Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Information Sciences 158 (2004) 89–115

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Effect of term distributions on centroid-based text categorization

Verayuth Lertnattee, Thanaruk Theeramunkong *

Information Technology Program, Sirindhorn International Institute of Technology, Bangkadi Campus, 131 Moo 5 Tiwanont Road, Bangkadi, Muang, Pathumthani 12000, Thailand

Received 27 December 2002; received in revised form 25 June 2003; accepted 15 July 2003

Abstract

Most of traditional text categorization approaches utilize term frequency (tf) and inverse document frequency (idf) for representing importance of words and/or terms in classifying a text document. This paper describes an approach to apply term distributions, in addition to tf and idf, to improve performance of centroid-based text categorization. Three types of term distributions, called inter-class, intra-class and in-collection distributions, are introduced. These distributions are useful to increase classification accuracy by exploiting information of (1) term distribution among classes, (2) term distribution within a class and (3) term distribution in the whole collection of training data. In addition, this paper investigates how these term distributions contribute to weight each term in documents, e.g., a high term distribution of a word promotes or demotes importance or classification power of that word. To this end, several centroid-based classifiers are constructed with different term weightings. Using various data sets, their performances are investigated and compared to a standard centroid-based classifier (TDIDF) and a centroid-based classifier modified with information gain. Moreover, we also compare them to two well-known methods: *k*-NN and naïve Bayes. In addition to a unigram model of document representation, a bigram model is also explored. Finally, the effectiveness of term distributions to improve classification accuracy is explored with regard to the training set size and the number of classes.

* Corresponding author. Tel.: +66-2-501-3505-20x2022/2004; fax: +66-2-501-3524.
E-mail addresses: verayuth@siit.tu.ac.th, verayuths@hotmail.com (V. Lertnattee), thanaruk@siit.tu.ac.th (T. Theeramunkong).

0020-0255/$ - see front matter © 2003 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2003.07.007

Keywords: Centroid-based classifier; Text categorization; Term distribution

1. Introduction

With the fast growth of online text information, there has been extreme need to find and organize relevant information in text documents. For this purpose, it is known that automatic text categorization (also known as text classification) becomes a significant tool to utilize text documents efficiently and effectively. As an application, it can improve text retrieval as it allows find class-based retrieval instead of full retrieval. Given statistics acquired from a training set of labeled documents, text categorization is a method to use these statistics to assign a class label to a new document. In the past, a variety of classification models were developed in different schemes, such as probabilistic models (i.e., Bayesian classification) [1,2], decision trees and rules [3], regression models [4,5], example-based models (e.g., *k*-nearest neighbor or *k*-NN) [5–8], linear models [9–12], support vector machine [5,13,14], neural networks [15] and so on. Among these methods, a variant of linear models called a centroid-based model is attractive since it has relatively less computation than other methods in both the learning and classification stages. The traditional centroid-based method [16], can be viewed as a specialization of so-called Rocchio method [9] and used in several works on text categorization [10,11,17,18]. Based on the vector space model, a centroid-based method computes beforehand, for each class (category), an explicit profile (or class prototype), which is a centroid vector for all positive training documents of that category. The classification task is to find the most similar class to the vector of the document we would like to classify, for example by the means of cosine similarity. Despite the less computation time, centroid-based methods were shown in several literatures including those in [9,12,16–19], to achieve relatively high classification accuracy. In a centroid-based model, an individual class is modeled by weighting terms appearing in training documents assigned to the class. This makes classification performance of the model strongly depend on the weighting method applied in the model. Most previous works of centroid-based classification focused on weighting factors related to frequency patterns of words or documents in the class. Moreover, they are often obtained from statistics within a class (i.e., positive examples of the class). The most popular factors are term frequency (tf) and inverse document frequent (idf). Even some previous works, such as those in [10,11], attempted to apply negative examples in term weighting, they are derived from frequency-based factors. In contrast to previous approaches, this paper proposes a novel approach to consider both positive and negative examples by means of term distribution factors that

enable the consideration of information inside and outside a class in a classification process.

In this paper, term distributions are shown to be useful in improving classification accuracy. Three types of term distributions, called inter-class, intra-class and in-collection distributions, are introduced. These distributions are expected to increase the classification accuracy by exploiting information of (1) term distribution among classes, (2) term distribution within a classes and (3) term distribution in the whole collection of training data. They are used to represent importance or classification power to weight that term in a document. Another objective of this paper is to investigate the pattern of how these term distributions contribute to weight a term in documents. For example, high term distribution of a word (or term) should promote or demote importance of that word. To this end, several centroid-based classifiers are constructed with different term weightings. Using various data sets, their performances are investigated and compared to a standard centroid-based classifier (tf $\times$ idf) and a variant of centroid-based classifier where tf $\times$ idf is modified with a popular feature goodness criterion called *information gain*. We also compare them to two well-known methods: k-NN and naïve Bayes. In addition to a unigram model of document representation, a bigram model is also explored. By the modified models, the effectiveness of term distributions to classification accuracy is also investigated. In the rest of this paper, Section 2 gives a formal description of the classification task. Section 3 presents centroid-based text categorization in three aspects: representation basics, class prototype construction and classification execution. The proposed term distributions for term weighting are given in Section 4. The data sets and experimental settings are described in Section 5. In Section 6, experimental results using four data sets are given. Section 7 provides discussion and some related works. A conclusion is made in Section 8.

2. Definition of text classification task

Text categorization or text classification (TC) is a task of assigning a Boolean value to each pair $\langle d_j, c_k\rangle \in D \times C$, where $D = \{d_1, d_2, \ldots, d_{|D|}\}$ is a domain of documents and $C = \{c_1, c_2, \ldots, c_{|C|}\}$ is a set of predefined *categories*. A value of T (*i.e., true*) is assigned to $\langle d_j, c_k\rangle$ when the document d_j is determined to belong to the category c_k. On the other hand, a value of F (*i.e., false*) is assigned to $\langle d_j, c_k\rangle$ when the document d_j is determined not to belong to the category c_k. In general, text classification is composed of two main phases, called *model training* phase and *classification* phase. In the training phase, the task is to approximate the unknown *target function* $U : D \times C \to \{T, F\}$ that describes how documents should be classified. Based on a training set, a

function $\hat{U}$ called the *classifier* (also called *rule*, *hypothesis*, or *model*) is acquired as the result of approximation. A good classifier is a model that coincides with the target function as much as possible.

The TC task discussed above is general. Anyway, there are some additional factors or constraints possible for this task. They include single-label vs. multi-label, category-pivoted vs. document-pivoted and hard vs. ranking classification. Single-label classification assigns exactly one category to each $d_j \in D$ while multi-label classification may give more than one categories to the same $d_j \in D$. A special case of single-label TC is *binary* TC where each $d_j \in D$ must be assigned either to category c_k or to its complement $\neg c_k$. From the pivot aspect, there are two different ways of using a text classifier. Given $d_j \in D$, the task to find all the $c_k \in C$ that the document d_j belongs to is called document-pivoted classification. Alternatively, given $c_k \in C$, the task to find all the $d_j \in D$ that the document d_j belongs to is named category-pivoted classification. This distinction is more pragmatic than conceptual and it occurs when the sets C and D might not be available in their entirety right from the scratch. Lastly, hard categorization is to assign T or F decision for each pair $\langle d_j, c_k \rangle$ while ranking categorization is to rank the categories in C according to their estimated appropriateness to d_j, without taking any hard decision on any of them. The task of ranking categorization is to approximate the unknown *target function* $U : D \times C \to \{T, F\}$ by generating a classifier $\hat{U} : D \times C \to [0, 1]$ that matches with the target function as much as possible. The result is to assign a number between 0 and 1 to each pair $\langle d_j, c_k \rangle$. This number represents the likelihood the document d_j is classified into the category c_k. Finally, for each $d_j \in D$, a ranked list of categories is obtained. This list would be of great help to a human expert in charge of taking the final categorization decision. By these definitions, the focused task in this work is evaluated as single-label, category-pivoted and hard classification.

