An enhanced text categorization method based on improved text frequency approach and mutual information algorithm*

Pei Zhili[1,2], Shi Xiaohu[1], Maurizio Marchese[3] and Liang Yanchun[1,3]**

(1. Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, College of Computer Science and Technology, Jilin University, Changchun 130012, China; 2. College of Mathematics and Computer Science, National University of Inner Mongolia, Tongliao 028043, China; 3. Department of Information and Communication Technology, University of Trento, Via Sommarive 14, 38050-Povo (TN), Italy)

Accepted on July 5, 2007

Abstract Text categorization plays an important role in data mining. Feature selection is the most important process of text categorization. Focused on feature selection, we present an improved text frequency method for filtering of low frequency features to deal with the data preprocessing, propose an improved mutual information algorithm for feature selection, and develop an improved tf.idf method for characteristic weights evaluation. The proposed method is applied to the benchmark test set Reuters-21578 Top10 to examine its effectiveness. Numerical results show that the precision, the recall and the value of F1 of the proposed method are all superior to those of existing conventional methods.

Keywords: text categorization, mutual information, feature selection, characteristic weights, classifier.

With the rapid development of computer science and communication technology, people can acquire more and more digitalized information, and consequently, more and more time is needed for managing this information. In this context, the development of automatic methods for text categorization is becoming a critical issue. Text categorization is the process of automatically determining text categories according to text content within a given taxonomy system[1—4]. It is an important research field of text mining, which is integrated by artificial intelligence and information retrieval technologies. In the 1990s, the application of statistical methods to this problem has attracted broad attention. In fact, they improved both the accuracy and the stability of the results compared with previous methods. There are five major steps in the statistical methods, namely: (1) reduce the feature's dimension and evaluate the characteristic weights; (2) train the text classifier with the training data; (3) codify the samples to be classified according to the selected features; (4) input the data to the trained classifier; (5) get the categorization of the input samples. The critical step for the overall performance and quality of the results is the first one.

The character dimension of the original vector space is normally huge, which often leads to data sparsity of the text representation. Moreover, the highly dimensional feature set tends to include lots of noises. Therefore, performing the character dimension reduction will greatly impact both the training time and the accuracy. Feature selection is the main method for the character dimension reduction, where a subset of the original feature set is selected using least features to present most characters of the original set[5—8]. The feature selection problem can be addressed in the following two main procedures: (1) given a fixed $m \ll n$ (with n the original feature vector space dimension), find m features that give the smallest expected generalization error; or (2) given a maximum allowable generalization error γ, find the smallest m. Feature selection algorithms are often composed of three components, namely search algorithm, evaluation function and performance function. In this context, the evaluation function plays an important role. It inputs a feature subset and outputs a numerical evaluation. The search algorithm's goal is to maximize this function. Many different evaluation functions have been proposed, such as term frequency (TF), document frequency (DF), term entropy (TE), mutual information (MI), information gain (IG), chi-square (CHI), term strength (TS), expected cross entropy (ECE), weight of evidence for

* Supported by National Natural Science Foundation of China (Grant Nos. 60673023, 60433020 and 10501017), the European Commission for TH/Asia Link/010 (Grant No. 111084) and the Inner Mongolia Natural Science Foundation (Grant No. 200711020807)

** To whom correspondence should be addressed. E-mail: ycliang@jlu.edu.cn

text (WET), and odds ratio (OR). Many researchers have applied and compared different evaluation functions. Their results show that the MI method has some advantages compared with others. However, in MI method, the feature's frequency is not considered, and those features possessing negative correlation to clustering are deleted, and so some special keywords possessing important roles might be removed. Focused on these shortcomings of the MI method, in this paper, we propose a number of improvements which eventually lead to a reduced number of selected features and therefore to more distinguishable classes.

