Computational Linguistics and Chinese Language Processing
Vol. 5, No. 2, August 2000, pp. 43-58
© Computational Linguistics Society of R.O.C.

Design and Evaluation of Approaches to Automatic Chinese Text Categorization

Jyh-Jong Tsay and Jing-Doo Wang[*]

Abstract

In this paper, we propose and evaluate approaches to categorizing Chinese texts, which consist of term extraction, term selection, term clustering and text classification. We propose a scalable approach which uses frequency counts to identify left and right boundaries of possibly significant terms. We used the combination of term selection and term clustering to reduce the dimension of the vector space to a practical level. While the huge number of possible Chinese terms makes most of the machine learning algorithms impractical, results obtained in an experiment on a CAN news collection show that the dimension could be dramatically reduced to 1200 while approximately the same level of classification accuracy was maintained using our approach. We also studied and compared the performance of three well known classifiers, the Rocchio linear classifier, naive Bayes probabilistic classifier and k-nearest neighbors(kNN) classifier, when they were applied to categorize Chinese texts. Overall, kNN achieved the best accuracy, about 78.3%, but required large amounts of computation time and memory when used to classify new texts. Rocchio was very time and memory efficient, and achieved a high level of accuracy, about 75.4%. In practical implementation, Rocchio may be a good choice.

Keywords: Term Clustering, Term Selection, Text Categorization.

1. Introduction

In recent years, we have seen a tremendous growth in the number of online text documents available on the Internet, in digital libraries and news sources. Effective location of information in these huge resources is difficult without good indexing as well as organization of text collections. Automatic text categorization, which is defined as the task of assigning predefined class (category) labels to free text documents, is one of

[*] Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan 62107, ROC. E-mail: {tsay,jdwang}@cs.ccu.edu.tw

the main techniques that are useful both in organizing and in locating information in these huge collections.

Many approaches to text categorization and web page classification have been proposed [2,9,12,20]. Most of them have been evaluated using English texts. Evaluation of these approaches using texts in Chinese and other oriental languages has been limited. In [22], Yang *et al.* proposed and evaluated several approaches to Chinese text categorization. The number of training and testing documents used was 2306, and the number of extracted terms used was 4711. Yang's work was quite preliminary and achieved classification accuracy of only about 67%. Since then, tremendous advances have been made in categorization techniques [20]. Most of the recently proposed techniques have not been evaluated using Chinese texts.

The objective of this study was to design and evaluate approaches to categorizing Chinese texts. In particular, we implemented and evaluated approaches which consist of the following processes: term extraction, term selection, term clustering and text classification. Note that in Chinese texts, although a sentence is composed of a sequence of terms, no white spaces are inserted to separate terms from each other. Term extraction which segments sentences into term sequences is a difficult task [5]. Several approaches have been proposed to extract terms from Chinese texts [4,13]. In this paper, we propose a scalable approach [17] which is based on String B-trees proposed in [7] and is capable of handling huge numbers of text documents. Our approach uses frequency counts to identify possible term boundaries as proposed in [13] and is able to identify new terms which occur very often in Chinese texts.

However, the number of terms in Chinese can be very large. It is very easy to encounter 10^6 or even more terms in moderately-sized collections. The huge number of possible terms results in very high dimensionality when documents was presented in a vector space model and makes many machine learning algorithms impractical. To reduce the dimension to a practical level, we propose to perform term selection and term clustering on extracted terms. In particular, we use the χ^2 statistic[16] to select terms that are highly correlated to class categories. In [16], we presented an extensive comparison of several measures for term selection in Chinese text categorization, such as the odds ratio, information gain, mutual information, and χ^2 statistic. Experimental results shows that the χ^2 statistic approach achieves the best performance. Notice that in term selection, if only a small number of terms is selected, a document may contain very few or even none of the selected terms, and thus will be classified into the default class. On the other hand, a large number of selected terms make automatic categorization computationally impractical. We thus allow a large number of terms to be selected and then perform term clustering to group similar terms into clusters.

A large number of algorithms for clustering are Available [11]. Most of them are unsupervised and ignore any class labels that are given. In this study, we used distributional clustering [2], which explicitly takes advantage of the class labels to group terms with similar class distributions into the same cluster. In an experiment on a collection of CNA news [1] articles, the number of terms extracted was 548363.

Experimental results show that the level of classification accuracy could be maintained while the dimension was reduced to 1200 by selecting 90000 terms first and then clustering them into 1200 clusters. Notice that term selection and term clustering also can compeensate for imprecision in term extraction as erroneous terms can be dropped out during term selection or grouped with more significant terms through term clustering. In addition to term selection and term clustering algorithms, there are others which can be applied to reduce the level of dimensionality, such as Principle Component Analysis (PCA) [6]. PCA is an unsupervised dimensional reduction technique, whereas distributional clustering is supervised and can take advantage of class labels to concentrate effort on the specific task of categorization. We expect distributional clustering to perform well in the context of text categorization.

In this paper, we also compare extensively three well-known classifiers, including the Rocchio linear classifier [12], naive Bayes probabilistic classifier, and k-nearest neighbor (kNN) classifier [20]. We observed in an experiment that the classification accuracy of Rocchio and kNN improved slightly as the dimension was reduced to 1200 by means of term selection and term clustering but that the accuracy of the naive Bayes classifier dropped slightly. This might have been due to the fact that term clustering refines the shapes of each cluster but distorts the distribution of each term. Overall, kNN achieved the best accuracy, about 78.3%, but required large amounts of computation time and memory when used to classify new texts. Rocchio is very time and memory efficient, and achieves accuracy of about 75.4%, which is slightly worse than kNN.