3. Centroid-based text categorization

In centroid-based text categorization, an explicit profile of a class (also called a class prototype) is calculated and used as the representative of all positive documents of the class. The classification task is to find the most similar class to the document we would like to classify, by way of comparing the document with the class prototype of the focused class. This approach is characterized by at least three factors; (1) representation basics, (2) class prototype construction: term weighting and normalization, and (3) classification execution: query weighting and similarity definition. Their details are described in the rest of this section.

3.1. Representation basics

The frequently used document representation in IR and TC is the so-called bag of words (BOW) where words in a document are used as basics for representing that document. There are also some works that use additional information such as word position [20] and word sequence [21] in the representation. In the centroid-based text categorization, a document (or a class) is represented by a vector using a vector space model with BOW [22–24]. In this representation, each element (or feature) in the vector is equivalent to a unique word with a weight. The method to give a weight to a word is varied work by work as described in the following section. In a more general framework, the concept of n-gram can be applied. Instead of a single isolated word, a sequence of n words will be used as representation basics. In several applications, not specific for classification, the most popular n-grams are 1-gram (unigram), 2-gram (bigram) and 3-gram (trigram). Alternatively, the combination of different n-grams, for instance the combination of unigram and bigram, can also be applied. The n-grams or their combinations form a set of so-called terms that are used for representing a document. Although a higher-gram provides more information and this may effect in improving classification accuracy, more training data and computational power are required. Therefore, unigram and bigram are considered in this work.

3.2. Class prototype construction: term weighting and normalization

Once we obtain a set of terms in a document, it is necessary to represent them numerically. Towards this, term weighting is applied to set a level of contribution of a term to a document. In the past, most of existing works [2,5,8,12,22] applied term frequency (tf) and inverse document frequency (idf) in the form of tf $\times$ idf for representing a document. In the vector space model, given a set of documents $D = \{d_1, d_2, \ldots, d_{|D|}\}$, a document d_j is represented by a vector $\vec{d}_j = \{w_{1j}, w_{2j}, \ldots, w_{mj}\}$, where w_{ij} is a weight assigned to a term t_i in the document. Here, assume that there are m unique terms in the universe. The tf $\times$ idf representation of the document is defined as follows.

$$\vec{d}_j = \begin{bmatrix} w_{1j} \\ w_{2j} \\ \vdots \\ w_{mj} \end{bmatrix} = \begin{bmatrix} \mathrm{tf}_{1j} \times \mathrm{idf}_1 \\ \mathrm{tf}_{2j} \times \mathrm{idf}_2 \\ \vdots \\ \mathrm{tf}_{mj} \times \mathrm{idf}_m \end{bmatrix} \tag{1}$$

In this definition, tf_{ij} is term frequency of a term t_i in a document d_j and idf_i is defined as $\log(|D|/\mathrm{df}_i)$. Here, $|D|$ is the total number of documents in a collection and df_i is the number of documents, which contain the term t_i. Three alternative types of term frequency [22] are (1) occurrence frequency, (2)

augmented normalized term frequency and (3) binary term frequency. The occurrence frequency, the simplest and intuitive one, corresponds to the number of occurrence of the term in a document. The augmented normalized term frequency is defined by $0.5 + 0.5 \times \mathrm{tf}/\mathrm{tf}_{\max}$ where tf is the occurrence frequency and $\mathrm{tf}_{\max}$ is the maximum term frequency in a document. This compensates for relatively high term frequency in the case of long documents. It works well when there are many technical meaningful terms in documents. The binary term frequency is nothing more than 1 for presence and 0 for absence of the term in the document. Term frequency alone may not be enough to represent the contribution of a term in a document. To achieve a better performance, the well-known inverse document frequency can be applied to eliminate the impact of frequent terms that exist in almost all documents.

Besides term weighting, normalization is another important factor to represent a document or a class. Without normalization, the classification result will strongly depend on the document length. A long document takes two advantages over a short document since it may include higher term frequencies and more unique terms in document representation [24]. The higher term frequency of a long document will increase the average contribution of its terms to the similarity between the document and the query. More unique terms also increase the similarity and chances of retrieval of longer documents in preference over shorter documents. To solve this issue, normally all relevant documents should be treated as equally important for classification or retrieval. Normalization by document length is incorporated into term weighting formula to equalize the length of document vectors. Although there are several normalization techniques including *cosine normalization* [12,17,18] and *byte length normalization* [23], the cosine normalization is the most commonly used. It can solve the problem of overweighting due to both higher term frequency and more unique terms. The cosine normalization is done by dividing all elements in a vector with the length of the vector, that is $\sqrt{\sum_i w_i^2}$ where w_i is the weight of the term t_i before normalization.

Given a class c_k with a set of its assigned documents, there are two possible alternatives to create a class prototype. One is to normalize each document vector in a class before summing up all document vectors to form a class prototype vector [16] (normalization then merging). The other is to sum up all document vectors before normalizing the result vector (merging then normalization). The latter one is also called a prototype vector [12]. Both methods obtain high classification accuracy with small time complexity. In our work, we used the latter method, i.e., merging then normalization. The class prototype can be derived as follows. Let $C_k = \{\vec{d} \mid \vec{d}$ is a document vector belonging to the class $c_k\}$ be a set document vectors assigned to the class c_k. Here, a class prototype $\vec{c}_k$ is obtained by summing up all document vectors in C_k and then normalizing the result by its size as follows.

$$\vec{c}_k = \frac{\sum_{\vec{d}_j \in C_k} \vec{d}_j}{\left\| \sum_{\vec{d}_j \in C_k} \vec{d}_j \right\|} \tag{2}$$

3.3. Classification execution: query weighting and similarity definition

The last but not least important factors are query weighting and similarity definition. For query weighting, term weighting described above can also be applied to a query or a test document (i.e., a document to be classified). The simple term weighting for a query is tf $\times$ idf. In the same way as class prototype construction, there are three possible types of term frequency: occurrence frequency, augmented normalized term frequency and binary term frequency. Once a class prototype vector and a query vector have been constructed, the similarity between these two vectors can be calculated. The most popular one is cosine distance [22,23]. This similarity can be calculated by the dot product between these two vectors. Therefore, the test document (d^t) will be assigned to the class C' whose class prototype vector is the most similar to the query vector ($\vec{d}^t$) of the test document.

$$C' = \arg\max_{C_k \in C} \sin(\vec{d}^t, \vec{c}_k) \tag{3}$$

$$= \arg\max_{C_k \in C} \cos(\vec{d}^t, \vec{c}_k) \tag{4}$$

$$= \arg\max_{C_k \in C} \frac{\vec{d}^t \cdot \vec{c}_k}{\|\vec{d}^t\| \|\vec{c}_k\|} \tag{5}$$

$$= \arg\max_{C_k \in C} \vec{d}^t \cdot \vec{c}_k \tag{6}$$

Here, as stated before, $\|\vec{c}_k\|$ is equal to 1 since the class prototype vector has been normalized. Moreover, the normalization of the test document has no effect on ranking. Therefore, the test document is assigned to the class when the dot product of the test document vector and the class prototype vector achieves its highest value.

