dfrac{P_i}{S_P} \times \log\left(\dfrac{N}{n_i}\right)}{\sqrt{\sum_{r=1}^{M}\left[f_{kr} \times \dfrac{P_r}{S_P} \times \log\left(\dfrac{N}{n_r}\right)\right]^2}}, & (document_k \in P) \\ \dfrac{f_{ki} \times \dfrac{N_i}{S_N} \times \log\left(\dfrac{N}{n_i}\right)}{\sqrt{\sum_{r=1}^{M}\left[f_{kr} \times \dfrac{N_r}{S_N} \times \log\left(\dfrac{N}{n_r}\right)\right]^2}}, & (document_k \in N) \end{cases} \quad (5)$$

where M is the number of all the features.

4.2 Iteratively building SVM classifier

In order to get more reliable negative examples, after document pre-processing, feature extracting and term weighting, we build a set of sub-classifiers by iteratively applying the SVM algorithm (http://www.csie.ntu.edu.tw/~cjlin/libsvm/) and construct the final text classifier. Also, the iterative process of building classifier can avoid missing reliable negative examples in U. We do not designate one specific classifier in the classifier set as the final one, instead, all of them are used to construct the final classifier based on voting method. Algorithm 2 is the implementation of constructing the text classifier (WVC: Weighted Voting Classifier) by using weighted voting method [14].

Algorithm of Constructing the Weighted Voting Classifier (WVC)
Input: positive example set P, unlabeled example set U, reliable negative example set RN_0.

1. **procedure Classifying**
2. Initialize
3. $PON \leftarrow P \cup RN_0$, $U \leftarrow U - RN_0$
4. $i \leftarrow 0$, $allP \leftarrow 0$
5. **repeat**
6. $SVM_i \leftarrow SVM\_training\ (PON)$
7. $p_i \leftarrow$ precision of SVM_i_predict (P)
8. $allP \leftarrow allP + p_i$
9. $RN_{i+1} \leftarrow$ the negative documents classified by SVM_i_predict (U)
10. $PON \leftarrow PON \cup RN_{i+1}$
11. $U \leftarrow U - RN_{i+1}$
12. $i \leftarrow i+1$
13. **until** RN_{i+1} is empty
14. $FinalClassifier = \sum_{k=0}^{i} \frac{p_k}{allP} SVM_k$
15. **end procedure**

Algorithm 2. Constructing the final classifier by using weighted voting method.

5. EXPERIMENTAL RESULTS

In this section, we built a PU classification system using techniques mentioned in this paper, and tested our method. In the experiment, we built our classifier using LIBSVM[1] (version 2.71) which can be downloaded at and evaluated our method C-CRNE by comparing with some baseline algorithms such as CPUE [28], 1-DNFII [14], 1-DNF [9]. The low frequency features (assigned a constant 5 in this paper) and stopwords in the dictionary are filtered out. The features in all documents are reordered and mapped using "ID" in accordance with the order of the features in dictionary. The performance metrics are proposed to evaluate the error rate of finding reliable negative examples and the accuracy of PU classification. Our experiments are performed over three datasets. Finally, our experimental results are also demonstrated.

5.1 Performance Metrics

To evaluate the performance of PU text classification, accuracy and AUC [36] are two typical evaluation metrics. Accuracy for text classification is the fraction of decisions that are correct, which measures how well it is doing at making the right decision. Suppose that *TP* (true positive) is the set of relevant documents that are retrieved by the classifier, and *TN* (true negative) is the set of non-relevant documents that are not retrieved by the classifier. *AllN* is the number of all the documents in the test dataset. Then the accuracy is calculated using Eq. 6 as follows.

$$accuracy = \frac{|TP| + |TN|}{AllN} \times 100\% \tag{6}$$

AUC is a better measure than accuracy and it considers the rank of positive examples in the classifier. AUC is calculated using Eq. 7 as follows:

$$AUC = \frac{S_0 - n_0(n_0 + 1)/2}{n_0 n_1} \tag{7}$$

where $S_0 = \sum r_i$, r_i is the rank of i_{th} positive example in the ranked list. n_0 and n_1 are the numbers of positive and negative examples respectively.