Recently, Huang *et al.* [10] evaluated the weight matrix approach, which estimates the relative importance of the keywords in each class and classifies a test news to the class that maximizes the sum of the weights of keywords appearing in that news. Although they achieved about 88% classification accuracy, their experiment was different from ours as well as those used in much related research [3,22]. First, the training news did not come from the same news source as the test news, but come from a thesaurus [19] that was carefully built by linguistic specialists. Second, the test news was classified by readers who could employ logic that was close to that assumed by the classification algorithms but different from that employed by the editors. Third, a piece of test news could be assigned to multiple classes when it covered topics from different classes. In fact, for a collection of 1136 news items, 1380 class labels were assigned, which indicates that about 20% of the test news iteems had multiple class labels. However, in the CNA news collection used in this study, each news items had exactly one predefined class no matter how many topics it covered. It is not clear whether or not the weight matrix approach can achieve the same performance when all the differences are removed.

The remainder of this paper is organized as follows. Section 2 sketches the String B-tree approach to term extraction. Section 3 describes the χ^2 statistic approach to term selection. Section 4 describes distributional clustering. Section 5 reviews the classifiers compared in this paper. Section 6 gives experimental results. Section 7 gives conclusions.

2. Term Extraction

In this paper, we propose a scalable approach [18] to term extraction, which is based on String B-trees (SB-trees) [7]. This approach can handle large text collections and can identify newly created terms frequently found in Chinese. It does not use a dictionary but rather uses frequency counts to identify the boundaries of possible terms as in [13]. We will describe the term extraction method in the following.

Let *w* be a string. For any character *x*, let *P (wx|w)* be the probability that w is followed by *x*, and let *P(xw|w)* be the probability that w is preceded by *x*. We say that w passes right boundary verification if *P(wx|w)* < θ_1 for all x and passes left boundary verification if *P(xw|w)* < θ_2 for all *x*. The probability *P(wx|w),* resp. *P(xw|w)*, is estimated by $\frac{TF(wx)}{TF(w)}$, resp. $\frac{TF(xw)}{TF(w)}$, where *TF(y)* is the term frequency of string y. String *w* is identified as a significant term if it passes both right and left boundary verifications. In this paper, we simply set $\theta_1 = \theta_2 = 1$, which means that w will be identified as a significant term when it has at least two distinct successor and predecessor characters. For each class, we build two SB-trees, one for all the suffixes [8] of the original texts used for right boundary verification, and the other for the suffixes of the reversed texts which is used for left boundary verification. Notice that SB-trees are scalable; they can maintain dynamic collections and identify new terms as new articles are inserted.

3. Term Selection

Term selection is performed to choose representative terms for each class such that these terms can distinguish one class from the others. After the term extraction process is completed, there are many terms remain that are not informative for categorization. In [16], we extensively compared several measures used for term selection in Chinese text categorization, such as the odds ratio, information gain, mutual information, and χ^2 statistic. Experimental results show that the χ^2 statistic approach achieves the best performance when combined with the naive Bayes classifier. In this study, we used the χ^2 statistic [21] approach to perform term selection.

For a term *t* and a class *c*, the χ^2 statistic measures the correlation between *t* and *c*. Let *A* be the number of times *t* and *c* co-occur, let *B* be the number of times *t* occurs without *c*, let *P* be the number of times *c* occurs without *t*, let *Q* be the number of times neither *t* nor *c* occur, and let *N* be the total number of documents. The χ^2 statistic is defined as $\chi^2(t,c) = \frac{N \times (AQ - BP)^2}{(A+P)\times(B+Q)\times(A+B)\times(P+Q)}$.

Notice that the χ^2 statistic approach prefers terms that are highly correlated with a particular class. For each term, the χ^2 statistic scores with regard to different classes can be different. In [21], Yang used the average or the maximum of the scores to select representative terms, which may result in a biased distribution of selected terms between classes. To avoid this situation, we select from each class the same number of terms having the largest χ^2 statistic in that class.

4 Term Clustering

We perform term clustering to further reduce the dimension of the vector space after the term selection process. In order to avoid the situation in which a document contains none of the selected terms, in term selection, we select a suitable large set of terms which may require a large amount of computation time and memory for classification. Term clustering groups similar terms into one cluster that no longer distinguishes between constituent terms. In this study, we used distributional clustering [2], which groups terms with similar distributions over classes into the same cluster. Note that distributional clustering can compensate for the drawback of term extraction, where incomplete terms are clustered into the group containing their original terms. On the other hand, when training data is sparse, performance may be improved by averaging statistics of similar words together so that the resulting estimates are more robust. We describe distributional clustering [2] in more detail in the following.