4. Term distributions

In the previous section, term weighting is designed on frequency patterns of words or documents in the class. This section illustrates the way to apply term distributions, using two basic questions: which term distributions are useful for exploiting different types of information in the training set and how they contribute to weighting terms. Their details are described separately in different subsections shown below.

4.1. Term distributions

Towards the first question, we need to figure out what are the characteristics of terms that are significant for representing a document or a class. In general, we can observe that a significant term shares some of the following properties.

- It should appear frequently in a certain class.
- It should appear in few documents.
- It should not distribute very differently among documents in the whole collection.
- It should distribute very differently among classes.
- It should not distribute very differently among documents in a class.

The first and second items can be coped by the conventional term frequency and inverse document frequency, respectively. However, for the rest, it is necessary to use distribution (relative information) instead of frequency (absolute information). Distribution related information that we can exploit includes distributions of terms among classes, within a class and in the whole collection. Three kinds of this information can be defined as inter-class standard deviation (icsd), class standard deviation (csd) and standard deviation (sd). Let tf_{ijk} be term frequency of the term t_i of the document d_j in the class c_k. The formal definitions of icsd, csd and sd are given below.

$$\mathrm{icsd}_i = \sqrt{\frac{\sum_k \left[\overline{\mathrm{tf}}_{ik} - \frac{\sum_k \overline{\mathrm{tf}}_{ik}}{|C|}\right]^2}{|C|}} \tag{7}$$

$$\mathrm{csd}_{ik} = \sqrt{\frac{\sum_{d_j \in C_k} [\mathrm{tf}_{ijk} - \overline{\mathrm{tf}}_{ik}]^2}{|C_k|}} \tag{8}$$

$$\mathrm{sd}_i = \sqrt{\frac{\sum_k \sum_{d_j \in C_k} \left[\mathrm{tf}_{ijk} - \frac{\sum_k \sum_{d_j \in C_k} \mathrm{tf}_{ijk}}{\sum_k |C_k|}\right]^2}{\sum_k |C_k|}} \tag{9}$$

where

$$\overline{\mathrm{tf}}_{ik} = \frac{\sum_{d_j \in C_k} \mathrm{tf}_{ijk}}{|C_k|}$$

is an average term frequency of the term t_i in all documents within the class c_k, $|C|$ is the number of classes and $|C_k|$ is the number of documents in the class c_k.

Inter-class standard deviation icsd_i: The inter-class standard deviation icsd_i of a term t_i is calculated from a set of average frequencies $\overline{\mathrm{tf}}_{ik}$, each of which is gathered from each class c_k. This deviation is an inter-class factor. Therefore, icsd for a term is independent of classes. A term with a high icsd distributes differently among classes and should have higher discriminating power for classification than the others. This factor promotes a term that exists in almost all classes but its frequencies for those classes are quite different. In this situation, the conventional factors tf and idf are not helpful.

Class-standard deviation csd_{ik}: The class standard deviation csd_{ik} of a term t_i in a class c_k is calculated from a set of term frequencies tf_{ijk}, each of which comes from term frequency of that term in a document in the class. This deviation is an intra-class factor. Therefore, csds for a term vary class by class. Different terms may appear with quite different frequencies among documents in the class. This difference can be alleviated by the way of this deviation. A term with a high csd will appear in most documents in the class with quite different frequencies and should not be a good representative term of the class. A low csd of a term may be triggered by either of the following two reasons. The occurrences of the term are nearly equal for all documents in the class or the term rarely occurs in the class.

Standard deviation sd_i: The standard deviation of a term t_i is calculated from a set of term frequencies tf_{ijk}, each of which comes from term frequency of that term in a document in the collection. The deviation is a collection factor. Therefore, sd for a term is independent of classes. Different terms may appear with quite different frequencies among documents in the collection. This difference can be also alleviated by the way of this deviation. A term with a high sd will appear in most documents in the collection with quite different frequencies. A low sd of a term may be caused by either of the following two reasons. The occurrences of the term are nearly equal for all documents in the collection or the term rarely occurs in the collection.

4.2. Enhancement of term weighting using term distributions

The second question is how the above-mentioned term distributions contribute to term weighting. The term distributions, icsd, csd and sd, can enhance the performance of a centroid-based classifier with the standard weighting $\mathrm{tf} \times \mathrm{idf}$. They can involve in either of the following two forms: these distributions should act as a promoter (multiplier) or a demoter (divisor) and how strong they affect the weight. To grasp these characteristics, term weighting can be designed using the following skeleton.

$$w_{ik} = \mathrm{tf}_{ik} \times \mathrm{idf}_i \times \mathrm{TDF}_{ik} \tag{10}$$

$$\mathrm{TDF}_{ik} = \mathrm{icsd}_i^{\alpha} \times \mathrm{csd}_{ik}^{\beta} \times \mathrm{sd}_i^{\gamma} \tag{11}$$

Here, w_{ik} is a weight given to the term t_i of the class c_k. The TDF_{ik} involves with term distributions. The parameters α, β and γ are numeric values used for setting the contribution levels of icsd, csd and sd to term weighting, respectively. For each parameter, a positive number means the factor acts as a promoter while a negative one means the factor acts as a demoter. Moreover, the larger a parameter is, the more the parameter contributes to term weighting as either a promoter or a demoter.

5. Data sets and experimental settings

5.1. Data sets and preprocessing

Four data sets are used in the experiments: (1) Drug Information (DI), (2) Newsgroups (News), (3) WebKB1 and (4) WebKB2. The first data set, DI is a set of web pages collected from www.rxlist.com. It includes 4480 English web pages with seven classes: *adverse drug reaction*, *clinical pharmacology*, *description*, *indications*, *overdose*, *patient information*, and *warning*. Each web page in this data set consists of informative content with a few links. Its structure is well organized. The second data set, Newsgroups contains 19,997 documents. It was used in many literatures, such as [25–27]. The articles are grouped into 20 different UseNet discussion groups. In this data set, some groups are very similar. The third and fourth data sets are constructed from WebKB containing 8145 web pages. Recently, it has frequently been used in several works, including those in [28,29]. These web pages were collected from departments of computer science from four universities with some additional pages from some other universities. The collection can be arranged to seven classes. In our experiment, we use the four most popular classes: *student*, *faculty*, *course* and *project* as our third data set called WebKB1. The total number of web pages is 4199. Alternatively, this reduced collection can be rearranged into five classes by university (WebKB2): *cornell*, *texas*, *washington*, *wisconsin* and *misc* (collected from some other universities). The pages in WebKB are varied in their styles, ranging from quite informative pages to link pages. Table 1 indicates the major characteristics of the data sets. More

Table 1
Characteristics of the four data sets

Data sets	DI	News	WebKB1	WebKB2
1. Type of docs	HTML	Plain Text + Header	HTML	HTML
2. No. of docs	4480	19,997	4199	4199
3. No. of classes	7	20	4	5
4. No. of docs/class	640	1000	Varied	Varied

Table 2
The distribution of the documents in WebKB1 and WebKB2

WebKB1	WebKB2					Subtotal
	Cornell	Texas	Wash	Wisc	Misc	
Course	44	38	77	85	686	930
Faculty	34	46	31	42	971	1124
Project	20	20	21	25	418	504
Student	128	148	126	156	1083	1641
Subtotal	226	252	255	308	3158	4199

detail about the document distribution of each class in WebKB is shown in Table 2.