However, if the number of negative examples is much more than the number of positive examples, then the accuracy can be very high even the number of *TP* is low. So, two most common metrics (*precision* and *recall*) are also applied to reflect the availability of the system [21]. *precision* for text classification is the fraction of documents assigned that are relevant to the class, which measures how well it is doing at rejecting irrelevant documents. *recall* is the fraction of relevant documents assigned by classifier, which measures how well it is doing at finding all the relevant documents. We assume that *T* is the set of relevant documents in test dataset, and *C* is the set of relevant documents assigned by classifier. Therefore, we use *precision* and *recall* in Eqs.8 and 9 as follows:

$$precision = \frac{|C \cap T|}{|C|} \times 100\% \tag{8}$$

[1] (http://www.csie.ntu.edu.tw/~cjlin/libsvm/)

$$recall = \frac{|C \cap T|}{|T|} \times 100\% \tag{9}$$

F-Measure, which is defined as the harmonic mean of *precision* and *recall* value, is also used to measure the performance of our method. For different specific request, according to the importance of the *precision* and *recall*, we define *F-Measure* in Eq.10 as follows:

$$F\text{-}Measure = \frac{(\beta^2 + 1)\, precision \times recall}{\beta^2 \times precision + recall} \tag{10}$$

where β is a weight for reflecting the relative importance of *precision* and *recall* value. Obviously, if β >1, then the *recall* value is more important then *precision* value. In this paper, β is assigned a constant 1.

We also define *ERR* to compare our proposed algorithm C-CRNE with CPUE, 1-DNFII and 1-DNF algorithms in the number of identified reliable negative data and the error rate. Assume that $RN(P_t)$ is the set of the positive documents in the reliable negative data, and P_t is the set of positive documents in the unlabeled example set. The *ERR*(%) is calculated by Eq.11 as follows:

$$ERR = \frac{|RN(P_t)|}{|P_t|} \tag{11}$$

5.2 Datasets

In this paper, we perform several experiments on Reuters Corpus Volume 1[2] (RCV1), Reuters-21578[3] and 20 Newsgroups[4], respectively.

- Reuters Corpus Volume 1

Reuters dataset currently is the most widely used standard benchmarks of test collection for text classification. Reuters Corpus Volume 1, which has 804414 documents collected from the Reuters newswire, is used as our training and test dataset. We use "topic codes" set in category codes (topic, industry, and region). In the "topic" hierarchy, four hierarchical groups are included: CCAT (Corporate/Industrial), ECAT (Economics), GCAT (Government/Social), and MCAT (Markets), which contain 789,670 documents. Among 789,670 documents, only 3000 documents of each class are used. 70% of them are selected as training set, and the remaining 30% of them are used as test set.

- Reuters-21578

Reuters-21,578 dataset, which has 21578 documents collected from the Reuters newswire, is treated as another training and test data set. Of the 135 topics in Reuters 21578, 9980 documents from the most populous 10 topics (as shown in Table 1) are used in this paper.

---

[2] http://trec.nist.gov/data/reuters/reuters.html.

[3] http://www.daviddlewis.com/resources/testcollections/reuters21578

[4] http://qwone.com/~jason/20Newsgroups/.

Table 1. The number of documents in each topic from Reuters-21578.

Topics	Acq	Corn	Crude	Earn	Grain
Num	2369	237	578	3964	582
Topics	Interest	Money	Ship	Trade	Wheat
Num	478	717	286	486	283

- 20 Newsgroups

The 20 Newsgroups dataset, collected by Ken Lang, consists about 20,000 newsgroup documents, which partition evenly across 20 different newsgroups. In this paper, 10 classes are applied to be our dataset, 9840 documents in total. In addition, we use "20news-bydate.tar.gz", 20 Newsgroups sorted by date, as our dataset which training set and test set have been spilt by the provider in advance. The number of documents in each topic is shown in Table 2 as follows.

Table 2. The number of documents in each topic from 20 Newgroups.

Topics	comp.graphics	comp.os.ms-windows.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	comp.windows.x
Num	973	985	982	963	988
Topics	sci.crypt	sci.electronics	sci.med	sci.space	soc.religion.christian
Num	991	984	990	987	997