In vector space model (VSM), the context in each document is mapped into a point of a multi-dimensional vector space. Every context could be represented by a vector of the space $d_i = (t_{i1}, w_{i1}; t_{i2}, w_{i2}, \cdots; t_{ik}, w_{ik}, \cdots; t_{im}, w_{im})$, where t_{ik} is the ith word and w_{ik} is the weight of the corresponding t_{ik}, used to indicate the importance of t_{ik}. In this way, the problems of context information representation and matching are transferred into those of vector space operations. For the crucial weight functions, w_{ik}, for the context clustering, many researchers have studied ways to evaluate them: Rocchio proposed a batch computational method based on positive and negative cases[9]; Widrow developed the real-time linear computing method[10]; Soucy proposed a method of computing the confidence interval of feature words to estimate their weight[11]; Franca developed a method of estimating weight of supervised terms[12]; while the most classic and the widely used method is tf.idf (term frequency & inverse document frequency)[13]. There are two main factors that should be considered in the tf.idf method: (1) term frequency (tf), namely the number of a feature in the context; (2) inverse document frequency (idf), which is a quantitative representation of the feature's distribution in the text set. However, the tf.idf method does not distinguish between the occurrences of the terms in different positions.

In this paper, an improved text frequency method for filtering of low frequency features is proposed to deal with data preprocessing, and then an improved mutual information algorithm is developed to perform feature selection. Also an improved tf.idf method is developed for characteristic weights evaluation. To examine the effectiveness of the proposed method, it is applied to the benchmark test set Reuters-21578 Top10. For comparison, the SVM, kNN and Rocchio methods are used as context classifiers. Numerical results show that the precision, the recall and the value of F1 of the proposed method are all superior to those of existing methods.

1 Improved text frequency method for filtering of low frequency features

In general, only a small part of low frequency features are useful for text clustering, whereas most low frequency features are noise data. Some feature selection methods, including the MI algorithm, are too dependent on the low frequency features. This means that many low frequency features of noise are selected into the feature set, to reduce the clustering effectiveness. Therefore, before feature selection, a preprocessing of low frequency features should be performed to filter the low frequency noise. To this end, text frequency is the simplest approach. The method is based on the following basic hypothesis: those text features with lower frequency than a given threshold do not contain or contain relatively little clustering information. Removing these kinds of features from the original feature space can not only reduce dimensions of low feature space, but also increase the accuracy of categorization. The disadvantage of this method is that some low frequency features might not be rare in a certain text type, and contain important clustering information. Based on the text frequency concept, an improved method for filtering of low frequency features is proposed to overcome the above defect:

Step 1: Count $\mathrm{freq}^{t_i} d_k$ (the frequency of feature t_i in text d_k, not equal to zero).

Step 2: Calculate R_{t_i} ($R_{t_i} = \min_k(\mathrm{freq}^{t_i} d_k)/\max_k(\mathrm{freq}^{t_i} d_k)$).

Step 3: Define $\mathrm{cri}(t_i) = (1 + p(t_i) R_{t_i}) \mathrm{DF}_{t_i}$ as the filtering criterion, that is to say, those features with lower $\mathrm{cri}(t_i)$ than a given threshold should be deleted. Here, $p(t_i)$ is the frequency of t_i in the training text set, and DF_{t_i} is the text frequency t_i.

2 Improved mutual information algorithm for feature selection

2.1 Mutual information algorithm

Mutual information (MI) is an important con-

cept in statistics and information theory. It represents the correlation between two statistics variables. The stronger the correlation, the greater the MI is, and vice versa. MI method in the text categorization means to study the correlation between the features and the types. The greater the MI of a feature and a text type is, the more relevant the feature is for that document. During feature selection, the features with greater MI should be selected. The outline of MI algorithm is: compute the MIs between a feature and all the types; select the biggest one as the MI of this feature; set a threshold during the feature selection and save all the features whose MI values are higher than the threshold. The MI of feature t_k and text type c_i could be computed by

$$\mathrm{MI}(t_k, c_i) = \log \frac{p(t_k, c_i)}{p(t_k)p(c_i)} \quad (1)$$

where $p(t_k, c_i)$ is the quotient of the number of t_k in c_i divided by the number of t_k in the whole training set; $p(t_k)$ the quotient of the number of t_k in the training set divided by the number of texts in the whole training set and $p(c_i)$ the quotient of the number of texts belonging to type c_i divided by the number of texts in the whole training set.

