Improving Linear Classifier for Chinese Text Categorization
對於中文文件分類線性分類器的改進

蔡志忠
Jyh-Jong Tsay
Dept. of Comp. Sci.
Natl. Chung Cheng Univ.
tsay@cs.ccu.edu.tw

王經篤*
Jing-Doo Wang
Dept. of Comp. Sci.
Natl. Chung Cheng Univ.
jdwang@cs.ccu.edu.tw

摘要

在本論文中，我們對每個類別增加代表點的數目，來彌補線性分類器只用一個代表點，表示一個類別的潛在缺點。為了顯示我們方法的效能，我們和*Rcchio*線性分類器與k個最近鄰近點分類器作比較，實驗結果顯示，我們提高線性分類器的精準度，且效能和k個最近鄰近點分類器相似，但我們分類所花費時間卻少很多。此外，在找新的代表點的時候，我們亦可提供建議去重整分類架構。

Abstract

In this paper we increase the number of representatives for each class to compensate for the potential weakness of linear classifier which compute one representative for each class. To evaluate the effectiveness of our approach, we compared with linear classifier produced by Rocchio algorithm and the k-Nearest Neighbor(kNN) classifier. Experimental results show that our approach improved linear classifier and achieved micro-averaged accuracy similar to that of k-Nearest Neighbor(kNN), with much less classification time. Furthermore, we could provide a suggestion to reorganize the structure of classes when identify new representatives for linear classifier.

*Lecturer, Department of Computer Science and Engineering, National Penghu Institute of Technology

關鍵詞:資訊存取，線性分類器，文件分類。

Keywords: Information Retrieval, Linear Classifier, Text Categorization.

1 Introduction

Systems for text retrieval, routing, categorization and other IR tasks rely heavily on linear classifiers[6]. The main idea of linear classifier is to construct a prototype vector G as one *representative* for a class C using a training set of documents. To determine whether or not class C is assigned to the request document X, it usually computes the cosine similarity δ between X and G. If δ is greater than a given threshold value, class C is assigned to X. In this study, we assign only the class with the highest δ to X, and the behavior of linear classifier is conceptually like to determine which region a point belongs to in a two dimensional Voronoi diagram[9]. The assumption of one representative per class results in the restriction of hypothesis space stretched by documents to the set of linear separable hyperplane regions[5, 16]. However, it is very difficult to construct a set of hyperplanes to separate classes from each other because the shape of each class is irregular and is hard to predict in the high dimensional vector space.

In this study, we increased the number of representatives for each class to compensate for the potential weakness of linear classifiers that compute one representative for each class. First, we classify the documents in the training set with the representatives derived from the original classes. Secondly, we partition the classified documents which are classified into the same class into s partitions via hypergraph partition package[4], where s is determined manually in this study. Thirdly, we find new representatives derived from the subclasses which consist of the miss-classified documents and the correct-classified documents. Then, we select the representatives of the subclasses whose classification precision evaluated by the validation set is greater than a given threshold. Finally, we classify the documents in the testing set with both these new representatives and those derived from the original classes. Note that the training data are divided into a training set and a validation set to avoid the overfitting problem[8]. The training set is used to find the representatives of the original classes and the validation set is used to choose useful representatives of the subclasses whose classification precision are greater than a given threshold.

To evaluate the effectiveness of our approach, we compared it with the linear classifier produced by Rocchio algorithm [2, 6], and the k-Nearest Neighbor(kNN) classifier[5, 16]. Experimental results show that the micro-averaged accuracy of our approach is better than that of linear classifier, and is similar to that of kNN, with much less classification time. Note that kNN is a well-known statistical approach, and is one of the best performers in text categorization[17]. Furthermore, we could observe the ambiguities between classes after the process of new representatives identification, and could provide a suggestion to reorganize the structure of classes in the future.

The remainder of this paper is organized as follows. Section reviews linear classifier. Section describes our approaches. Section gives experimental results. Section gives conclusion and discussion.