5.3 Results

To evaluate the effectiveness and efficiency of our proposed C-CRNE method, we run our algorithm and three baseline algorithms over three datasets. In the experiment, 15% positive samples S are selected randomly from P and added into U. Initial classifier is trained using P-S and RN. Tables 3, 4, and 5 show the number of reliable negative examples extracted by C-CRNE, CPUE, 1-DNFII and 1-DNF, and the corresponding error rates over three datasets. The *F-Measures* achieved by each method are also shown in the tables. And, we run 1-DNFII on $\lambda = 0.2$ setting [14]. Results show that the number of the reliable negative documents identified by C-CRNE is significantly more than that identified by CPUE, 1-DNFII, and 1-DNF. Although 1-DNF achieves the lowest error rates, it obtains the least reliable negative examples. C-CRNE overcomes CPUE and 1-DNFII in error rates. So, the comparisons indicate that C-CRNE can identify more reliable negative documents with very low error rates. Although there is no necessary correlation between the number of reliable negative examples and ERR. They are two different evaluation metrics that could evaluate the performance of a classifier. If the number of final reliable negative examples is small, then the accuracy and precision of the final classifier may decrease a lot. ERR is defined to calculate the ratio of the number of the positive documents in reliable negative data and the number of positive documents in the unlabeled example set. The goal of our classifier is to make a tradeoff between the two values, that is, identifying more reliable negative examples and making the ERR as low as possible.

Table 3. The number of reliable negative examples and the error rates in Reuters Corpus Volume 1.

		CCAT	ECAT	GCAT	MCAT
1-DNF	RN	346	297	369	304
	ERR(%)	0.28	0.00	0.46	0.00
	F-Measure	0.8961	0.8797	0.9065.	0.8829
1-DNFII	RN	1564	1917	2488	2368
	ERR(%)	1.16	0.87	1.45	0.98
	F-Measure	0.9185	0.896	0.9306	0.9141
CPUE	RN	1862	2074	2763	2613
	ERR(%)	1.04	0.75	1.29	0.86
	F-Measure	0.9285	0.9013	0.938	0.923
C-CRNE	RN	2200	2397	3244	3128
	ERR(%)	0.70	0.43	0.88	0.59
	F-Measure	0.951	0.9272	0.9615	0.9535

Table 4. The number of reliable negative examples and the error rates in Reuters-21578 dataset.

		Acq	Corn	Crude	Earn	Grain	Interest	Money	Ship	Trade	Wheat
1-DNF	RN	172	213	118	209	140	270	252	368	110	217
	ERR(%)	0.00	0.00	0.00	0.14	0.00	0.00	0.00	1.53	0.00	0.00
	F-Measure	0.93	0.882	0.9123	0.934	0.9058	0.8969	0.8935	0.891	0.9052	0.899
1-DNF II	RN	757	769	446	818	789	1120	1195	1401	371	898
	ERR(%)	1.06	1.75	0.00	0.28	0.00	0.00	0.00	1.73	0.00	0.00
	F-Measure	0.948	0.895	0.9325	0.962	0.936	0.9188	0.9128	0.908	0.9285	0.916
CPUE	RN	863	878	524	907	951	1270	1305	1546	498	1034
	ERR(%)	1.01	0.88	0.83	0.25	0.81	1.00	2.00	1.67	0.00	0.00
	F-Measure	0.955	0.911	0.9395	0.968	0.9476	0.9257	0.9208	0.918	0.944	0.9244
C-CR NE	RN	1231	1348	862	1255	1332	1673	1552	1862	814	1368
	ERR(%)	0.66	0.60	0.57	0.17	0.42	0.7	1.12	1.58	0.00	0.00
	F-Measure	0.985	0.956	0.9619	0.986	0.97	0.9474	0.9511	0.947	0.98	0.952

Table 5. The number of reliable negative examples and the error rates in 20 Newgroups dataset.

		comp. graphics	comp. os.ms-windows. misc	comp. sys.ibm.pc. hardware	comp. sys.mac.hardware	comp. windows.x	sci.crypt	sci.electronics	sci.med	sci.space	soc.religion. christian
1-DNF	RN	114	108	117	122	112	98	123	114	115	118
	ERR(%)	0.00	0.00	0.00	0.60	0.00	0.00	0.54	0.00	0.00	0.00
	F-Measure	0.848	0.843	0.841	0.832	0.821	0.858	0.863	0.855	0.866	0.863
1-DNFII	RN	429	403	579	577	512	403	549	381	410	380
	ERR(%)	1.76	1.19	1.74	1.77	1.19	1.21	1.80	1.32	1.71	1.75
	F-Measure	0.867	0.855	0.849	0.845	0.854	0.874	0.88	0.887	0.898	0.898
CPUE	RN	664	688	766	683	734	624	701	531	589	538
	ERR(%)	1.67	1.06	1.55	1.31	0.96	1.09	1.73	1.15	1.56	1.60
	F-Measure	0.88	0.877	0.868	0.861	0.863	0.878	0.888	0.89	0.903	0.904
C-CRNE	RN	887	1077	1020	1326	928	784	1055	1147	1145	940
	ERR(%)	1.07	0.78	0.88	0.64	0.69	0.78	1.53	0.73	0.84	1.13