Term clustering algorithms define a similarity measure between terms and group similar terms into term clusters. In distributional clustering, the difference between two term distributions is measured by Kullback-Leibler (KL) divergence. For term t_i and term t_j, the KL divergence, denoted as $D(P(C \mid t_i) \parallel P(C \mid t_j))$, is defined as $-\sum_{k=1}^{|C|} P(C_k \mid t_i) \log \frac{P(C_k \mid t_i)}{P(C_k \mid t_j)}$, where |*C*| is the number of classes and $P(C_k \mid t_i)$ is the probability of class C_k given term t_i. To avoid the odd properties of KL divergence, such as asymmetry, we use the average KL divergence defined as $\frac{P(t_i)}{P(t_i \vee t_j)} \cdot D(P(C \mid t_i) \parallel P(C \mid t_i \vee t_j)) + \frac{P(t_j)}{P(t_i \vee t_j)} \cdot D(P(C \mid t_j) \parallel P(C \mid t_i \vee t_j))$, where $t_i \vee t_j$ represents clustering of term t_i and term t_j into one group. Based on the average KL divergence, we apply a simple greedy agglomerative algorithm to cluster terms as follows. Let *M* be the number of final clusters. Initially, *M* terms are selected as seeds. Each term represents a singleton cluster. The following process is repeated until all the terms have been added: the two most similar clusters are merged into one cluster, and then the term that has the highest χ^2 statistic measure among the remaining terms is added as a singleton cluster. The initial *M* seeding terms are uniformly selected from all classes. That is, from each class, the $M/|C|$ terms that have the highest χ^2 statistic measure are selected as initial seeds. This avoids the problem of bias [2], where the *M* initial clusters may prefer some classes.

5. Classifiers

In this paper, we compare three wellknown classifiers, including the Rocchio linear classifier, naive Bayes (NB) probabilistic classifier and k-nearest neighbor (kNN) classifier, which are reviewed in the following sections.

5.1 Rocchio Linear Classifier

The Rocchio algorithm is a training algorithm [12] for linear classifiers and was initially developed for information retrieval in the vector space model. The basic idea is to construct one prototype vector per class, using a training set of documents. Given a class, the training document collection consists of positive and negative examples. Positive examples are those documents belonging to that class, while negative examples are those documents not belonging to that class. The prototype vector of a class is the centroid of positive examples, tuned using negative examples. Let D_i be a document in the training collection *D*, represented as a vector $(d_{i,1}, d_{i,2}, \cdots, d_{i,n})$, where $d_{i,j}$ is the weight assigned to the *j*th term and *n* is the dimension of the document space. To determine $d_{i,j}$, we use the TF-IDF weighting method [15], which has been shown to be effective when used in the vector space model. Let $tf_{i,j}$ be the term frequency of the *j*th term in document D_i, and let df_j be the document frequency of the *j*th term in training collection *D*. In this paper, the TF-IDF weight is defined as $d_{i,j} = \log_2(tf_{i,j} + 1) * \log_2(\frac{N}{df_j})$, where *N* is the total number of documents in the training collection.

The prototype vector $G_i = (g_{i,1}, g_{i,2}, \cdots, g_{i,n})$ of class C_i is defined as $G_i = \frac{\sum_{D_i \in C_k} D_i}{|C_k|} - \eta \frac{\sum_{D_i \in D - C_k} D_i}{|D - C_k|}$, where η is the parameter that adjusts the relative impact of positive and negative examples. We have experimented with different values for η, including 0.25, 0.5, 0.75 and 1. The best choice of η in our experiment was found to be 0.5 when $n = 90000$, and was 0.25 when $n = 1200$. To classify a request document *X*, we compute the cosine similarity between *X* and each prototype vector G_i, and assign to *X* the class whose prototype vector has the highest degree of cosine similarity with *X*. Cosine similarity is defined as $CosSim(X, G_i) = \frac{\sum_{j=1}^{n} x_j \cdot g_{i,j}}{\sqrt{\sum_{j=1}^{n} x_j^2} \sqrt{\sum_{j=1}^{n} g_{i,j}^2}}$.

5.2 Naive Bayes (NB) Classifier

The Naive Bayes (NB) probabilistic classifiers have been studied for application to machine learning [14]. The basic idea in NB is to use the joint probabilities of terms and classes to estimate the probabilities of classes given a document. The naive part is the assumption of term independence, i.e., the conditional probability of a term, given a class, is assumed to be independent from the conditional probabilities of other words given that class. This assumption makes computation for NB classifiers far more efficient than that for the non-naive Bayes approaches [20] whose time complexity are exponential.

Let *X* be a request document; NB assigns to *X* the most probable class C_{NB} defined as $C_{NB} = \arg\max_{C_k \in C} P(C_k \mid X)$. By Bayes' theorem, $P(C_k \mid X) = \frac{P(X \mid C_k)P(C_k)}{\sum_{C_i \in C} P(X \mid C_i)P(C_i)}$. Due to the assumption of term independence, $P(X \mid C_k) = \Pi_{j=1}^{|X|} P(t_j \mid C_k)$, where $P(t_j \mid C_k)$ is the conditional probability of term t_j given class C_k. Notice that the above equation works well when every term appears in every document. However, the product becomes 0 when some terms do not appear in the given document. We use $P(t_j \mid C_k) = \frac{1 + TF(t_j, C_k)}{|T| + \sum_j^{|T|} TF(t_j, C_k)}$ in order to approximate $P(t_j \mid C_k)$ to avoid the possibility that the product will become 0, where $TF(t_j, C_k)$ is the frequency of occurrence of term t_j in documents of class C_k and |*T*| is the total number of distinct terms used in the domain of document representation. The formula used to predict the probability of class value C_k for a given document *X* is $P(C_k \mid X) = \frac{P(C_k)\Pi_{t_j \in X} P(t_j \mid C_k)^{TF(t_j, X)}}{\sum_i P(C_i)\Pi_{t_j \in X} P(t_j \mid C_i)^{TF(t_j, X)}}$.