2 Linear Classifier

Linear classifier is a simple approach for classification[6]. The main idea of linear classifier is to construct a feature vector as one representative for each class(category). For each class C_i, linear classifier computes prototype vector $G_i = (g_{i,1}, \ldots, g_{i,n})$, where n is the dimension of the vector space, and each element $g_{i,j}$ corresponds to the weight of the jth feature of G_i. The elements in vector G_i are learned from positive examples tuned by negative examples. Positive examples are those documents belonging to that class while negative examples are those documents not belonging to that class. 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 follows:

$$CosSim(X, G_i) = \frac{\sum_{j=1}^{n} x_j \cdot g_{i,j}}{\sqrt{\sum_{j=1}^{n} x_j^2}\sqrt{\sum_{j=1}^{n} g_{i,j}^2}}$$

In this study, we use the Rocchio algorithm[6] to construct the representative G_i for class C_i. Let W be a document in the training collection, represented as a vector $(w_1, w_2, \cdots, w_n)$, where w_j is the weight assigned to the jth term. To determine w_j, we use the TF-IDF weighting method[10], which has been shown to be effective when used in the vector space model. Let tf_j be the term frequency of the jth term in document W, and let df_j be the document frequency of the jth term in training collection. In this study, the TF-IDF weight is defined as $d_j = \log_2(tf_j + 1) * \log_2(\frac{|D|}{df_j})$, where D is the set of documents in the training collection and $|D|$ is the number of documents in D. Let P and N be the set of positive and negative examples with respect to class C_i in the training corpus. $|P|$ and $|N|$ are the number of examples in P and N, respectively. The prototype vector G_i is defined as follows:

$$G_i = \frac{\sum_{W \in P} W}{|P|} - \eta \frac{\sum_{W \in N} W}{|N|}$$

where η is the parameter that adjusts the relative impact of positive and negative examples. In this study, we choose $\eta = 0.25$ according to the experiments in [13, 11].

3 Our Approach

In this study, we increase the number of representatives for each class to compensate for the potential weakness of linear classifier which compute one representative for each class. We give an outline of our approach as follows.

step 1. compute the representatives of the original classes with documents in the training set.

step 2. classify documents in the training set with the representatives computed in step 1.

step 3. identify the subclasses by partitioning the documents which are classified into the same class in step 2 into s partitions, where s is determined manually in this study.

step 4. compute the representatives of the subclasses identified in step 3.

step 5. classify documents in the validation set with the representatives computed in step 4.

step 6. select the representatives of the subclasses whose classification precision achieved in step 5 is greater than a given threshold.

Step 1, 2 and 5 are standard processes of linear classifier as described in Section . In step 3, we obtain the subclasses by partitioning the documents which are classified into the same class in step 2 into s partitions, where s is determined manually in this study. In step 4, we modify the Rocchio algorithm to compute the representatives of the subclasses. In step 6, we select the representatives of the subclasses according to their classification precision achieved in step 5. We next explain the details of step 3, 4 and 6 in Section , Section and Section , respectively.

3.1 Definitions and Notations

We give definitions and notations for the identification of subclasses as follows. Let C be the set of predefined classes, and $|C|$ be the number of predefined classes. Let C_i be the set of documents in the training set that belong to the ith class, and F_j be the set of documents in the training set which are classified to the jth class. $|C_i|$ and $|F_j|$ are the number of documents in C_i and F_j, respectively. Let $H_{i,j}$ be the set of documents in C_i that is classified to F_j. That is, $H_{i,j} = C_i \cap F_j$. Let $h_{i,j} = |H_{i,j}|$. Note that $C_i = \bigcup_{j=1}^{j=|c|} H_{i,j}$ and $F_j = \bigcup_{i=1}^{i=|c|} H_{i,j}$. The confusion matrix $H = (h_{i,j})$, as shown in Table 1, consists of the statistics of the classified documents in the training set. We identify the subclasses by dividing F_j into s partitions as F_j^1,F_j^2, ..., F_j^s. Define $H_{i,j}^r = H_{i,j} \cap F_j^r$ and $h_{i,j}^r = |H_{i,j}^r|$, as shown in Table 2.

3.2 Subclass Identification

We isolated the subclass $H_{i,j}^r$ to form a new representative in step 3. We describe the process of the identification of the subclasses in detail as follows.

step 3.1 transfer the documents in F_j into a hypergraph such that a vertex v represents one document and a hyperedge e represents the set of documents in which term t appears.

step 3.2 partition the vertices(documents) in the hypergraph constructed in step 3.1 into s partitions, where s is determined manually.

step 3.3 gather the vertices(documents) which belong to C_i and are classified to F_j and are in the rth partition as a new subclasses $H_{i,j}^r$.

step 3.4 compute the representatives of those subclasses constructed in step 3.3 using the formula of subclass representative described in Section .