For the HTML-based data sets (i.e., DI and WebKB), all HTML tags are eliminated from the documents in order to make the classification process depend not on tag sets but on the content of web documents. By the similar reason, all headers are omitted from Newsgroups documents, the e-mail-based data set. For all data sets, a stop word list is applied to take away some common words, such as *a*, *for*, *the* and so on, from the documents. This means when a unigram model is occupied, a vector is constructed from all features (words) except stop words. In the case of a bigram model, after eliminating stop words, any two contiguous words are combined into a term for the representation basic. Moreover, we ignore terms occurring less than three times in this case.

5.2. Experimental settings

The following five experiments are performed. The first experiment aims to investigate the effect of a single term distribution on accuracy improvement of a standard centroid-based classifier. Only one term distribution factor is added, in turn, to the standand $\mathrm{tf} \times \mathrm{idf}$ as either a multiplier (for promoting) or a divisor (for demoting). The representation basic applied is a unigram model. In the second experiment, multiple term distribution factors are combined in different manners, and the efficiencies of these combinations are evaluated. At this step, we call the classifiers that incorporate term distribution factors in their weighting, term-distribution-based centroid-based classifiers (later called TCBs). As the third experiment, top 10 TCBs obtained from the second experiment are selected for investigating the effect for term distribution factors in different types of frequency-based factor in query weighting in both unigram and bigram models. Three types of query weighting are investigated: term frequency, binary and augmented normalized term frequency. In this experiment and the latter, the TCBs will be compared to a number of well-known methods as a baseline for comparison: a standard centroid-based classifier (for

short, SCB), a centroid-based classifier modified the term weighting with information gain (for short, SCB*IG), k-NN and naïve Bayes (for short, NB). In the fourth experiment, we study the effect of the size of training sets on classification accuracy with term distribution weighting. In the last experiment, the WebKB data set is arranged into 20 groups (WebKB12) based on the combination of topics and universities. This experiment investigates the performance of TCBs when more classes are provided while retaining the same number of documents. The number of documents per class is naturally smaller than those of WebKB1 and WebKB2. In all experiments except the fourth one, a data set is split into two parts: 90% for the training set and 10% for the test set. In the fourth experiment, since the objective is to investigate the effect of training set size, we fix the size of a test set to 10% of the whole data set but vary the size of a training set from 10% to 90%. All experiments perform 10-fold cross validation.

One of the most important factors towards the meaningful evaluation is the way to set classifier parameters. Parameters that are applied to these classifiers are determined by some preliminary experiments. For SCB, we apply the standard term weighting, tf $\times$ idf. For SCB*IG, a term goodness criterion called information gain (IG) (see Appendix A) is applied for adjusting the weight in SCB, resulting in tf $\times$ idf $\times$ IG. The k values in k-NN are set to 20 for DI, 30 for Newsgroups and 50 for both WebKB1, WebKB2 and WebKB12. Moreover, term weighting used in k-NN is $(0.5 + 0.5 \times \mathrm{tf}/\mathrm{tf}_{\max}) \times \mathrm{idf}$ where $\mathrm{tf}_{\max}$ is the maximum term frequency in a document. The k and this term weighting performed well in our pretests. For NB, two possible alternative methods to calculate the posterior probability $P(t_i|c_k)$ are binary frequency and occurrence frequency. The occurrence frequency is selected for comparison since it outperforms the binary frequency. Except the third and fifth experiments, the query weighting for TCBs is tf $\times$ idf by default. As the performance indicator, classification accuracy is applied. It is defined as the ratio of the number of documents assigned with their correct classes to the total number of documents in the test set.

6. Experimental results

6.1. Effects of single additional term distribution factors

In the first experiment, three term distribution factors, i.e., icsd, csd and sd are individually evaluated by adding each term factor one by one to the standard tf $\times$ idf. For clarity, individual tf and idf are also evaluated. The representation basic applied is a unigram model. The query weighting is tf $\times$ idf. The result is shown in Table 3. The bold indicates term weightings, which achieve a higher performance than tf $\times$ idf.

Table 3
The effect of single additional term distribution factors to tf × idf

Methods	DI	News	WebKB1	WebKB2	Avg.
tf	70.33	71.88	62.25	65.80	67.57
idf	44.26	74.13	39.41	22.46	45.07
tf × idf (SCB)	91.67	74.76	77.71	88.76	83.23
tf × idf × csd	69.26	58.48	55.63	42.58	56.49
tf × idf/csd	90.87	**82.53**	76.87	60.75	77.76
tf × idf × icsd	84.49	56.70	54.42	**88.86**	71.11
tf × idf/icsd	72.54	**80.99**	50.32	24.20	57.01
tf × idf × sd	77.28	57.69	48.75	49.11	58.21
tf × idf/sd	88.55	**82.70**	73.45	**90.71**	**83.85**

The result shows that the standard tf × idf (SCB) performs better than both tf and idf. Although a single term distribution factor may improve classification accuracy for some data sets and the tf × idf/sd performs slightly better than SCB on average, unfortunately there is no single term distribution factor that significantly improves average accuracy over the standard tf × idf for all data sets. An interesting phenomenon we can observe in this experiment is that term distribution factors have some effects on classification accuracy in either roles of promoting or demoting the weight. It is quite clear that sd and csd should act as a demoter (divisor) than a promoter (multiplier) while icsd may work as a promoter.

6.2. Effect of multiple term distribution factors

This experiment investigates the combination of term distribution factors in improving the classification accuracy. Although the previous experiment suggests the role of each term distribution factor, all possible combinations are explored in this experiment. Two following issues are taken into account: (1) which factors are suitable to work together and (2) what is the appropriate combination of these factors. To the end, we perform all combinations of icsd, csd and sd by varying the power of each factor between −1 and 1 with a step of 0.5 and using it to modify the standard tf × idf. At this point, a positive number means the factor acts as a promoter while a negative one means the factor acts as a demoter. The total number of combinations is 125 ($=5\times5\times5$). These combinations include tf × idf and six single-factor term weightings. By the result, we find out that there are only 19 patterns giving better performance than tf × idf. The 20 best (top 20) and the 20 worst classifiers, according to average accuracy on the four data sets, are selected for evaluation. Table 4—panel A (panel B) shows the number of the best (worst) classifiers for each power of icsd, csd and sd. Moreover, the numbers in parentheses show the numbers of the top 10 classifiers for each power. For more detail, the

Table 4
Descriptive analysis of term distribution factors with different power of each factor. Panel A: the best 20 and panel B: the worst 20 (best 10 and worst 10 in parenthesis)

Term distribution factors	Power of the factor					Total methods
	−1	−0.5	0	0.5	1	
Panel A						
icsd	0(0)	0(0)	6(2)	9(5)	5(3)	20(10)
csd	5(4)	7(4)	6(2)	2(0)	0(0)	20(10)
sd	9(4)	7(4)	4(2)	0(0)	0(0)	20(10)
Panel B						
icsd	6(1)	3(1)	3(2)	3(3)	5(3)	20(10)
csd	4(0)	0(0)	1(0)	6(2)	9(8)	20(10)
sd	1(0)	1(0)	1(0)	6(3)	11(7)	20(10)

characteristics and performances of the top 20 term weightings are shown in Table 5.

Table 4(panel A) provides the same conclusion as the result obtained from the first experiment. That is, sd and csd are suitable to be a demoter rather than a promoter while icsd performs opposite. There are almost no negative results, except csd, and it is more obvious in the case of the top 10. On the other hand, Table 4(panel B) shows that the performance is low if sd and csd are applied as a promoter. However, it is not clear whether using icsd as a demoter harms the performance. Table 5 also emphasizes the classifiers that outperform the standard tf $\times$ idf in all four data sets, with a mark '$*$'. Here, there are nine classifiers that are raised up. This fact shows that there are some common term distributions that are useful generally in all data sets.