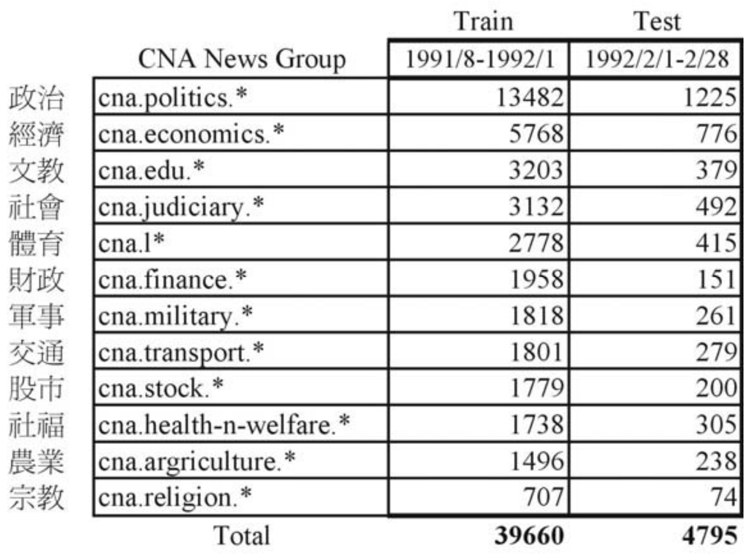
5.3 k-Nearest Neighbor (kNN) Classifier

Given an arbitrary request document *X*, kNN ranks its nearest neighbors among the training documents and uses the classes of the *k* top-ranking neighbors to predict the classes of *X*. The similarity score of each neighbor document when it is compared to *X* is used as the weight of the class of the neighboring document, and the sum of the class weights over the *k* nearest neighbors is used to perform class ranking[20].

In a kNN algorithm, each training document D_i as well as the request document *X* are represented by means of vectors as $(d_{i,1}, d_{i,2}, \cdots, d_{i,n})$ and $(x_1, x_2, \cdots, x_n)$, respectively. To conduct categorization, the cosine similarity between each D_i and *X* is calculated. The training documents are sorted using the cosine similarity metric in descending order. Then the *k* top-ranking documents are selected. The final score of the request document *X* when compared to each class is calculated by summing the cosine similarity metric of these *k* selected documents and their class association. The class with the highest score is assigned to *X*. We have performed an experiment using different values of *k*, including 5, 10, 15, 20, 30, 50, 100, 150, 200 and 300. The best choice of *k* in our experiment is 15 when n = 90000 and is 10 when n = 1200.

6. Experimental Results

In our experiment, we used Chinese news articles from the Central News Agency (CNA)[1]. We used news articles spanning a period of one year, from 1/1/1991 to 12/31/1991, to extract terms. News articles from the six-month period 8/1/1991 to 1/31/1992 were used as training data to train classifiers. The testing data consisted of news articles from the one-month period 2/1/1992 to 2/28/1992. All the news articles were preclassified into 12 classes, listted in Figure 1. Note that the number of texts used was far larger than that employed in previous related researches [10,22]. As a result, the conclusions drawn based on our experimental results are believed to be more reliable.



		Train	Test
	CNA News Group	1991/8-1992/1	1992/2/1-2/28
政治	cna.politics.*	13482	1225
經濟	cna.economics.*	5768	776
文教	cna.edu.*	3203	379
社會	cna.judiciary.*	3132	492
體育	cna.l*	2778	415
財政	cna.finance.*	1958	151
軍事	cna.military.*	1818	261
交通	cna.transport.*	1801	279
股市	cna.stock.*	1779	200
社福	cna.health-n-welfare.*	1738	305
農業	cna.argriculture.*	1496	238
宗教	cna.religion.*	707	74
	Total	**39660**	**4795**

Figure 1 *The distribution of CAN news articles.*

The news articles were not uniformly distributed over the classes, as shown in Figure 1. We, thus, measure the classification accuracy at both micro and macro levels. Three performance measures were used to evaluate the performance of each classifier: *MicroAccuracy*, *MacroAccuracy* and

AccuracyVariance. Let |*C*| be the number of predefined classes, and let $|C_i|$ be the number of testing news articles that are preclassified into the *i*th class, and let $N = \sum_{i=1}^{i=|C|} |C_i|$ be the total number of testing news articles. Let $|H_{i,j}|$ be the number of testing news articles in C_i that are classified into C_j. Let $Acc(i) = \frac{|H_{i,i}|}{|C_i|}$ be the classification accuracy within class C_i. MicroAccuracy is defined as $\frac{\sum_{i=1}^{i=|C|} |H_{i,i}|}{N}$, which represents the overall average of classification accuracy. MacroAccuracy is defined as $\frac{\sum_{i=1}^{i=|C|} Acc(i)}{|C|}$, which represents the average of the classification accuracy within classes. AccuracyVariance is defined as $\frac{\sum_{i=1}^{i=|C|} (Acc(i) - MacroAccuracy)^2}{|C|}$, which represents the variance of accuracy among classes.