In step 3.1, we constructed a hypergraph in which a vertex v represents one document and a hyperedge e represents the set of documents in which term t appears. The weight

$$
\begin{array}{c c}
 & \begin{array}{cccccc} C_1 & C_2 & \cdots & C_j & \cdots & C_{|c|} \end{array} \\
\begin{array}{c} C_1 \\ C_2 \\ \vdots \\ C_i \\ \vdots \\ C_{|C|} \end{array} &
\begin{pmatrix}
h_{1,1} & h_{1,2} & \cdots & h_{1,j} & \cdots & h_{1,|c|} \\
h_{2,1} & h_{2,2} & \cdots & h_{2,j} & \cdots & h_{2,|c|} \\
\vdots & \vdots & & \vdots & & \vdots \\
h_{i,1} & h_{i,2} & \cdots & h_{i,j} & \cdots & h_{i,|c|} \\
\vdots & \vdots & & \vdots & & \vdots \\
h_{|c|,1} & h_{|c|,2} & \cdots & h_{|c|,j} & \cdots & h_{|c|,|c|}
\end{pmatrix}
\end{array}
$$

Table 1: The confusion matrix H.

$H_{1,j}$	$H^1_{1,j}$	$H^2_{1,j}$	$\cdots$	$H^r_{1,j}$	$\cdots$	$H^s_{1,j}$
$H_{2,j}$	$H^1_{2,j}$	$H^2_{2,j}$	$\cdots$	$H^r_{2,j}$	$\cdots$	$H^s_{2,j}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$H_{i,j}$	$H^1_{i,j}$	$H^2_{i,j}$	$\cdots$	$H^r_{i,j}$	$\cdots$	$H^s_{i,j}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$H_{\|c\|,j}$	$H^1_{\|c\|,j}$	$H^2_{\|c\|,j}$	$\cdots$	$H^r_{\|c\|,j}$	$\cdots$	$H^s_{\|c\|,j}$
	F^1_j	F^2_j	$\cdots$	F^r_j	$\cdots$	F^s_j

Table 2: Partition F_j into s partitions.

of the hyperedge e is determined by $tf \cdot idf$ of term t[10] and is defined as follows.

$$Weight(t) = \log_2(tf+1) * \log_2(\frac{|D|}{df})$$

where $|D|$ is the number of training documents; tf is the term frequency of term t and df is the document frequency of term t in training collection. In step 3.2, we partition the vertices(documents) of a hypergraph into s roughly equal parts using hypergraph partition package[4] such that the total weight of hyperedges connecting vertices in different parts was minimized. Note that the hypergraph partitioning algorithm is an effective and scalable clustering method. Intuitively, the documents which had common terms would be clustered together[14]. In step 3.3, as shown in Table 2, we identified the subclass $H_{i,j}^r$ that belonged to the $H_{i,j}$ and was in the rth partition of F_j. In this study, we only take the subclass whose $h_{i,j}^r >= 5$ into consideration.

3.3 Subclass Representative

We modified the Rocchio algorithm to construct the representatives derived from the subclasses which were identified in Section . As shown in Figure 1, the documents in the training set D consists of $C_1, \cdots, C_k$ and each class C_i consists of $|c| * s$ subclasses at most before representative qualification described in Section . We describe the modification of the Rocchio algorithm to construct the representative of the subclass $H_{i,j}^r$ in the following.

As shown in Figure 1, let P be the set of documents that belong to the subclass $H_{i,j}^r$, and $P' = C_i - H_{i,j}^r$ be the set of documents that belong to class C_i but do not belong to subclass $H_{i,j}^r$. Let $N = D - C_i$ be the set of documents in the training set D that does not belong to class C_i. The representative $G_{i,j}^r$ of subclass $H_{i,j}^r$ was given as follows.

$$G_{i,j}^r = \alpha \frac{\sum_{W \in P} W}{|P|} - \beta \frac{\sum_{W \in P'} W}{|P'|} - \eta \frac{\sum_{W \in N} W}{|N|}$$

In this study, we chose $\alpha = 1$, $\beta = 0$ and $\eta = 1$. Note that we chose β as 0 in above equation because we used the representative $G_{i,j}^r$ of the subclass $H_{i,j}^r$ to distinguish class C_i from the other classes $C_j (i \neq j)$, but didn't use that representative to distinguish the $H_{i,j}^r$ from the other subclasses which derived form class C_i.