The best term distribution in this experiment is $\mathrm{sqrt}(\mathrm{icsd}/(\mathrm{csd} \times \mathrm{sd}))$. That is, the powers are 0.5 for icsd, and −0.5 for both csd and sd. However, it is observed that the appropriate powers of term distribution factors depend on some characteristics of data sets. For instance, when the power of csd changes from −0.5 to −1.0 (TCB1 to TCB3 in Table 5), the performances for DI and WebKB1 decrease but those for Newsgroups and WebKB2 increase. This suggests that csd, a class dependency factor, is more important in Newsgroups and WebKB2 than DI and WebKB1.

From another viewpoint, our proposed term distribution factors adjust term weighting with two approaches: class dependence and class independence. As class dependence approach, csd plays as a demoter and emphasizes terms that occur with nearly equal in frequency within an individual class, as more important. As class independence approach, sd acts similarly to csd by promoting terms that occurs almost equally in every document in a collection (not in a certain class) or appears rarely in only few documents. Another class independence factor is icsd. It performs as a promoter by stressing the terms that

Table 5
Classification accuracy of the 20 best term weightings

Methods	Power of			Sum of power	Term weightings	DI	News	WebKB1	WebKB2	Avg.
	icsd	csd	sd							
TCB1*	0.5	−0.5	−0.5	−0.5	tf× idf× sqrt(icsd/(csd × sd))	96.81	79.52	82.45	92.67	87.86
TCB2*	0.5	−1	0	−0.5	tf × idf× sqrt(icsd) /csd	95.16	79.73	81.90	93.17	87.49
TCB3*	0.5	−1	−0.5	−1	tf × idf× sqrt(icsd)/ (csd× sqrt(sd))	92.25	83.17	78.88	93.71	87.00
TCB4*	1	−0.5	−1	−0.5	tf × idf × icsd/ (sqrt(csd) × sd)	96.65	77.70	82.90	90.21	86.87
TCB5*	0.5	0	−1	−0.5	tf × idf× sqrt(icsd)/sd	96.14	77.67	81.50	91.24	86.63
TCB6*	0.5	−0.5	−1	−1	tf × idf × sqrt(icsd)/ (sqrt(csd) × sd)	92.57	83.13	78.64	91.62	86.49
TCB7	1	−1	−1	−1	tf × idf × icsd /(csd × sd)	91.07	82.17	80.09	92.28	86.40
TCB8*	1	−1	−0.5	−0.5	tf × idf × icsd/ (csd × sqrt(sd))	94.80	78.79	80.16	91.14	86.22
TCB9*	0	−0.5	0	−0.5	tf × idf/sqrt(csd)	93.75	80.70	80.90	89.19	86.13
TCB10*	0	0	−0.5	−0.5	tf × idf/sqrt(sd)	92.90	78.97	79.11	92.86	85.96
TCB11	0.5	−0.5	0	0	tf × idf× sqrt(icsd/csd)	96.45	74.40	80.28	92.02	85.79
TCB12	0	−0.5	−0.5	−1	tf × idf /sqrt(csd × sd)	90.56	83.08	76.28	92.40	85.58
TCB13	1	−0.5	−0.5	0	tf × idf × icsd/ sqrt(csd × sd)	96.18	71.49	80.81	89.76	84.56
TCB14	0.5	0	−0.5	0	tf × idf× sqrt(icsd/sd)	95.58	72.69	78.64	90.93	84.46
TCB15	0	0.5	−1	−0.5	tf × idf× sqrt(csd)/sd	90.92	78.21	77.02	91.40	84.39
TCB16	0	0	−1	−1	tf × idf/sd	88.55	82.70	73.45	90.71	83.85
TCB17	1	0	−1	0	tf × idf × icsd/sd	96.00	69.76	79.95	89.50	83.80
TCB18	0.5	0.5	−1	0	tf × idf× sqrt(icsd × csd)/sd	93.95	71.73	78.45	90.24	83.59
TCB19	0.5	−1	−1	−1.5	tf × idf × sqrt(icsd)/ (csd × sd)	90.67	82.09	70.64	90.64	83.51
SCB	0	0	0	0	tf × idf	91.67	74.76	77.71	88.76	83.23

appear very differently in terms of occurrence frequencies among several classes. In order to grasp what important terms look like, we focus on only class independence factors (i.e., icsd and sd), and list 30 terms with the highest icsd/sd values for each data set. We also compare them to terms listed in order of information gain as shown in Appendix B. We can observe that several terms overlap between these two criteria. From this observation and the detail given in Section 4, TCB1 is achieved the best performance due to icsd/sd which sets the higher weight of the important terms. In addition, csd selects the terms which frequently appear with nearly equal occurrences in term frequency as more important terms.

6.3. Experiments with different query weightings, and unigram/bigram models

In this experiment, top 10 TCBs obtained from the previous experiment are selected for exploring the effect for term distribution factors in different types of query weighting in both unigram and bigram models. In this experiment, the TCBs are compared to SCB, SCB*IG, *k*-NN and NB. Three types of query weighting are investigated: term frequency (*n*), binary (*b*) and augmented normalized term frequency (*a*). The simple query weighting (*n*) sets term frequency (or occurrence frequency) tf as the weight for a term in a query. The binary query weighting (*b*) sets either 0 or 1 for terms in a query. The augmented normalized term frequency (*a*) defines $0.5 + 0.5 \times \mathrm{tf}/\mathrm{tf}_{\max}$ as a weight for a term in a query. This query term weighting is applied for all centroid-based classifiers, i.e., TCBs, SCB and SCB*IG. Furthermore, the query term weighting is modified by multiplying the original weight with inverse document frequency (idf). The results for unigram and bigram models are shown in Table 6(panels A and B), respectively.

According to the results, we found out that the TCBs outperformed SCB, SCB*IG, *k*-NN and NB for almost cases, in both unigram and bigram models, independently of query weighting. Normally the bigram model gains better performance than the unigram model. In the bigram, the term distributions are still useful to improve classification accuracy. However, it is hard to determine which query weighting performs better than the others but term distributions are helpful for all types of query weighting. For SCB*IG, the accuracy on DI significantly improves. However, a little bit lower performance than SCB on average. The TCB1, TCB2 and TCB3 seem to achieve higher accuracy than the others even TCB4 and TCB6 perform better in the bigram model for DI and News, respectively.

6.4. Effect of training set size on classification accuracy

In this experiment, we evaluate the effect of training set size on the performance of TCBs. First of all, the whole data set is spitted into two parts: 90%

Table 6
Accuracy of the top 10 TCBs with different types of query weight compared to SCB, SCB*IG, *k*-NN and NB for (panel A) unigram and (panel B) bigram models