When feature t_k relies on only one text type c_i, the MI value $\mathrm{MI}(t_k, c_i)$ is very large; when t_k and c_i are independent, $\mathrm{MI}(t_k, c_i)$ is zero, while when t_k rarely emerges in c_i, $\mathrm{MI}(t_k, c_i)$ is negative, that is to say, they are negatively correlated. The features with lower frequency have greater influence on MI values. Therefore, the low frequency features often have relatively big mutual information. Since MI favors features with low frequency, it can easily lead to over-fitting problem[14].

The mutual information (MI) of feature t_k in the whole training set could be defined as:

$$\begin{aligned}\mathrm{MI}(t_k) &= \sum_{i=1}^{m} \mathrm{MI}(t_k, c_i) \\ &= \sum_{i=1}^{m} p(c_i) \log \frac{p(t_k, c_i)}{p(t_k)p(c_i)} \end{aligned} \quad (2)$$

Generally, in the feature selection process, the features with larger $\mathrm{MI}(t_k)$ values should be selected.

2.2 Improved mutual information algorithm

Examining Eq. (1), several disadvantages could be found:

(i) The frequency of feature t_k in the text type c_i is not considered in Eq. (1): this might lead to the fact that different frequency features will have the same MI value. For example, assume that there are two features A and B, if the probability of feature A in all the texts is 0.8, and in text type c_i is 0.2, and those of B are 0.2 and 0.05, respectively, then A and B have the same MI values according to Eq. (1). But obviously, compared with B, A is more relevant to c_i. Therefore, we propose to revise Eq. (1) as:

$$\mathrm{MI}(t_k, c_i) = p(t_k \mid c_i) \log \frac{p(t_k, c_i)}{p(t_k)p(c_i)} \quad (3)$$

where $p(t_k \mid c_i)$ is the quotient of the number of t_k in c_i divided by all the text numbers in c_i. By introducing the term $p(t_k \mid c_i)$, the importance of A and B could be distinguished. Then those features with even distribution among different types are filtered, while those features with higher frequency in one or several text types but lower frequency in other types are identified and maintained.

(ii) According to Eq. (1), the more the feature t_k relies on only one text type c_i, the larger $\mathrm{MI}(t_k, c_i)$ is, while if t_k rarely emerges in c_i, $\mathrm{MI}(t_k, c_i)$ is negative, namely, they are negatively correlated. In the case that t_k and c_i are negatively correlated, t_k seldom emerges in c_i, but is favor to emerge in the other types. In MI algorithm, only N features with the largest MI values are preserved, while those with negative MI values features are eliminated. The positive correlated features of one text type could represent this text type, while those negative correlated ones of one text type could represent the differences between other types and this type. In this sense, those features with large negative MI values are of the same importance for the text clustering than features with large positive MI values. Therefore, to save those features with large negative MI values, Eq. (3) could be further improved as:

$$\mathrm{MI}(t_k, c_i) = p(t_k \mid c_i) \log \left| \frac{p(t_k, c_i)}{p(t_k)p(c_i)} \right| \quad (4)$$

(iii) A great number of research results show that good feature selection should rely on many parameters, such as frequency, dispersion, and concentration. Frequency is the most commonly used parameter of feature selection. It is assumed that the more frequently a feature emerges in a text type, the more likely it can represent this text type. Therefore, the more frequent features in a text type are selected as the type features representing this type of texts.

Besides, people usually consider those effective features more likely to be emerged in one or several text types, but not in all types evenly. This point could be characterized by the concept of "concentration". Moreover, among the type features of a certain text type, those evenly emerged in the whole type are more effective than those emerge densely in one or several types of texts. The concept of "dispersion" might characterize this point. Obviously, an effective feature is more likely to have the higher frequency, the bigger dispersion, and the greater concentration, while in Eq. (1), only the frequency parameter is considered. This limitation assumes that the scales of the content of all text types are approximately equal. However, this is not always the case. Hence, it can easily lead to the problem of "over-fitting" of a single feature. Therefore, Eq. (4) might be revised by

$$\mathrm{MI}(t_k, c_i) = p(t_k \mid c_i)\log\left|\frac{p(t_k, c_i)}{p(t_k)p(c_i)}\right|\exp(R_k/R) \quad (5)$$

where R_k is the number of feature t_k in text type c_i and R the number of t_k in the whole training set.