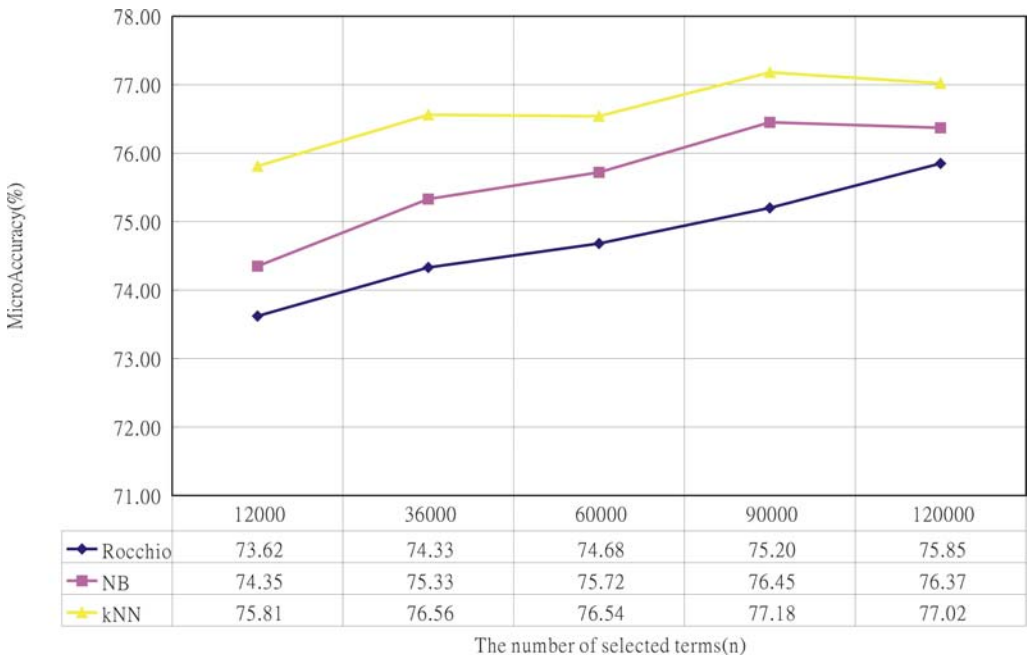
	n=90000	n=1200
Rocchio	00:21:13	00:04:26
naive Bayes	00:14:17	00:08:01
kNN	02:39:25	01:54:22

Figure 2 *Classification time.*

6.1 Dimension Reduction

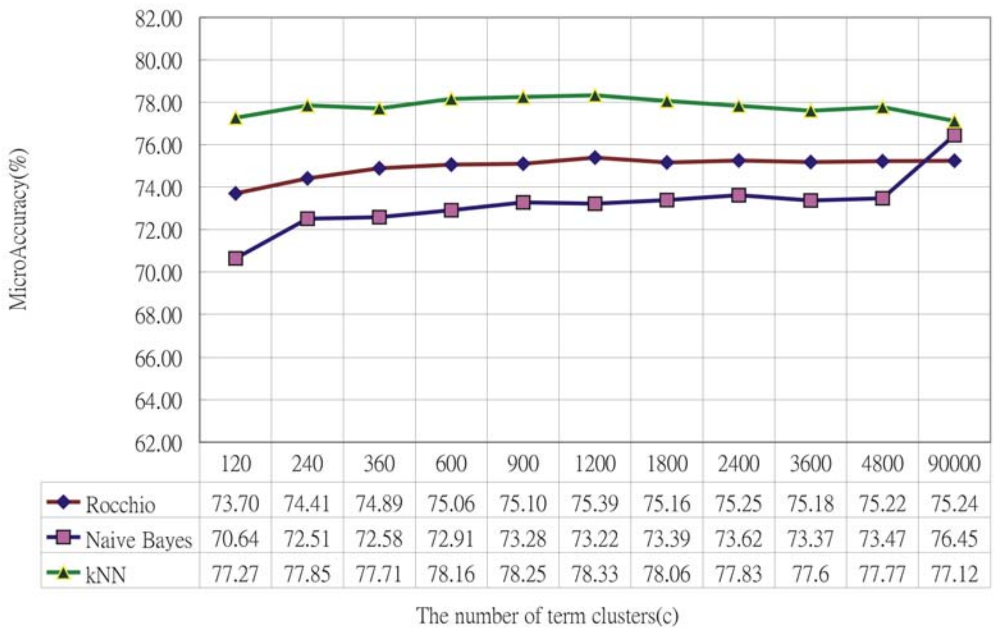
We performed term extraction, term selection and term clustering to reduce the dimension. Both the space and time required to classify new documents could be reduced as the dimension of the vector space was reduced. Figure 2 shows the time needed to classify new documents, measured on a PC with a Pentium II 233 CPU, 128MB RAM and an IDE HardDisk, for dimension n = 90000 and 1200, respectively.