Methods	DI			News			WebKB1			WebKB2		
	n	*b*	*a*	*n*	*b*	*a*	*n*	*b*	*a*	*n*	*b*	*a*
Panel A (Unigram)												
TCB1	96.81	**97.86**	97.81	79.52	79.66	79.78	82.45	84.66	84.59	92.67	90.83	91.43
TCB2	95.16	95.96	95.87	79.73	80.93	80.95	81.90	**85.33**	85.12	93.17	93.05	93.21
TCB3	92.25	92.90	92.90	83.17	83.44	**83.64**	78.88	82.62	82.14	**93.71**	92.47	92.95
TCB4	96.65	97.46	97.39	77.70	77.72	77.88	82.90	85.12	85.02	90.21	88.02	88.83
TCB5	96.14	97.25	97.21	77.67	78.16	78.14	81.50	83.54	83.26	91.24	89.16	89.76
TCB6	92.57	93.06	93.01	83.13	83.30	83.46	78.64	81.07	80.61	91.62	89.19	89.76
TCB7	91.07	92.10	92.08	82.17	82.95	82.95	80.09	83.83	83.54	92.28	90.52	90.93
TCB8	94.80	96.36	96.32	78.79	79.53	79.62	80.16	84.12	84.02	91.14	89.69	90.12
TCB9	93.75	94.62	94.55	80.70	80.83	80.91	80.90	83.19	82.95	89.19	91.21	91.07
TCB10	92.90	94.51	94.38	78.97	79.38	79.57	79.11	81.47	81.21	92.86	91.71	92.19
SCB	91.67	92.99	93.01	74.76	75.29	75.37	77.71	78.66	78.73	88.76	91.12	91.07
SCB*IG	96.19	97.43	97.39	60.83	59.31	60.40	75.02	78.78	78.26	90.26	89.59	89.95
k-NN		94.60			82.69			68.33			89.16	
NB		95.00			80.82			81.40			87.45	
Panel B (Bigram)												
TCB1	98.73	99.35	99.35	81.83	82.37	82.36	84.19	86.71	86.35	93.88	93.36	93.43
TCB2	90.33	94.75	94.53	82.27	83.00	83.04	83.66	**87.88**	87.47	94.67	94.71	94.81
TCB3	97.90	99.24	99.22	85.15	85.20	85.24	82.47	85.69	85.26	**95.52**	94.98	95.19
TCB4	98.64	**99.38**	99.33	80.32	80.94	80.88	84.57	87.43	87.09	92.02	91.19	91.45
TCB5	98.04	98.68	98.68	81.22	81.94	81.91	83.40	85.76	85.43	92.74	92.17	92.36
TCB6	98.95	99.31	99.31	85.58	85.66	**85.71**	82.78	85.45	85.12	94.14	93.36	93.52
TCB7	85.25	90.13	89.71	84.80	84.98	84.92	82.88	86.78	86.31	94.05	93.47	93.57
TCB8	80.36	85.98	85.60	81.43	82.31	82.25	81.88	86.88	86.19	92.76	92.33	92.45
TCB9	98.42	98.93	98.91	82.77	83.05	83.01	82.47	85.16	84.76	93.81	94.88	94.88
TCB10	97.54	98.17	98.15	82.37	82.71	82.75	81.09	84.14	83.71	94.43	94.17	94.26
SCB	96.07	97.41	97.37	77.40	78.44	78.37	79.14	81.71	81.31	92.31	93.62	93.74
SCB*IG	97.83	99.00	98.88	62.50	61.82	62.50	76.07	80.50	80.23	92.24	92.97	93.02
k-NN		97.48			82.75			70.16			91.62	
NB		96.76			82.83			82.21			94.02	

for the training set and 10% for the test set. We fix the size of a test set to 10% of the whole data set but vary the size of a training set from 10% to 90%. For comparison to the other methods, the best classifier TCB1 is chosen as the representative of TCBs. The result is shown in Fig. 1.

The result shows that the TCB1 achieves higher classification accuracy than SCB, SCB*IG, *k*-NN and NB on average even *k*-NN and NB perform a little bit better than TCB1 for the News data set. When the full training set is used (i.e., 0.9), TCB1, SCB, SCB*IG, *k*-NN and NB gains up to 87.86%, 83.23%, 80.57%, 83.69% and 86.17% on average, respectively. That is, the performance gaps between TCB1 and the other approaches; SCB, SCB*IG, *k*-NN and NB

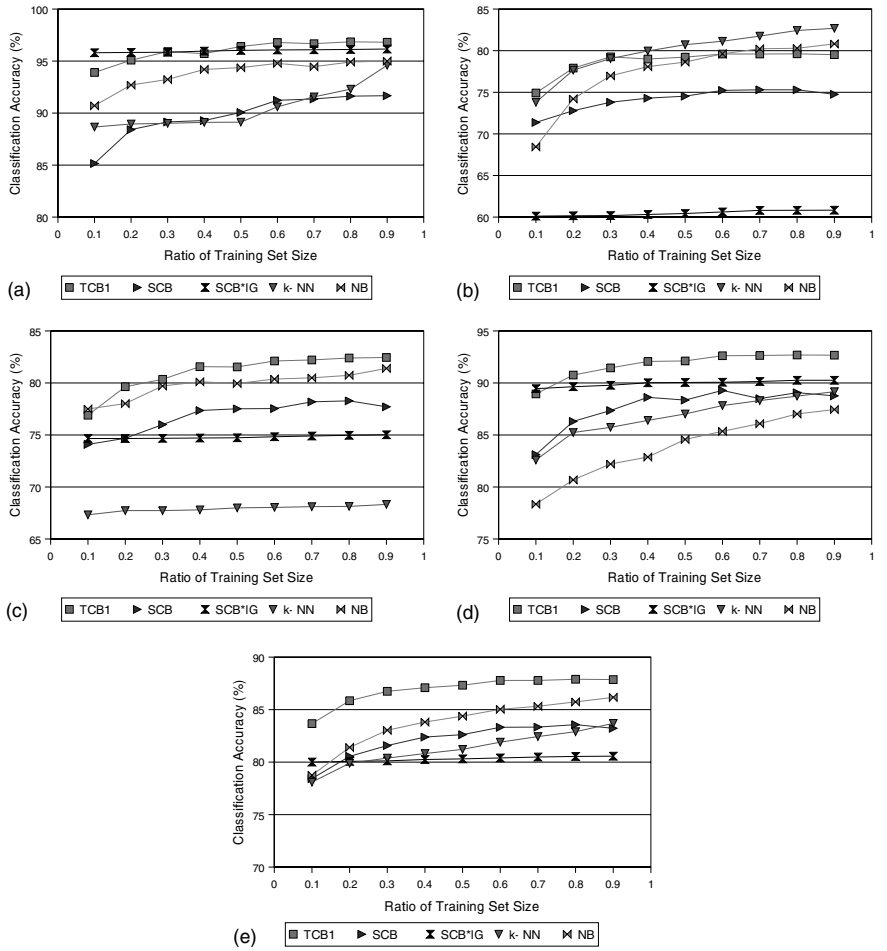

Fig. 1. Effect of training set size on classification accuracy: (a) DI, (b) News, (c) WebKB1, (d) WebKB2 and (e) average on the four data sets.

are 4.64%, 7.30%, 4.17% and 1.70%, respectively. On the other hand, when the training set is reduced down to 0.1 of the whole data set, they perform on average 83.67%, 78.44%, 80.01%, 78.09% and 78.75%, respectively. That is, TCB1 outperforms SCB, SCB*IG, *k*-NN and NB with gaps of 5.23%, 3.66%, 5.57% and 4.91%, respectively. The result from SCB*IG performs better than other classifiers excepts TCB1 in a lower portion of training set size. This result indicates that term distributions are useful for improving classification accuracy in both large and small training sets.

6.5. Efficiency of TCBs with a larger number of classes

In this experiment, the WebKB data set is rearranged to form 20 classes based on the combination of four topics (WebKB1) and five universities (WebKB2). Intuitively, the classification task based on these fine-grained classes is much more difficult than those for WebKB1 and WebKB2 shown in the previous experiments due to the following two reasons. First, more classes may confuse a classifier since some classes will look similar to the others. Secondly, fewer training documents per class may cause the constructed class prototypes not good enough to be a representative of all documents in the class. The results are shown in Table 7.