Moreover, we can see from Eq. (2), that those features present in more classes will have larger $\mathrm{MI}(t_k)$ values, while those present in less classes will have smaller $\mathrm{MI}(t_k)$ values. Therefore, some features in specific classes are easy to be ignored. To overcome this shortcoming, those features with the kth largest $\mathrm{MI}(t, c_i)$ values should be selected into the feature set $T(t)$, that is,

$$T(t) = \bigcup_{i=1}^{m}\{t \mid \mathrm{MI}(t, c_i) \in \text{the largest } k\text{th } \mathrm{MI}(t, c_i)\} \quad (6)$$

where $\mathrm{MI}(t, c_i)$ is defined by Eq. (5).

The time and space complexity of the improved method is the same as the original one. This feature selection strategy could consider both features belonging to multiple types and those belonging to only one or several types.

To sum up, the feature selection process should be performed in three steps:

① General preprocessing of training data: filtering the inactive words, merging synonyms, and processing conjugate, in order to determine the starting feature set T_1.

② The preprocessing of low frequency features: using the method described in Section 1 to filter the low frequency features with little usages, in order to define feature set T_2.

③ According to Eqs. (5) and (6), selecting an appropriate number features from T_2, to arrive at the final feature set T_3.

3 Improved computing method for the features' weights

3.1 tf.idf method

There are two parts in the tf.idf method, namely, tf (term frequency), the number of the feature in the whole text set, and idf (inverse document frequency), a quantitative representation of the feature's distribution in the text set, commonly computed by $\log_2(N/n_k+0.01)$ (where N is the number of texts in the text set, n_k is the number of the feature). The weight between feature t_k and text d_i could be obtained by

$$w_{ik} = tf_{ik}\times\log_2(N/n_k+0.01) \quad k = 1, 2, \cdots, P \quad (7)$$

where tf_{ik} is the number of t_k in d_i, and P the number of features. For computational convenience, normalization is often performed, therefore, Eq. (7) could be rewritten as:

$$w_{ik} = \frac{tf_{ik}\times\log_2(N/n_k+0.01)}{\sqrt{\sum_{k=1}^{P} tf_{ik}\times\log_2(N/n_k+0.01)}} \quad k = 1, 2, \cdots, P \quad (8)$$

3.2 Improved tf.idf method

Eq. (8) is proposed based on such a consideration: the most effective feature words for text clustering should be those that emerge frequently in some certain text sets while emerge seldom in other text sets. However, according to Eq. (8), the computation of tf_{ik} does not distinguish the differences of the feature words from different positions in the text contents. Our improved method divides text contents into several regions, such as header region, abstract region, and body region. In structured documents (such as news, articles, scientific documents, and reports), the features in the header region can represent more exactly text content than those in the abstract region. Similarly, the features in the abstract region are more effective than those in the body region. Moreover, there is also a difference of the effective-

ness between different features in the same region of the text content. For the above reasons, the computation of tf_{ik} is modified by

$$tf_{ik} = \sum_{j=1}^{3} \alpha_j \times tf_{ik}^{j} \times \log_2(M/m_j) \qquad (9)$$

where $j = 1, 2, 3$ correspond to the feature t_k in the header region, abstract region, and body region, respectively; α_j are the coefficients, and $\alpha_1 > \alpha_2 > \alpha_3 \geqslant 1$; M is the number of t_k in the text, m_j is the number of t_k in the jth region of the text content.