F-Measure	0.895	0.89	0.883	0.88	0.879	0.893	0.897	0.906	0.915	0.923

To We also test the influence brought by the first step, that is, the performance of classification. In this paper, three baseline algorithms PEBL [9] and WVC [14] (C-CRNE, CPUE, and 1-DNFII) are selected to compare with our method.

We observe that the performance of text classification using our method outperforms them on three datasets over thousands of documents. To compare the performance of different techniques clearly, Fig.2 plots *F-Measure* performance of PEBL and WVC (C-CRNE, CPUE, and 1-DNFII) using three datasets. The result indicates that average *F-Measures* of our proposed method WVC (C-CRNE) are higher than other three methods for each dataset.

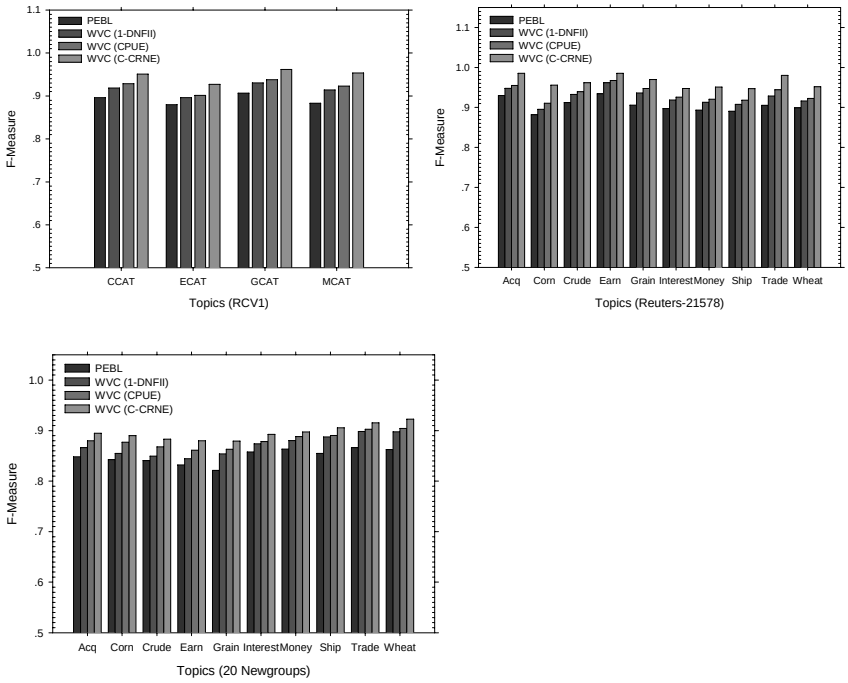

Fig. 2. The performance of the four text classifying methods for each topic on three datasets.

There are two steps in the process of PU text classification. The first step is collecting reliable negative examples from unlabeled dataset. The second step is iteratively training classifier using positive examples, reliable negative examples and unlabeled examples. The performance of a PU text classifier largely depends on whether it can collect more reliable negative examples with low error rates in the first step. In 1-DNF algorithm, it only considers the frequency difference of feature that occurs in *P* and *U*, but does not take into account the frequency of feature itself that occurs in *P*. For example, suppose we want to search documents about "music". The frequency of a feature "sport" is 0.2% in *P* and 0.1% in *U*. Obviously, it is not a positive feature, which conflicts with the results in 1-DNF. In this case, the number of features in positive feature set will be much more, but the number of documents in reliable negative example

set is much less or even zero. The improved 1-DNF algorithm (called constrained 1-DNF or 1-DNFII) not only considers the frequency difference of feature that occurs in *P* and *U*, but also takes into account the frequency of feature itself that occurs in *P*. That is, a feature is regarded as a positive one only when meeting two conditions: first, the frequency of the feature occurred in *P* is greater than that in *U*, and the frequency of the feature in *P* is greater than a fixed threshold. CPUE (Clustering Positive and Unlabeled Examples) identifies the reliable negative examples by clustering positive and unlabeled examples at the same time, which is time-consuming and reduces the accuracy. The error rates increase because the positive examples are used to cluster unlabeled examples directly, which many unlabeled examples are mistaken for non-reliable-negative examples. It would make the number of reliable negative examples reduce a lot. C-CRNE does not extract *RN* directly. It removes as many probable positive examples as possible. It defines a radius between a document and a class center. After clustering, the rest documents in *U* are regarded as reliable negative examples. From the perspective of the number of reliable negative examples and the error rates, C-CRNE achieves better performance.