From the result, we observe that the TCBs still outperform SCB, SCB*IG, *k*-NN and NB even in the case of more classes. For unigram model, referring the results of WebKB1 and WebKB2 in Table 6(panel A), we can observe that the accuracy of *k*-NN and NB drops approximately 10.44% and 18.74%, respectively, comparing with the harder case, i.e., WebKB1 while the accuracy of SCB and TCBs decreases less than 10%. In the bigram model (Table 6(panel B)), the accuracy of *k*-NN and NB drops 8.19% and 11.00% but all types of centroid-based classifiers decreases approximately 7%. This fact indicates that centroid-based classifiers gain more advantage than the other methods when more classes exist. Moreover, TCBs still outperforms SCB and SCB*IG when more classes and fewer documents per class are provided in classification.

Table 7
Classification accuracy of TCBs, SCB, *k*-NN and NB on WebKB with 20 classes

Methods	Unigram			Bigram		
	n	*b*	*a*	*n*	*b*	*a*
TCB1	74.76	75.07	75.47	77.19	79.45	79.04
TCB2	71.33	**76.00**	75.78	76.52	**81.09**	80.57
TCB3	71.61	74.07	73.95	76.66	79.62	79.16
TCB4	73.69	73.85	73.95	76.35	78.54	78.35
TCB5	72.26	72.99	73.28	75.30	77.85	77.52
TCB6	70.52	71.38	71.42	76.09	78.11	78.09
TCB7	71.57	74.11	73.76	75.61	78.59	78.38
TCB8	71.73	75.73	75.38	75.40	80.54	80.02
TCB9	69.90	73.26	73.16	74.73	77.85	77.35
TCB10	68.78	71.37	71.49	73.18	76.33	75.42
SCB	67.28	69.47	69.80	71.68	74.64	74.54
SCB*IG	66.47	69.35	69.61	68.66	73.02	72.92
k-NN		57.89			61.97	
NB		62.66			71.21	

6.6. Summary of experimental results

After investigating our approach in several environments using the four data sets, we found that the proposed term distribution factors: icsd (inter-class factor), csd (intra-class factor) and sd (collection factor) can improve classification accuracy over the standard term weighing tf × idf. Although adding only one term distribution factor (with power = 1 and −1) to the standard term weighting tf × idf does not significantly improves the performance of a standard centroid-based classifier, some appropriate combinations of these term distributions enable the constructed classifiers to clearly outperform the standard one. The first two experiments ensure that icsd should be a promoter while sd and csd should be demoters. With different contribution levels (−1 to 1 with step 0.5), the only 19 out of 125 combinations outperform the standard tf × idf. Among these combinations, nine patterns (i.e., TCB1 to TCB10 with the exception of TCB7) performed better than tf × idf for all data sets. In the third experiment, the top 10 classifiers (including all of these superior classifiers) are compared to standard centroid-based classifiers with tf × idf and a modified term weighting tf × idf × IG. Moreover, two popular methods: *k*-NN and naïve Bayes are used. These 10 TCBs gain higher accuracy in both unigram and bigram models and in all types of query weighting. While the bigram model gives higher accuracy than the unigram, it is hard to specify which query weighting performs better than the others. However, the results indicate that the term distributions are useful for any kind of models and query weighting. The last two experiments show that TCBs have robustness and scalability by outperforming others classifiers even when the training sets are reduced and there are more classes, each of which has fewer documents.

7. Discussion and related works

Term weighting plays an important role to achieve high performance in text classification. In the past, most approaches [7,10,12,22,24,30] were proposed using frequency-based factors, such as term frequency and inverse document frequency, for setting weights for terms. In these approaches, the way to solve the problem caused by the situation that a long document may suppress a short document is to perform normalization on document vectors or class prototype vectors. That is, a vector for representing any document or any class is transformed into a unit vector the length of which equals to 1. In spite of this, it is doubtful whether such frequency-based term weighting is enough to reflect the importance of terms in the representation of a document or a class or not. There were some works on adjusting weights using relevance feedback approach. Among them, two popular schemas are the vector space model and the probalistic networks. For the vector space model, the Rocchio feedback model

[9,31,32] is the most common used method. The method attempts to use both positive and negative instances in term weighting. One can expect more effective profile representation generated from relevance feedback. For probabilistic networks approach, a query can be modified by the addition of the first m terms taken from a list where all terms present in documents deemed relevant are ranked [33].

The probalistic indexing technique was suggested by Fuhr [34] and Joachims [11] has analysed a probabilistic consideration of this technique to the Rocchio classifier with tf $\times$ idf term weighting. In [13], Deng et al. introduced an approach to use statistics in a class , call ''category relevance factor'' to improve classification accuracy. Recently, Debole and Sebastiani [35] have evaluated some feature selection methods such as chi-square, information gain and gain ratio. These feature selection methods were applied into term weighting for substituting idf on three classifiers: k-NN, NB and Rocchio. From the result, these methods might be useful for k-NN and support vector machine but seem useless for Rocchio. However, there is still no report to use term distribution in a systematic way as shown in this paper.

8. Conclusion

This paper showed that term distributions were useful for improving accuracy in centroid-based classification. Three types of term distributions: interclass standard deviation (icsd), class standard deviation (csd) and standard deviation (sd), were introduced to exploit information outside/inside a class and that of the collection. The distributions were used to represent discriminating power of each term and then to weight that term. To investigate the pattern of how these term distributions contribute to weighting each term in documents, we varied term distributions in their contribution to term weighting and then constructed a number of centroid-based classifiers with different term weightings. The effectiveness of term distributions was explored using various data sets. As baselines, a standard centroid-based classifier with tf $\times$ idf, a centroid-based classifiers with tf $\times$ idf $\times$ IG and two well-known methods, k-NN and naïve Bayes are employed. Furthermore, both unigram and bigram models were investigated. The experimental results showed the benefits of term distributions in classification. It was shown that there was a certain pattern that term distributions contribute to the term weighting. It can be claimed that terms with a low sd and a low csd should be emphasized while terms with a high icsd should get more importance. Finally, this paper also gave an analysis of the effect of the training set size and the number of classes on accuracy improvement by term distributions. Independently of the training set size and the number of classes, term distributions were shown to be useful.

Although encouraging results have been obtained using term distributions on centroid-based classifiers, there is still much work remaining to be investigated. They include how to automatically determine contribution levels of term distributions on term weighting and how much term distribution has an effect on any other classification methods. We believe that contribution levels are varied for different data sets but there should be a way to grasp their patterns from the training data set. It is also likely that term distributions may be useful for improving other classification schemes. These two issues are left for our future works.

Acknowledgements

This work has been supported by National Electronics and Computer Technology Center (NECTEC) under project number NT-B-06-4C-13-508. We thank to Dr. McCallum for supporting rainbow-20000915 software to perform naïve Bayes classifier for comparison.

Appendix A. Information gain

Information gain (IG) is frequently used as a term goodness function in text categorization. It is a function which measure the amount of information obtained for category prediction by knowing the presence or absence of a term in a document. The general equation of IG, which is modified to more general than the one used in binary classification is defined as below

$$\mathrm{IG}(t_i) = -\sum_{i=1}^{|C|} P(c_k)\log P(c_k) + \sum_{\{t\in t_i,\bar{t}_i\}} P(t)\sum_{k=1}^{|C|} P(c_k|t)\log_2 P(c_k|t) \qquad (12)$$

Here, $|C|$ is the number of classes in training data. The $P(c_k)$ is the probability of the class c_k, $P(t)$ is the probability of the term t and $P(c_k|t)$ is the conditional probability of the class c_k given term t. This equation is adapted from the simple one that used in binary classification models for m-ary category models. Normally, information gain used as a term selection. In term weighting, all terms are scaled by IG which promotes the important terms in a training corpus.