Moreover, though Eq. (8) considers the features' distribution in the text set, it does not consider the shape of the distribution. So the information of words is introduced in the method. It is determined by the gain relation between the information entropy of text set and the conditional entropy of features in the text. Then the gain relation is added into the weight computational equation. Eq. (8) is revised by

$$w_{ik} = \frac{tf_{ik} \times \log_2(N/n_k + 0.01) \times IG_k}{\sqrt{\sum_{k=1}^{m} (tf_{ik})^2 \times [\log_2(N/n_k + 0.01) \times IG_k]^2}} \qquad (10)$$

Here, IG_k is the information gain of t_k, which might be obtained by

$$IG_k = H(D) - H(D \mid t_k) \qquad (11)$$

$H(D)$ is the information entropy of text set D, which can be obtained by

$$H(D) = -\sum_{d_i \in D} (p(d_i) \times \log_2 p(d_i)) \qquad (12)$$

$H(D \mid t_k)$ is the conditional entropy of feature t_k, which can be obtained by

$$H(D \mid t_k) = -\sum_{d_i \in D} (p(d_i \mid t_k) \times \log_2 p(d_i + t_k)) \qquad (13)$$

$p(d_i)$ is the probability of text d_i, and it is computed by[15]

$$p(d_i) = \frac{|\text{wordset}(d_i)|}{\sum_{d_i \in D} |\text{wordset}(d_i)|} \qquad (14)$$

where $|\text{wordset}(d_i)|$ refers to the number of the different features in text d_i. However, Eq. (14) only considers the number of features, so two texts with the same feature but different feature frequencies will be regarded as having the same probability. To overcome this disadvantage, we redefine $p(d_i)$ as:

$$p(d_i) = \frac{\text{wordfreq}(d_i)}{\sum_{d_i \in D} \text{wordfreq}(d_i)} \qquad (15)$$

where $\text{wordfreq}(d_i)$ refers to the sum of feature frequency of all the features in the text d_i.

4 Numerical simulation

4.1 Evaluation methods

Precision, recall and $F1$ value (general categorization rate) are the most commonly used test criteria for the categorization results. They reflect the performance of categorization methods from different perspectives. Their formulas are given in what follows.

The precision of the jth text type is

$$P_j = (l_j/m_j) \times 100\%, \quad j = 1, 2, \cdots, n \qquad (16)$$

where l_j is the number of correct clustered texts in the jth text type, m_j the number of texts being clustered in the jth text type and n the number of classes.

The recall of the jth text type is

$$R_j = (l_j/n_j) \times 100\%, \quad j = 1, 2, \cdots, n \qquad (17)$$

where n_j is the real number of texts in the jth text type.

$F1$ value is the integration of precision and recall, the $F1$ value of the jth text type is

$$F_1 = \frac{P_j \times R_j \times 2}{P_j + R_j} \qquad (18)$$

Since the categorization system includes many text types, we can compute precision, recall the $F1$ value[16] for each of them and for their average, namely the micro- and macro-average. In this study, the macro-average method is applied. Their formulas are as follows:

The macro-precision is

$$\text{Macro}P = \frac{1}{n}\sum_{j=1}^{n} P_j \qquad (19)$$

The macro-recall is

$$\text{Macro}R = \frac{1}{n}\sum_{j=1}^{n} R_j \qquad (20)$$

The macro-$F1$ value is

$$\text{Macro}F_1 = \frac{\text{Macro}P \times \text{Macro}R \times 2}{\text{Macro}P + \text{Macro}P} \qquad (21)$$

4.2 Numerical results

To examine the effectiveness of our improved methods, Rocchio, kNN and SVM methods are used

as the context classifiers, and numerical simulations are performed. Simulations include the data preprocessing algorithm, the feature selection algorithm and the computing method for the weight of features developed in this paper. The benchmark text categorization set of Reuters-21578 is selected for the experimental data. The data set is taken from the news texts of Reuters, which are all about economics, with a total of 21578 texts, under 5 main artificially clustered categories, which are topics, organizations, exchanges, places, and people. There are many subcategories in each principal category. The biggest 10 categories among the total of 135 categories include 10427 samples, and the corresponding dataset is named Reuters-21578 Top10 (Table 1). Since not all of these 10427 documents have been labeled with class number, only 9980 suitable and effective samples have been selected as our experimental data set. 5 fold cross validation method is applied in our simulation.

Table 1. The top 10 Reuters-21578 categories

Class	Earn	ACQ	MoneyFx	Grain	Crude	Trade	Interest	Wheat	Ship	Corn
Document count	3987	2448	801	628	634	551	513	306	305	254

To examine the effectiveness of the proposed method for preprocessing of low frequency features, we perform all the three methods with our proposed method and the conventional text frequency method. Here, the feature selection algorithm and the computational method for the weight of features are all taken as the original ones, namely using Eq. (1) and Eq. (8). Figs. 1—3 show comparisons of the three methods.