In the term extraction process, terms that appeared fewer than 10 times or in only one document were dropped out. We then used frequency counts to identify significant terms. The number of significant terms extracted was 548363. Term selection was then performed to select a subset of most representative terms. In order to find an appropriate number *p* of selected terms, we experimented for different values of *p*, including 12000, 36000, 60000, 90000 and 120000. We choose a *p* value of 90000 because kNN and NB achieved the best MicroAccuracy results of 77.12% and 76.45%, respectively, when *p* was 90000, as indicated in Figure 3. The selected terms were clustered using distributional clustering into term clusters. To choose a suitable number *c* of term clusters, we experimented with different values of *c*, including 120, 240, 360, 600, 900, 1200, 1800, 2400, 3600 and 4800. We choose a *c* value of 1200 because kNN and Rocchio achieved the best performance when *c* was 1200, as shown in Figure 4.



	12000	36000	60000	90000	120000
Rocchio	73.62	74.33	74.68	75.20	75.85
NB	74.35	75.33	75.72	76.45	76.37
kNN	75.81	76.56	76.54	77.18	77.02

Figure 3 *MicroAccuacy comparison(term selection).*



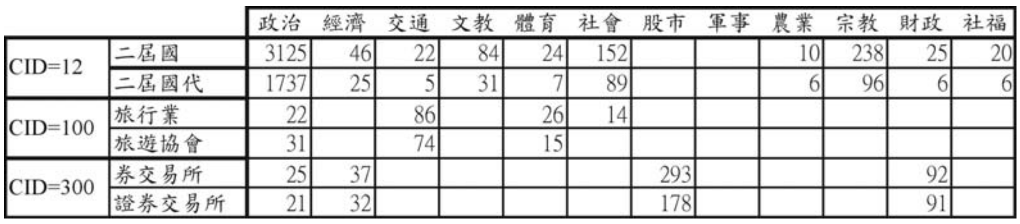
	120	240	360	600	900	1200	1800	2400	3600	4800	90000
Rocchio	73.70	74.41	74.89	75.06	75.10	75.39	75.16	75.25	75.18	75.22	75.24
Naive Bayes	70.64	72.51	72.58	72.91	73.28	73.22	73.39	73.62	73.37	73.47	76.45
kNN	77.27	77.85	77.71	78.16	78.25	78.33	78.06	77.83	77.6	77.77	77.12

Figure 4** MicroAccuacy comparison(term clustering)**.

Cluster ID(CID)

	12	100	207	225	300
1	二屆國	公路和	交響	犯案	今天在東京
2	二屆國代	在交通	交響樂團	刑事警察	交易所
3	二屆國代選舉	的快樂	巡迴演出	在逃	券交易所
4	的候選人	的班	的音樂	收押	證券交易所
5	候選人	旅行業	的舞	判處死刑	
6	候選人的	旅客的	奏會	官認為	
7	國大代表	旅遊協會	國立藝	押回	
8	國代候選人	泰航	國樂	前科	
9	國代選舉	機票	演奏	看守所	
10			演奏會	書指出	
11			舞蹈	處死刑	
12			樂家	被告	
13			樂團	槍枝	
14			鋼琴	辦案	
15			藝術學	警方在	

***Figure 5** Term clustering examples.*



		政治	經濟	交通	文教	體育	社會	股市	軍事	農業	宗教	財政	社福
CID=12	二屆國	3125	46	22	84	24	152			10	238	25	20
	二屆國代	1737	25	5	31	7	89			6	96	6	6
CID=100	旅行業	22		86		26	14						
	旅遊協會	31		74		15							
CID=300	券交易所	25	37					293				92	
	證券交易所	21	32					178				91	

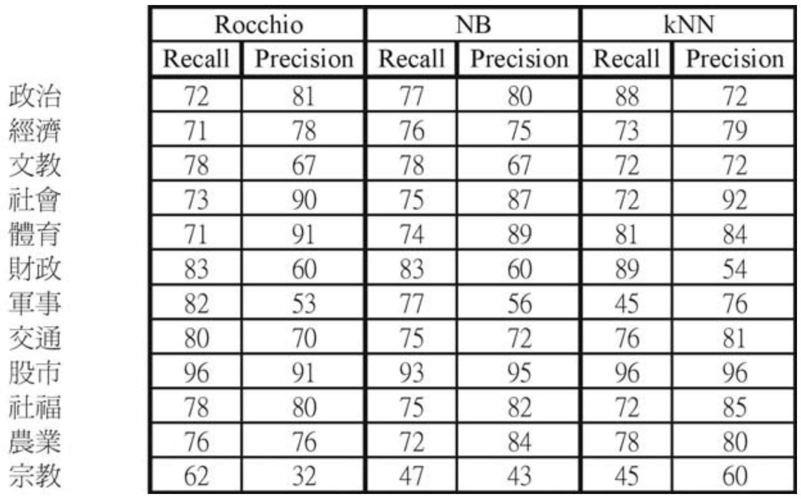
Figure 6 *Term frequencies in each class.*

Figure 5 shows some examples of term groups. In addition to clustering similar terms to reduce the dimension, term clustering can also cluster redundant substrings that are erroneously identified during term extraction into the group that contains their original terms. For example, as shown in Figure6,"二屆國" and "二屆國代" are clustered into group 12; "證券交易所" and "券交易所" are clustered into group 300. On the other hand, the averaging statistics of similar words may result in more robust estimates. For example, "旅行業"(a travel agent) and "旅遊協會"(a travel agency association) are similar words and are clustered into group 100.