3.4 Representative Qualification

In this study, we selected those new representatives whose classification precision evaluated by the validation set is greater than a given threshold θ. We classified the documents in the validation set with the representatives obtained in Section and computed the precision of each representative. Then, we selected the representatives whose classification precision evaluated by the validation set was greater than θ. The choice for the value of θ was according to the micro-level accuracy θ^1 achieved by the linear classifier in the testing set. In this study, we had $\theta > \theta^1$ in order to achieve higher precision and better performance than that linear classifier did.

4 Experiments

4.1 Data Source

In our experiment, we used Chinese news articles from the Central News Agency(CNA). 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. To avoid overfitting problem[8], the training data are partitioned into a training set and a validation set. In this study, the training set consists of two-thirds of the training data and the validation set consists of the remaining data. All the news articles were preclassified into 12 classes, as listed in Table 3.

4.2 Document Representation

The representation of Chinese texts consists of the following steps: term extraction, term

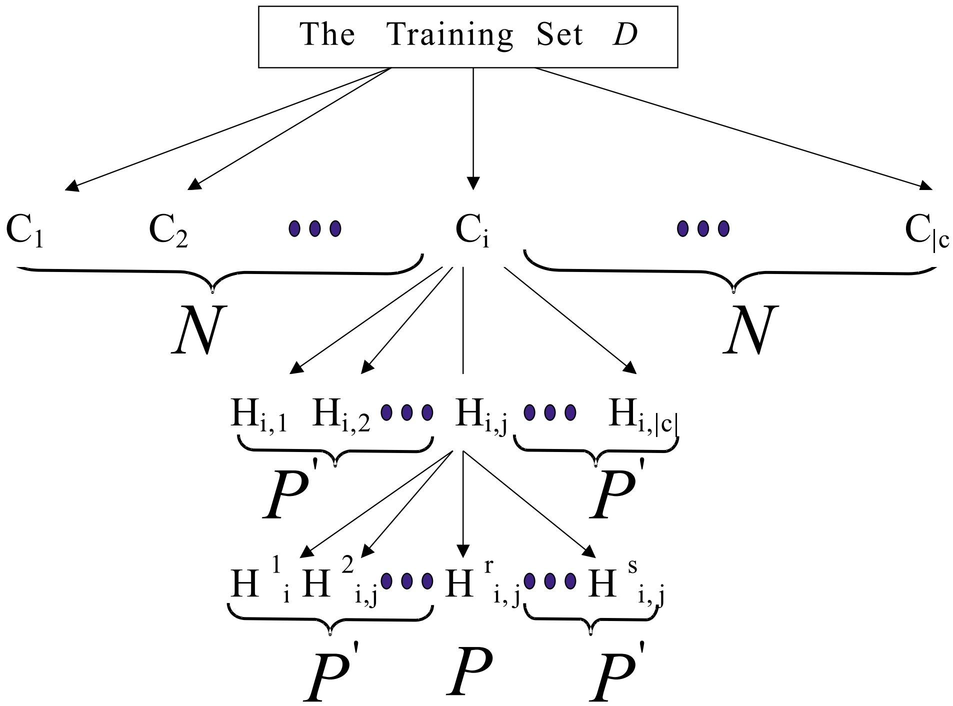

Figure 1: The modification of the Rocchio algorithm for the subclass $H^r_{i,j}$.

		Train Data		Test Data
		1991/8-1992/1		1992/2/1-2/28
	CNA News Group	Training Set	Validation Set	Test Set
C_1	cna.politics.*	8988	4494	1225
C_2	cna.economics.*	3846	1922	776
C_3	cna.transport.*	1200	601	279
C_4	cna.edu.*	2136	1067	379
C_5	cna.l*	1852	926	415
C_6	cna.judiciary.*	2088	1044	492
C_7	cna.stock.*	1186	593	200
C_8	cna.military.*	1212	606	261
C_9	cna.argriculture.*	997	499	238
C_{10}	cna.religion.*	471	236	74
C_{11}	cna.finance.*	1306	652	151
C_{12}	cna.health-n-welfare.*	1158	580	305
	Total	**26440**	**13220**	**4795**

Table 3: CNA news statistics.

selection and term clustering. In term extraction, we adopt a scalable approach[15] to extract significant terms, which is based on String B-trees(SB-trees)[3]. In term selection, we adopt χ^2 statistics[18] to select the most representative terms from the extracted terms. In term clustering, terms which are highly correlated are clustered into the same group. Distributional clustering[1] can reduces the dimension of the vector space to a practical level for Chinese text categorization[13, 12].