The average accuracy and AUC graphs are plotted (as shown in Figs. 3 and 4) by using PEBL and WVC (C-CRNE, CPUE, and 1-DNFII) over three datasets. The result indicates that average *accuracy* and AUC of our proposed method WVC (C-CRNE) are higher than other three methods for each dataset.

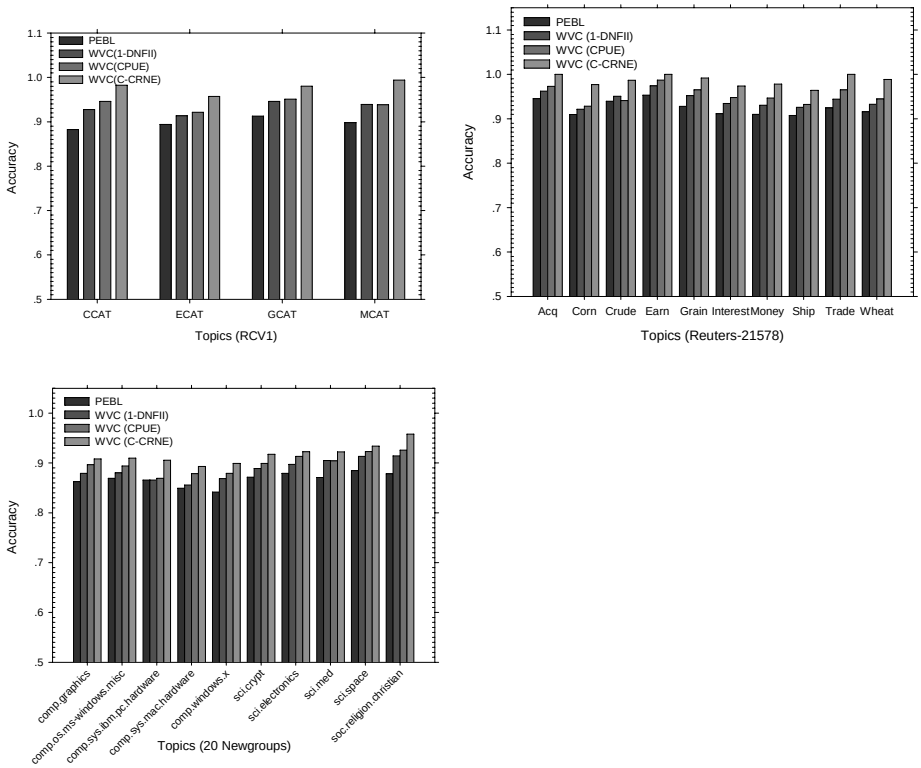

Fig. 3. The accuracy of the four text classifying methods for each topic on three datasets.