In [2], Baker claimed that performance can be improved by means of term clustering when training data is sparse because by averaging statistics of similar words, more robust estimates can be obtained. This was confirmed by our experiment. Note that our training data was quite sparse as the average number of none-zero items in training vectors was 106 when *n* is 90000, and was 79 when *c* is 1200. The memory space could be reduced by 25% $(\frac{106-79}{106}=0.25)$, and the averaged statistics of terms were more robust estimates when the percentage of none-zero items increased from 0.12%(=106/90000) to 6.58%(=79/1200) due to term clustering.

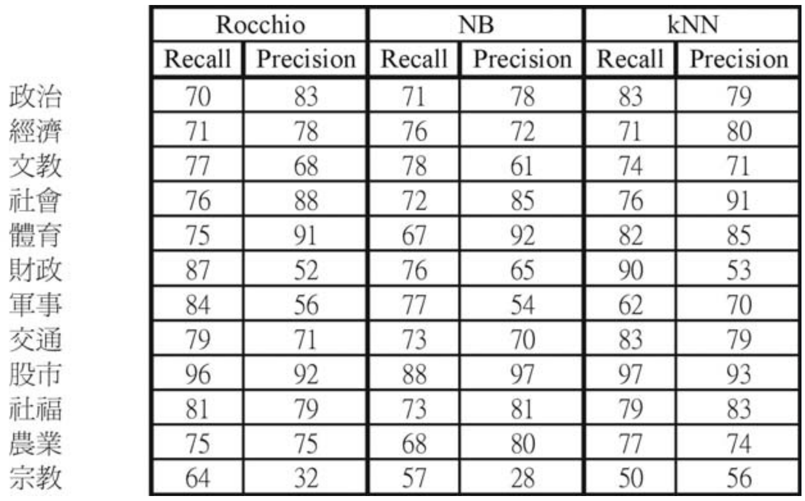
6.2 Classifiers Comparison

Overall, kNN achieved the best MicroAccuracy results, and Rocchio achieved slightly worse results, as shown in Figure 7 and Figure 8. Note that the MicroAccuracy results for Rocchio and kNN improved slightly from 75.24% and 77.12% to 75.39% and 78.33%, respectively, when the dimension of the vector space was reduced from 90000 to 1200 by means of distributional clustering. However, the performance of naïve Bayes dropped when terms were clustered. This might have been due to the fact that naive Bayes is more sensitive to term distributions which might be distorted by term clustering.



	Rocchio		NB		kNN	
	Recall	Precision	Recall	Precision	Recall	Precision
政治	72	81	77	80	88	72
經濟	71	78	76	75	73	79
文教	78	67	78	67	72	72
社會	73	90	75	87	72	92
體育	71	91	74	89	81	84
財政	83	60	83	60	89	54
軍事	82	53	77	56	45	76
交通	80	70	75	72	76	81
股市	96	91	93	95	96	96
社福	78	80	75	82	72	85
農業	76	76	72	84	78	80
宗教	62	32	47	43	45	60

Figure 7 *Recall(%)/precision(%) comparison(n=90000).*



	Rocchio		NB		kNN	
	Recall	Precision	Recall	Precision	Recall	Precision
政治	70	83	71	78	83	79
經濟	71	78	76	72	71	80
文教	77	68	78	61	74	71
社會	76	88	72	85	76	91
體育	75	91	67	92	82	85
財政	87	52	76	65	90	53
軍事	84	56	77	54	62	70
交通	79	71	73	70	83	79
股市	96	92	88	97	97	93
社福	81	79	73	81	79	83
農業	75	75	68	80	77	74
宗教	64	32	57	28	50	56

Figure 8 *Recall(%)/precision(%) comparison(n=1200).*

kNN prefered large classes as its MacroAccuracy result, 73.88%, was the lowest, but its MicroAccuracy result, 77.12%, was the best, as indicated in Figure 7. For highly related classes, kNN may prefer a larger class as the probability that the *k* nearest neighbors will belong to the larger class is higher. kNN achieved much better recall results than Rocchio for the class Politics (政治), which was the largest class in our news collections. However, Rocchio achieved much better recall results than kNN did for the class Military (軍事). Note that the class Politics (政治) and the class Military (軍事) were highly correlated, as observed in [17], and that the class Politics (政治) was 5 times larger than the class Military (軍事).

In practical implementation, Rocchio could be a good choice. Rocchio is quite time and memory efficient because the time and memory requirements for the classification process are proportional to the number of classes. However, the time and memory requirements for kNN are proportional to the number of training documents. Rocchio is more noise tolerant than kNN and NB, as shown by the fact that the performance of kNN and NB worsened but the performance of Rocchio improved when *n* was changed from 90000 to 120000, as shown in Figure 3. Rocchio produced slightly worse MicroAccuracy results than kNN did, but can be improved to produce results approaching the performance of kNN by taking more than one representative to represent each class in [17].