From Fig. 1, we can see that the Rocchio classifier takes the maximum accuracy value as 76.8% by using conventional text frequency method when the feature frequency threshold is 20; while it takes the maximum accuracy value as 79.7% by using our proposed method when the threshold is 25. Fig. 2 shows that with the conventional method, the maximum accuracy value of kNN classifier is 79.1% when the threshold is 25, and with the proposed method the maximum accuracy is 81.5% at threshold 30. Fig. 3 shows that with the conventional method, the maximum accuracy value of SVM classifier is 81.9% when the threshold is 20, while with our proposed method it is 83.8% at threshold 25. The results show that our proposed preprocessing method improves the clustering accuracy as compared with the conventional text frequency method. Therefore, filtering the useless low frequency features seems to enhance clustering accuracy, while as the threshold value further increases, the accuracy decreases.

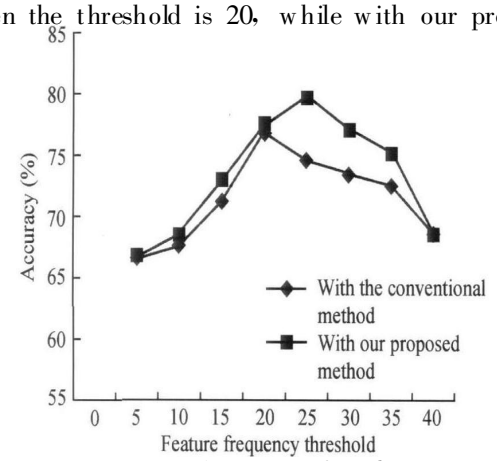

Fig. 1. Comparisons of Rocchio.

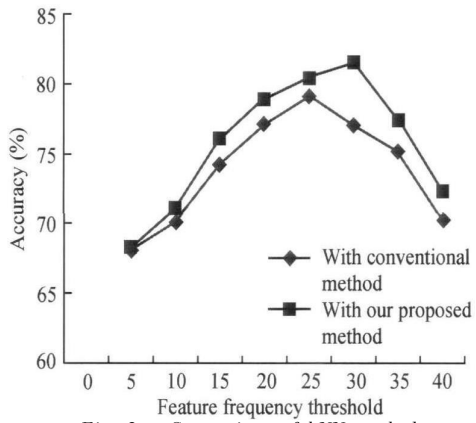

Fig. 2. Comparisons of kNN method.

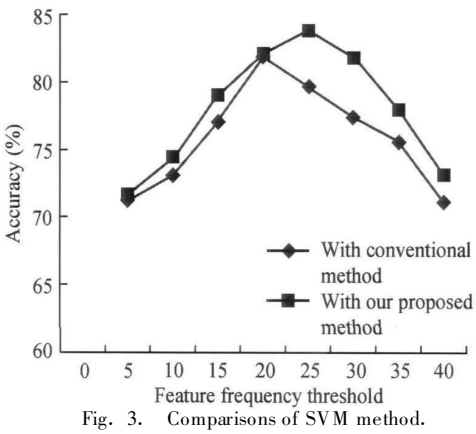

Fig. 3. Comparisons of SVM method.

Table 2 shows the comparison results for the three methods with different feature selection algorithms and computational method for the weight of features. From Table 2, we can see that when the improved feature selection algorithm (Eqs. (5), (6)) is applied, the mean values of MacroP, MacroR and MacroF1 are increased by 0.5%, 0.3% and 0.4%, respectively; when the improved computing method for the weight of features (Eq. (9)) is applied, those values are increased by 0.5%, 1.3%, and 0.9%; while when Eqs. (5), (6) and Eq. (9) are all applied, those values are increased by 6.6%, 6.4% and 6.4%, respectively. Thus, the combination of the two improved methods increases the clustering effectiveness significantly.