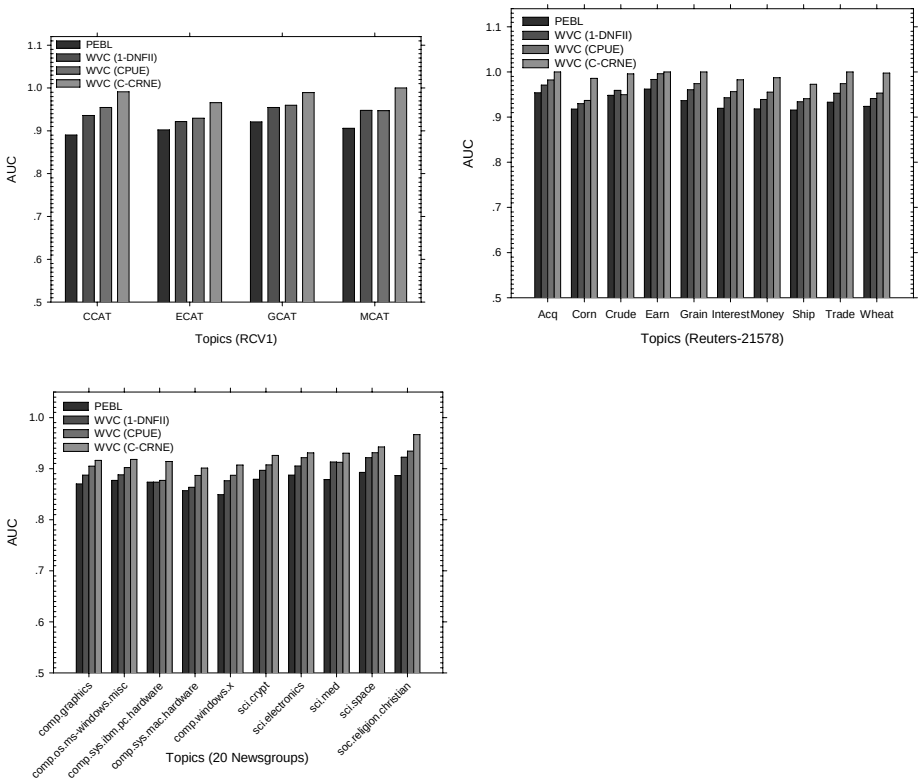

Fig. 4. The AUC of the four text classifying methods for each topic on three datasets.

In order to verify the effectiveness of TFIPNDF, we run the classifying system using different term weighting methods. Tables 6, 7 and 8 compare the performance of *F-Measure* achieved by our classifying method using TFIPNDF, TF-RF [37], TFIDF and Entropy Weighting [38] for each topic on the three datasets. We observe that the performance of classifying using TFIPNDF weighting outperforms TF-RF, TFIDF and Entropy Weighting on each dataset.

Table 6. The comparison of *F-Measures* achieved by our weighted voting classifying method using TFIPNDF, TF-RF, TFIDF and Entropy weighting for Reuters Corpus Volume 1.

	CCAT	ECAT	GCAT	MCAT
Entropy Weighting	0.918	0.9139	0.9408	0.9325
TFIDF	0.935	0.9173	0.9371	0.9378
TF-RF	0.94	0.9215	0.9497	0.9433
TFIPNDF	0.951	0.9272	0.9615	0.9535

Table 7. The comparison of *F-Measures* achieved by our weighted voting classifying method using TFIPNDF, TF-RF, TFIDF and Entropy weighting for Reuters-21578.

	Acq	Corn	Crude	Earn	Grain	Interest	Money	Ship	Trade	Wheat
Entropy Weighting	0.9586	0.9213	0.9227	0.9487	0.9412	0.9065	0.9177	0.9038	0.9424	0.9169
TFIDF	0.952	0.917	0.9306	0.9640	0.9372	0.9233	0.9058	0.9231	0.9462	0.9247
TF-RF	0.9671	0.9359	0.9462	0.9781	0.953	0.9396	0.9406	0.9345	0.9617	0.9482
TFIPNDF	0.9853	0.9553	0.9619	0.9855	0.97	0.9474	0.9511	0.9469	0.98	0.952

Table 8. The comparison of F-Measures achieved by our weighted voting classifying method using TFIPNDF, TF-RF, TFIDF and Entropy weighting for 20 Newsgroups.

	comp.graphics	comp.os.ms-window.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware	comp.windows.x	sci.crypt	sci.electronics	sci.med	sci.space	Soc.religion.christian
Entropy Weighting	0.8553	0.8462	0.8511	0.8346	0.8285	0.8613	0.8642	0.8722	0.8923	0.8838
TFIDF	0.863	0.8591	0.8358	0.8502	0.8349	0.8575	0.8748	0.8844	0.8811	0.8889
TF-RF	0.8755	0.8783	0.8652	0.8739	0.8511	0.8707	0.885	0.8913	0.8996	0.9036
TFIPNDF	0.8947	0.8898	0.883	0.88	0.8791	0.8927	0.8971	0.9056	0.9153	0.9226