In our experiment, we use one year news, 1/1/1991-12/31/1991, to extract *Chinese frequent strings*(CFS)[7] and the number of significant terms extracted is 548363. We select 90000 of the extracted terms, and then group them into 4800 clusters because the choice of 90000 and 4800 achieves the best performance as indicated in [13, 11]. Therefore, each document D_i is transformed into a vector as $(d_{i,1}, \ldots, d_{i,n})$, where n is $4,800$ and $d_{i,j}$ is the $tf \cdot idf$ weight[10] of the jth term in D_i.

4.3 Performance Measures

We measure the classification accuracy at both micro and macro levels. Three performance measures are 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 that are preclassified to the ith 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 in C_i that are classified to C_j. Let $Acc(i) = |H_{i,i}|/|C_i|$ 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. Note that we measured the classification time on a PC with Pentium III 450(CPU) and 192MB RAM.

In order to discuss the biased situation that some classifiers prefer large classes than small classes, we also adopt the performance measures, recall, precision and F_1 measure. Recall(R) is the percentage of the documents for a given class(category) that are classified correctly. Precision(P) is the percentage of the classified documents for a given class that are classified correctly. The F_1 measure is one of the common measures to combine the recall and precision, and is defined as $F_1 = \frac{2RP}{(R+P)}$.

4.4 Experimental Results

4.4.1 Improving Linear Classifier

First of all, as shown in Table 4, we had the confusion matrix H by preclassifying the documents in the training set. Secondly, we identified the subclass $H_{i,j}^r$ by partitioning the documents in class F_j which were classified to the jth class into s partitions, where s was determined manually and the value s experimented with included 2, 4, 8, 16, 32, 64, 128 and 256 in this study. As shown in Table 5, we isolated 1017 subclasses before representative qualification when $s = 64$. Note that we only took those subclasses whose $h_{i,j}^r$ were greater than or equal to 5 into consideration in this study. Thirdly, we used the representatives of the identified subclasses to classify the documents in the validation set, and had representative qualification by selecting the representatives whose precision was greater than a given threshold 80%, where the value of 80% was according to the MicroAccuracy, about 75%, achieved by the Rocchio linear classifier in this study. Finally, we classified the news in the testing set with the representatives which consisted of qualified representatives and those derived from original classes. The comparison of the performance with respect to different value s is shown in Table 6. The best MicroAccuracy our approach achieved was 77.54% when $s = 64$, and its corresponding MacroAccuracy and AccuracyVariance were 77.22% and 75.30, respectively, and the number of representatives was 580, 568 derived from subclasses and 12 derived form original class. We chose the case when $s = 64$ for further discussions.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}
C_1	6697	277	104	221	47	231	25	874	94	287	42	89
C_2	263	3040	69	36	9	26	46	34	88	20	164	50
C_3	13	33	1051	13	6	17	2	15	19	7	10	14
C_4	99	53	23	1684	57	15	7	38	11	84	4	60
C_5	40	17	44	209	1441	10	3	12	16	38	3	19
C_6	126	26	81	27	14	1615	1	45	48	43	7	55
C_7	5	26	1	1	2	2	1094	0	2	2	50	1
C_8	56	6	8	15	1	10	3	1095	3	10	0	5
C_9	28	96	11	8	6	7	6	3	800	7	4	22
C_{10}	48	2	5	14	1	6	0	7	2	369	1	17
C_{11}	47	103	10	4	0	18	18	2	7	4	1089	4
C_{12}	22	14	11	15	5	8	4	5	13	28	6	1027