7. Conclusions

In this paper, we have proposed and evaluated approaches to categorizing Chinese texts, which consist of term extraction, term selection, term clustering and text classification. For term extraction, we have proposed an approach based on String B-trees. It is scalable and is capable of handling very large numbers of text collections. We use the χ^2 statistic to perform term selection and use distributional clustering to perform term clustering to reduce the dimension of the vector space. Although many redundant terms are identified as significant terms during the term extraction process, the combination of term selection and term clustering somehow can compensate for this drawback by either filtering them out or clustering them into the group containing their original terms. Results of an experiment on a CNA news collection shows that the dimension could be reduced from 90000 to 1200 while approximately the same level of classification accuracy was maintained. We have also studies and compared the performance of three well known classifiers, the Rocchio linear classifier (Rocchio), naive Bayes (NB) probabilistic classifier and k-nearest neighbors (kNN) classifier, when they were applied to categorize Chinese texts. Overall, kNN achieved the best accuracy, about 78.3%, but required large amounts of computation time and memory to classify new texts. Rocchio was very time and memory efficient, and achieved accuracy of about 75.4%. In practical implementation, Rocchio may be a good choice. In addition, we have recently shown [17] that the performance of the Rocchio linear classifier can be improved to approximate that of kNN by taking multiple representative vectors to represent one class.

Acknowledgements

We would like to thank Dr. Chien, Lee-Feng and Mr. Lee, Min-Jer for kind help in gathering the CNA news articles.

References

[1] Central News Agency. http://www.can.com.tw/index.html

[2] Douglas Baker and Kachites McCallum. “Distributional clustering of words for text classification.” *In Proceedings of the 21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval* (SIGIR’98), pages 96-103. 1998.

[3] Chen Chun-Liang and Lee-Feng Chien. “PAT-tree-based online corpus collection and classification.” *In The Fourth International Workshop on Information Retrieval with Asian Languages*(IRAL’99), pages 78-82. 1999.

[4] Chien Lee-Feng. “PAT-Tree-Based keyword extraction for Chinese information retrieval.” *In Proceedings of the 20th Ann Int ACM SIFIR Conference on Research and Development in Information Retrieval*(SIGIR’97), pages 50-58. 1997.

[5] Chien Lee-Feng and Hsiao-Tieh Pu. “Important issues on Chinese information retrieval.” *In Computation Linguistics and Chinese Language Processing*, pages 205-221. 1996.

[6] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W.Furnas, and R.A. Harshman. “Indexing by latent semantic analysis.” *Journal of the American Society for Information Science*, 41(6):391-407. 1990.

[7] Paolo Ferragina and Roberto Grossi. “The String B-tree: A new data structure for string search in external memory and its application.” *Journal of ACM*, 46(2):236-280. 1999.

[8] William B.Frakes and Rick Kazman. Information Retrieval Data Structures Algorithm. Prentice Hall, Englewood Cliffs, New Jersey 0732. 1992.

[9] Marko Frobelink and Dunja Mladenic. “Turning yahoo into an automatic web-page classifier.” *In Proceedings of the 13th European Conference on Aritficial Intelligence*, pages 473-474. 1998.

[10] Huang Sen-Yuan, Yi-Ling Chou, and Ja-Chen Lin. “Automatic classification for news written in Chinese.” *Computer Processing of Oriental Languages*, 12(2):143-159. 1998.

[11] Leonard Kaufman and Peter J. Rousseeuw. “Finding Groups in Data Analysis : An Introduction to Cluster Analysis.” John Wiley and Sons,Inc., New York. 1990.

[12] David D. Lewis, Robert E. Schapire, James P. Callan, and Ron Papka. “Training algorithms for linear text classifiers.” *In Proceedings of the 19th Ann Int ACM SIFIR Conference on Research and Development in Information Retrieval* (SIGIR’96), pages 298-306. 1996.

[13] Lin Yih-Jeng, Ming-Shing Yu, Shyh-Yang Hwang, and Ming-Jer Wu. “A way to extract unknown words without dictionary from Chinese corpus and its applications.” *In Research on Computational Linguistics Conference* (ROCLING XI), pages 217-226. 1998.

[14] Tom M. Mitchell. Machine Learning. The McGraw-Hill Companies, Inc. 1997 .

[15] Amitabh Kumar Singhal. “Term Weighting Revisited. “ *PHD theses*, Cornell University. 1997.

[16] Tsay Jyh-Jong and Jing-Doo Wang. “Term selection with distributional clustering for Chinese text categorization using n-grams.” *In Research on Computational Linguistics Conference XII*, pages 151-170. 1999.

[17] Tsay Jyh-Jong and Jing-Doo Wang. “Improving automatic Chinese text categorization by error correction.” *In The Fifth International Workshop on Information Retrieval with Asian Languages*(IRAL2000), pages 1-8. 2000.

[18] Jyh-Jong Tsay and Jing-Doo Wang. "A scalable approach for Chinese term extraction." *In 2000 International Computer Sympoyium*(ICS2000), Taiwan, R.O.C, pages 246-253. 2000.

[19] R.C.Yang. *The Thesaurus of Daily Wordings*. Book-Spring Publishing Company, Taiwan. 1995.

[20] Yang Yiming and Xin Liu. "A re-examination of text categorization methods." *In Proceedings of the 22th Ann Int ACM SIFIR Conference on Research and Development in Information Retrieval*(SIGIR'99),pages 42-49. 1999.

[21] Yang Yiming and Jan O.Pedersen. "A comparative study on feature selection in text categorization." *In Proceedings of the Fourteenth International Conference on Machine Learning*(ICML'97), pages 412-420. 1997.

[22] Yang Yun-Yan. "A study of document auto-classification in mandarin Chinese." *In Research on Computational Linguistics Conference*(ROCLING VI), pages 217-233. 1993.