6. CONCLUSION

In this paper, we present a novel clustering-based method for collecting reliable negative examples. Compared with traditional methods, our proposed C-CRNE collects more reliable negative examples from *U*. We conduct a series quantitative analysis for the effectiveness and efficiency of our method and several baseline algorithms. Experimental results draw important conclusions as follow. Traditional methods collect reliable negative examples directly such as 1-DNF and 1-DNFII. CPUE also uses clustering-based method to collect reliable negative examples but it does not reach high efficiency and accuracy. Our method overcomes their weakness. It removes as many probable positive examples from *U* as possible. The rest documents in *U* are regarded as reliable negative examples. This method not only obtains more reliable negative examples but also has low error rates and high efficiency. When building text classifier, TFIPNDF can be considered to make up for a defect of TFIDF in text classification. And the experimental result proved that the performance of classifying using TFIPNDF weighting outperforms TFIDF for each dataset. And, the first step in PU classification is a key step in building classifier. We also built a weighted voting classifier by iteratively applying the SVM algorithm using TFIPNDF. Due to the good performance generated by the first step, our classifier outperforms the other three state-of-art classifiers.

ACKNOWLEDGEMENT

We are grateful to NIST for permitting and mailing RCV 1 dataset, which is distributed on two CDs, to us.

REFERENCES

1. W.B. Croft, D. Metzler, and T. Strohman, *Search Engines: Informaton Retrieval in Practice*, Addison Wesley, 2009.
2. W. Liu, and T. Wang, "Online active multi-field learning for efficient email spam filtering," *Knowledge and Information Systems*, Vol. 33, no. 1, 2012, pp. 117-136.
3. G. Fumera, I. Pillai, and F. Roli, "Spam filtering based on the analysis of text information embedded into images," *Journal of Machine Learning Research*, Vol. 7, 2006, pp. 2699-2720.
4. X.G. Qi, and B.D.Davison, "Web page classification: feature and algorithms," *ACM Computing Surveys*, Vol. 41, no. 2, 2009.
5. I. Anotonellis, C. Bouras, and V. Poulopoulos, "Personalized news categorization through scalable text classification," *Frontiers of WWW Research and Development-APWEB*, *Lecture Notes in Computer Science*, Vol. 3841, 2006, pp. 391-401.
6. M. Hu, and B. Liu, "Mining and summarizing customer review," *Proceedings of ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 2004, pp. 168-177.
7. S. Kim, and E. Hovy, "Determining the sentiment of opinions," *Proceedings of the Intl. Conf. on Computational Linguistics*, 2004.
8. B. Liu, W. S. Lee, P. S. Yu, and X. L. Li, "Partially supervised classification of text documents," *The Nineteenth International Conference on Machine Learning*, 2002, pp. 384-397.
9. H. Yu, J. W. Han, and K. C. C. Chang, PEBL: positive example based learning for web page classification using SVM. *Proceedings of the Knowledge Discovery and Data Mining*, 2002, pp. 239-248.
10. X. L. Li, and B. Liu, "Learning to classify texts using positive and unlabeled data," *The International Joint Conference on Artificial Intelligence*, 2003, pp. 587-594.
11. B. Liu, Y. Dai, X. L. Li, W. S. Lee, and P. S. Yu, "Building text classifiers using positive and unlabeled examples," *Proceedings of the Third IEEE International Conference on Data Mining*, 2003, pp.179-186.
12. W. S. Lee, and B. Liu, "Learning with positive and unlabeled examples using weighted logistic regression," *Proceedings of the Twentieth Intl. Conf. on Machine Learning*, 2003, pp. 448-455.
13. L. M. Manevitz, and M. Yousef, "One-class SVMs for document classification," T*he Journal of Machine Learning Research*, Vol. 2, 2002, pp. 139-154.
14. T. Peng, W. L. Zuo, and F. L. He, SVM based adaptive learning method for text classification from positive and unlabeled documents, *Knowledge and Information Systems*, Vol. 16, no. 3, 2008, pp. 281-301.
15. S. Yu, and C. P. Li, "A graph based PU-learning approach for text classification. machine learning and data mining in pattern recognition, *Lecture Notes in Computer Science*, Vol. 4571, 2007, pp. 574-584.
16. Y. S. Xiao, B. Liu, J. Yin, L. B. Cao, C. Q. Zhang, and Z. F. Hao, "Similarity-based approach for positive and unlabeled learning," *Proceeding of the Twenty-Second International Joint Conference on Artificial Intelligence*, 2011, pp. 1577-1582.
17. J. Wu, and M. Y. Lu, "Asymmetric semi-supervised boosting scheme for interactive