Table 4: The confusion matrix H : the statistics of the classified news in the training set.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	# of subclasses
C_1	64	29	6	17	1	25	2	64	3	29	2	5	247
C_2	24	64	1	0	0	0	2	1	2	0	11	1	106
C_3	0	1	64	0	0	0	0	0	0	0	0	0	65
C_4	3	1	0	64	0	0	0	1	0	2	0	0	71
C_5	0	0	1	16	64	0	0	0	0	0	0	0	81
C_6	6	0	2	0	0	64	0	0	0	0	0	1	73
C_7	0	0	0	0	0	0	64	0	0	0	0	0	64
C_8	0	0	0	0	0	0	0	64	0	0	0	0	64
C_9	0	3	0	0	0	0	0	0	64	0	0	0	67
C_{10}	0	0	0	0	0	0	0	0	0	46	0	1	47
C_{11}	1	3	0	0	0	0	0	0	0	0	64	0	68
C_{12}	0	0	0	0	0	0	0	0	0	0	0	64	64
												Total	1017

Table 5: The distribution of subclasses(s=64).

	s : the number of partitions							
	2	4	8	16	32	64	128	256
MicroAccuracy	75.68	76.02	76.23	76.27	76.18	77.54	77.02	76.53
MacroAccuracy	77.63	77.28	76.36	75.88	75.27	77.22	77.17	76.69
AccuracyVariance	75.44	82.97	108.72	108.56	88.76	75.30	84.62	80.82
# of representatives	39	60	130	227	349	580	906	1052
Classification Time	00:06:13	00:06:18	00:06:38	00:06:58	00:07:34	00:08:56	00:11:55	00:13:59

Table 6: The comparison of different number of partitions.

4.4.2 Overall Comparison

To evaluate the effectiveness of our approach, we compared with the linear classifier produced by Rocchio algorithm and the k-Nearest Neighbor(kNN) classifier. Our approach improved linear classifiers and achieved the MicroAccuracy similar to that of kNN did, with much less classification time. Our approach also avoided the biased situation[14] that prefers large classes than small classes.

We briefly describe kNN classifier for completeness as follows. 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 the X. The similarity score of each neighbor document to the X is used as the weight of the class of the neighbor document, and the sum of class weights over the k nearest neighbors are used for class ranking[16]. Note that kNN is a well-known statistical approach, and is one of the best performers in text categorization[17]. We have performed an experiment using different values of k, including 5, 10, 15, 20, 30, 50, 100 and 200. The best choice of k in our experiment is 50.

As shown in Table 7, the value 77.54% of MicroAccuracy our approach achieved was better than the value 75.20% of that Rocchio did, and was similar to the value 77.62 of that kNN did; the MacroAccuracy and AccuracyVariance our approach achieved were similar to that the Rocchio did, and are better than that kNN did. Furthermore, the classification time of our approach, about 9 minutes, was much less than that of kNN, about 1 hours and 29 minutes. On the other hand, as shown in Table 8, most of the values of F_1 measure our approach achieved were better than or equal to that Rocchio did, except class C_8. That is, our approach improved the performance of linear classifier while avoided the biased situation[14] that prefers large classes than small classes.

4.4.3 Suggestions for Reorganizing Class Structure

We might provide a suggestion to reorganize the structure of classes with the representatives whose classification precision evaluated in the validation was low. We could observe the ambiguities between classes due to the characteristic of linear classifier via those subclasses whose representatives achieved low precision in the validation set. As shown in Table 9, there were the distribution of the number of the subclasses whose precision was lower than 50%. Class C_1(Politics), for example, was a confused class that highly corrected with the other classes because there were 45 representatives derived from class C_1 to distinguish from the other classes but failed to pass representative qualification. Class C_2(Economics) highly correlated with class C_{11}(Finance) as there were 11 representatives derived from class C_2 to distinguish from class C_{11} as shown in Table 5, but there were 5 representatives as shown in Table 9. Similarly, there were 3 representatives derived from C_{11} to distinguish from C_2 as shown in Table 5, but all failed to pass representative qualification as shown in Table 9.

5 Conclusions

In this paper, we have improved linear classifiers by increasing the number of representatives for each class to compensate the potential weakness of linear classifier which compute one representative for each class. We identify new representatives derived from the subclasses which are respectively isolated from the miss-classified documents and the correct-classified documents via hypergraph partition package. Then, we select the representatives of the subclasses whose classification precision evaluated by the validation set is greater than a given threshold. Finally, we classify the documents in the testing set with the representatives which consist of these new representatives and those derived from original classes. To evaluate the effectiveness of our approach, we have compared with linear classifier produced by Rocchio algorithm and the k-Nearest Neighbor(kNN) classifier. Experimental results show that our approach improves linear classifier and achieves the MicroAccuracy similar to that

	Rocchio	Our Approach	kNN
MicroAccuracy	75.20	77.54	77.62
MacroAccuracy	77.46	77.22	75.78
AccuracyVariance	74.97	75.30	152.88
# of representatives	12	580	39960
Classification Time	00:06:07	00:08:56	01:28:48

Table 7: Performance comparison.