Table 2. Numerical results of the original and improved methods

Algorithms	Eq. (1)+Eq. (8)			Eqs. (5), (6)+Eq. (8)		
	MacroP	MacroR	MacroF1	MacroP	MacroR	MacroF1
Rocchio	0.801	0.793	0.797	0.804	0.795	0.799
kNN	0.811	0.820	0.815	0.819	0.824	0.821
SVM	0.841	0.836	0.838	0.845	0.839	0.842
Means	0.818	0.816	0.817	0.823	0.819	0.821
Algorithms	Eq. (1)+Eq. (9)			Eqs. (5), (6)+Eq. (9)		
	MacroP	MacroR	MacroF1	MacroP	MacroR	MacroF1
Rocchio	0.810	0.799	0.804	0.839	0.856	0.847
kNN	0.816	0.840	0.828	0.880	0.885	0.882
SVM	0.843	0.848	0.845	0.932	0.899	0.915
Means	0.823	0.829	0.826	0.884	0.880	0.881

5 Conclusions

Focusing on the feature selection and the evaluation of characteristic weights, we have proposed an improved MI algorithm for feature selection and an improved tf. idf method for the evaluation of characteristic weights. To test the effectiveness of the proposed methods, the benchmark text set, namely Reuters-21578 Top10 is applied for the text clustering. The SVM, kNN and Rocchio methods are used as classifiers for comparisons. Numerical results show that the precision, the rate of recall and the value of F1 of the proposed method are all better than those of the original methods, especially when the two improvements are used together.

References

1 David DL. An evaluation of phrasal and clustered representations on a text categorization task. In: Proceedings of 15th ACM International Conference on Research and Development in Information Retrieval, 1992, 37—50

2 Fuhr N and Buchley C. A probabilistic learning approach for document indexing. ACM Transactions on Information Systems, 1991, 9(3): 223—248

3 Dumais ST, Platt J, Heckerman D, et al. Inductive learning algorithms and representations for text categorization. In: Proceedings of the International Conference on Information and Knowledge Management, 1998, 148—155

4 Joachims T. A probability analysis of the Rocchio algorithm with TFIDF for text categorization. In: Proceedings of the 14th International Conference on Machine Learning (ICML-97), 1997, 43—151

5 Yang Y and Chute CG. An example-based mapping method for text categorization and retrieval. ACM Transaction on Information Systems 1994, 12(3): 252—277

6 Yang Y. Expert network: effective and efficient learning from human decisions in text categorization and retrieval. In: Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval (SIGIR'94), 1994, 13—22

7 Vapnik V. The Nature of Statistical Learning Theory. New York: Springer-Verlag, 1995

8 Mineau GW. A simple KNN algorithm for text categorization. In: International Conference on Data Mining. San Jose, California, USA: IEEE Computer Society, 2001, 647—648

9 Rocchio J. Relevance feedback in information retrieval. In: Salton, the SMART Retrieval System: Experiments in Automatic Document Processing. Englewood Cliffs, New Jersey: Prentice-Hall, 1971, 313—323

10 Widrow B and Stearns SD. Adaptive Signal Processing. Englewood Cliffs, New Jersey: Prentice-Hall, 1979

11 Soucy P and Mineau GW. Beyond TFIDF weighting for text categorization in the vector space mode. In: International Joint Conference on Artificial Intelligence. Edinburgh, Scotland: UK, 2005, 1130—1135

12 Franca D and Fabrizio S. Supervised term weighting for automated text categorization. In: Proceedings of the 2003 ACM Symposium on Applied Computing. Melbourne, Florida, USA: ACM, 2003, 784—788

13 Salton G and Buckley B. Term weighting approaches in automatic text retrieval. Information Processing and Management, 1998, 24(5): 513—523

14 Yang Y and Pedersen JP. A comparative study on feature selection in text categorization. In: Proceedings of the 14th International Conference on Machine Learning, 1997, 412—420

15 Zhu JB and Yao T. FIFA: a simple and effective approach to text topic automatic identification. In: Proceedings of International Conference on Multilingual Information Processing, 2002, 207—215

16 Fabrizio S. Machine learning in automated text categorization. ACM Computing Surveys, 2002, 34(1): 1—47