image retrieval," *ETRI Journal*, Vol. 32, no. 5, 2010, pp. 766-776.
18. Z. M. Li, L. Li, Y. J. Liu, and J. W.Bao, "An improved method for support vector machine-based active feedback," *2008 3rd International Conference on Pervasive Computing and Applications*, Vols. 1 and 2, 2008, pp. 389-393.
19. Z. H. Zhou, K. J. Chen, and H. B. Dai, "Enhancing relevance feedback in image retrieval using unlabeled data," *ACM Transactions on Information Systems*, Vol. 24, no. 2, 2006, pp. 219-244.
20. L. Y. Sheng, and A. Ortega, "Graph based partially supervised learning of documents," *2011 IEEE International Workshop on Machine Learning for Signal Processing*, 2011, pp. 1-6.
21. B. Liu, *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*, 2nd Edition, Springer, 2011.
22. R. Cooley, B. Mobasher, and J. Srivastava, "Data preparation for mining world wide web browsing patterns," *Knowledge and Information Systems*, Vol. 1, no. 1, 1999, pp. 5-32.
23. S. L. Chuang, and L. F. Chien, "Enriching web taxonomies through subject categorization of query terms from search engine logs," *Decision Support Systems*, Vol. 35, no. 1, 2003, pp. 113-127.
24. F. Denis, "PAC Learning from positive statistical queries," *Proceedings of Intl. Conf. on Algorithmic Learning Theory (ALT'98)*, 1998, pp. 112-126.
25. S. Muggleton, "Learning from the positive data," *Inductive Logic Programming Workshop*, 1996, pp. 358-376.
26. S. Schölkopf, J. Platt, J. Shawe, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," T*echnical Report MSR-TR-99-87*, *Microsoft Research*, 1999, pp. 1443-1471.
27. K. Aas, and L. Eikvil, "Text categorization: a survey," *Norwegian Computing Center*, 1999.
28. B. Z. Zhang, and W. L. Zuo, "A novel reliable negative method based on clustering for learning from positive and unlabeled examples," *AIRS 2008*, *Lecture Notes in Computer Science*, Vol. 4993, 2008, pp. 385-392.
29. B. Z. Zhang, and W. L. Zuo, "Reliable negative extracting based on KNN for learning from positive and unlabeled examples," *Journal of Computers*, Vol. 4, no. 1, 2009, pp. 94-101.
30. S. R. Pan, Y. Zhang, and X. Li, "Dynamic classifier ensemble for positive unlabeled text stream classification," *Knowledge and Information Systems*, Vol. 33, no. 2, 2012, pp. 267-287.
31. S. Yu, X. Y. Zhou, and C. P. Li, "Semi-supervised text classification using positive and unlabeled data," *Advances in IntelligentIT: Active Media Technology, Frontiers in Artificial Intelligence and Applications*, Vol. 138, 2006, pp. 249-254.
32. I. T. Nagy, R. Farkas, and J. Csirik, "On positive and unlabeled learning for text classification," *Proceedings of 14th International Conferences on Text, Speech and Dialogue*, *Lecture Notes in Artificial Intelligence*, Vol. 6836, 2011, pp. 219-226.
33. F. Lu, and Q. Y. Bai, "Semi-supervised text categorization with only a few positive and unlabeled documents," *2010 3rd International Conference on Biomedical Engineering and Informatics*, Vols. 1-7, 2010, pp. 3075-3079.
34. X. L. Li, B. Liu, and S. K. Ng, "Learning to classify documents with only a small positive training set," *Machine Learning ECML*, *Lecture Notes in Computer Science*, Vol. 4701, 2007,

pp. 201-213.
35. Q. Qiu, Y. Zhang, and J. P. Zhu, "Building a text classifier by a keyword and wikipedia knowledge," *Proceedings of 5th International Conference on Advanced Data Mining and Applications*, *Lecture Notes in Computer Science*, Vol. 5678, 2009, pp. 277-287.
36. J. Huang, and C.X. Ling, "Using AUC and accuracy in evaluating learning algorithms", *IEEE Transaction on Knowledge and Data Engineering*, Vol. 17, no. 3, 2005, pp. 299-310.
37. M. Lan, G-L. Tan, and H-B. Low, "Proposing a new term weighting scheme for text categorization", *American Association for Artificial Intelligence*, Vol. 6, 2006, pp. 763-768.
38. S. T. Dumais, "Improving the retrieval information from external sources," *Behaviour Research Methods, Instruments and Computers*, Vol. 23, no. 2, 1991, pp. 229-236.