	Rocchio			Our Approach			kNN		
	Precision	Recall	F_1	Precision	Recall	F_1	Precision	Recall	F_1
C_1	82	70	0.755	79	82	0.805	74	86	0.796
C_2	78	70	0.738	78	72	0.749	79	69	0.737
C_3	69	80	0.741	74	81	0.773	78	80	0.790
C_4	68	78	0.727	73	75	0.740	72	76	0.739
C_5	91	74	0.816	90	76	0.824	88	80	0.838
C_6	88	75	0.810	88	75	0.810	92	72	0.808
C_7	91	96	0.934	93	96	0.945	92	97	0.944
C_8	55	83	0.662	68	64	0.659	73	57	0.640
C_9	74	75	0.745	74	79	0.764	76	77	0.765
C_{10}	35	61	0.445	44	62	0.515	61	50	0.550
C_{11}	52	87	0.651	53	85	0.653	54	88	0.669
C_{12}	80	80	0.800	81	79	0.800	85	77	0.808
Average			0.735			0.753			0.757

Table 8: Precision(%)/recall(%)/F_1 measure Comparison.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	# of subclasses
C_1	0	14	4	4	1	7	1	10	0	4	0	0	45
C_2	12	0	1	0	0	0	1	0	0	0	5	0	19
C_3	0	1	6	0	0	0	0	0	0	0	0	0	7
C_4	2	0	0	2	0	0	0	1	0	2	0	0	7
C_5	0	0	1	6	3	0	0	0	0	0	0	0	10
C_6	5	0	2	0	0	1	0	0	0	0	0	0	8
C_7	0	0	0	0	0	0	11	0	0	0	0	0	11
C_8	0	0	0	0	0	0	0	13	0	0	0	0	13
C_9	0	2	0	0	0	0	0	0	10	0	0	0	12
C_{10}	0	0	0	0	0	0	0	0	0	18	0	0	18
C_{11}	1	3	0	0	0	0	0	0	0	0	12	0	16
C_{12}	0	0	0	0	0	0	0	0	0	0	0	3	3
												Total	169

Table 9: The distribution of the number of the representatives whose precision $< 50\%$.

of k-Nearest Neighbor(kNN) did, but takes much less classification time. Our approach also avoids the biased situation that prefers large classes than small classes. Furthermore, we might provide a suggestion to reorganize class structure via the subclasses whose representatives achieved low precision in the validation set.

Acknowledgment. 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] 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.

[2] Willian W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. In *Proceedings of the 19th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR'96)*, pages 307–315, 1996.

[3] 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.

[4] George Karypis and Vipin Kumar. hmetis : A hypergraph partitioning package. Technical report, University of Minnesota, Department of Computer Science and Engineering, 1998.

[5] Wai Lam. Using a generalized instance set for automatic text categorization. In *Proceedings of the 21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR'98)*, pages 81–89, 1998.

[6] 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 SIGIR Conference on Research and Development in Information Retrieval(SIGIR'96)*, pages 298–306, 1996.

[7] Yih-Jeng Lin, 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.

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

[9] Ketan Mulmuley. *Computational Geometry : An Introduction Through Randomized Algorithm*. Prentice Hall, 1994.

[10] Amitabh Kumar Singhal. *Term Weighting Revisited*. PhD thesis, Cornell University, 1997.

[11] Jyh-Jong Tsay and Jing-Doo Wang. Comparing classifiers for automatic Chinese text categorization. In *1999 National Computer Symposium, Taiwan, R.O.C*, pages B–274–B–281, 1999.

[12] Jyh-Jong Tsay 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.

[13] Jyh-Jong Tsay and Jing-Doo Wang. Design and evaluation of approaches for automatic chinese text categorization. *International Journal of Computational Linguistics and Chinese Language Processing(CLCLP)*, 5(2):43–58, August 2000.

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

[15] 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.

[16] Yiming Yang. An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*, 1(1/2):67–88, 1999.

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

[18] Yiming Yang 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.