Appendix B. Thirty terms with highest icsd/sd and information gain

We list 30 terms with the highest icsd/sd values for each data set. We also compare them to the 30 terms with the highest information gain.

Terms with the highest information gain (IG) and icsd/sd are shown as follows:

DI		News		WebKB1		WebKB2	
IG	icsd/sd	IG	icsd/sd	IG	icsd/sd	IG	icsd/sd
supplied	description	windows	bike	professor	instructor	wisc	cornell
description	nursing	god	sale	assignments	syllabus	utexas	austin
indications	supplied	dod	game	instructor	hours	cornell	utexas
overdosage	carcinogenesis	government	car	syllabus	class	washington	wisc
pharmacology	indications	writes	clipper	research	group	austin	dayton
contraindica-tions	fertility	team	cars	class	assignments	madison	texas
adverse	overdosage	game	baseball	hours	fax	wisconsin	washington
reactions	mutagenesis	people	writes	student	research	dayton	ithaca
interactions	mothers	car	israel	homework	rainbowtime	wi	tx
carcinogenesis	milk	bike	israeli	interests	final	texas	wa
mothers	store	article	playoffs	rainbowtime	grading	sciences	seattle
clinical	molecular	hockey	key	fax	interests	seattle	wi
store	potential	encryption	sahak	associate	associate	tx	madison
mutagenesis	caution	jesus	melkonian	university	faculty	wa	wisconsin
fertility	reactions	clipper	arab	exam	rainbow-phonenum	uw	ut
nursing	excreted	israel	nsa	final	office	ithaca	upson
pregnancy	adverse	christian	keys	ph	introduction	cse	sieg
pregnant	contraindicated	bible	appressian	rainbow-phonenum	handouts	upson	sciences
information	rats	sale	ohanus	publications	exam	sieg	postal
dosage	human	christians	clh	faculty	homework	ut	cse
excreted	deg	baseball	team	lecture	due	tay	taylor
studies	fetus	christ	ride	grading	topics	taylor	utcs

DI		News		WebKB1		WebKB2	
IG	icsd/sd	IG	icsd/sd	IG	icsd/sd	IG	icsd/sd
milk	category	fbi	article	rainbow-threedigit	fall	rainbow-sixdigit	uw
category	reported	season	arabs	due	exams	csrainbow-threedigit	autumn
precautions	carcinogenic	dos	chip	notes	publications	cs	csrainbow-threedigit
molecular	teratogenic	games	extermination	midterm	laboratory	postal	chateau
warnings	crystalline	space	riding	pm	prerequisites	science	quarter
reported	stearate	graphics	encryption	handouts	assignment	street	practicum
patient	ingredients	religion	occupied	assignment	midterm	ny	cserainbow threedigit
human	interactions	chip	escrow	department	pm	west	crainbow-fourdigit

References

[1] K. Nigam, A.K. McCallum, S. Thrun, T.M. Mitchell, Text classification from labeled and unlabeled documents using EM, Machine Learning 39 (2/3) (2000) 103–134. Available from <http://www.cs.cmu.edu/knigam/papers/emcat-mlj99.ps>.

[2] Y. Yang, An evaluation of statistical approaches to text categorization, Information Retrieval 1 (1/2) (1999) 69–90. Available from <http://www.cs.cmu.edu/yiming/papers.yy/irj99.ps>.

[3] C.d. Apté, F.J. Damerau, S.M. Weiss, Automated learning of decision rules for text categorization, ACM Transactions on Information Systems 12 (3) (1994) 233–251. Available from <http://www.acm.org/pubs/articles/journals/tois/1994-12-3/p233-apte/p233-apte.pdf>.

[4] Y. Yang, C.G. Chute, An example-based mapping method for text categorization and retrieval, ACM Transactions on Information Systems 12 (3) (1994) 252–277. Available from <http://www.acm.org/pubs/articles/journals/tois/1994-12-3/p252-yang/p252-yang.pdf>.

[5] Y. Yang, X. Liu, A re-examination of text categorization methods, in: M.A. Hearst, F. Gey, R. Tong (Eds.), Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, ACM Press, New York, US, Berkeley, US, 1999, pp. 42–49. Available from <http://www.cs.cmu.edu/yiming/papers.yy/sigir99.ps>.

[6] L.S. Larkey, Automatic essay grading using text categorization techniques, in: W.B. Croft, A. Moffat, C.J. van Rijsbergen, R. Wilkinson, J. Zobel (Eds.), Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval, ACM Press, New York, US, Melbourne, Australia, 1998, pp. 90–95. Available from <http://cobar.cs.umass.edu/pubfiles/ir-121.ps>.

[7] D.B. Skalak, Prototype and feature selection by sampling and random mutation hill climbing algorithms, in: International Conference on Machine Learning, 1994, pp. 293–301. Available from <citeseer.nj.nec.com/skalak94prototype.html>.

[8] E.-H. Han, G. Karypis, V. Kumar, Text categorization using weight-adjusted *k*-nearest neighbor classification, in: D. Cheung, Q. Li, G. Williams (Eds.), Proceedings of PAKDD-01, 5th Pacific-Asia Conferenece on Knowledge Discovery and Data Mining, Springer Verlag, Heidelberg, Germany, Hong Kong, China, 2001, pp. 53–65, published in the "Lecture Notes in Computer Science" series, number 2035. Available from <http://link.springer.de/link/service/series/0558/papers/2035/20350053.pdf>.

[9] D.A. Hull, Improving text retrieval for the routing problem using latent semantic indexing, in: W.B. Croft, C.J. van Rijsbergen (Eds.), Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval, Springer Verlag, Heidelberg, Germany, Dublin, Ireland, 1994, pp. 282–289. Available from <http://www.acm.org/pubs/articles/proceedings/ir/188490/p282-hull/p282-hull.pdf>.

[10] D.J. Ittner, D.D. Lewis, D.D. Ahn, Text categorization of low quality images, in: Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, US, 1995, pp. 301–315. Available from <http://www.research.att.com/lewis/papers/ittner95.ps>.

[11] T. Joachims, A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, in: D.H. Fisher (Ed.), Proceedings of ICML-97, 14th International Conference on Machine Learning, Morgan Kaufmann Publishers, San Francisco, US, Nashville, US, 1997, pp. 143–151. Available from <citeseer.nj.nec.com/joachims96probabilistic.html>.

[12] W.T. Chuang, A. Tiyyagura, J. Yang, G. Giuffrida, A fast algorithm for hierarchical text classification, in: Y. Kambayashi, M. Mohania, A. Tjoa (Eds.), Proceedings of DaWaK-00, 2nd International Conference on Data Warehousing and Knowledge Discovery, Springer Verlag, Heidelberg, Germany, London, UK, 2000, pp. 409–418, published in the "Lecture Notes in Computer Science" series, number 1874. Available from <http://www.cs.iastate.edu/yang/Papers/dawak00.ps>.

[13] Z.-H. Deng, S.-W. Tang, D.-Q. Yang, M. Zang, X.-B. Wu, M. Yang, A linear text classification algorithm based on category relevance factors, in: Proceedings of ICADL-02, 5th International Conference on Asian Digital Libraries, ACM Press, New York, US, Singapore, 2002, pp. 88–98.

[14] T. Joachims, Text categorization with support vector machines: learning with many relevant features, in: C. Nédellec, C. Rouveirol (Eds.), Proceedings of ECML-98, 10th European Conference on Machine Learning, no. 1398, Springer Verlag, Heidelberg, Germany, Chemnitz, Germany, 1998, pp. 137–142. Available from <citeseer.nj.nec.com/joachims98text.html>.

[15] H.T. Ng, W.B. Goh, K.L. Low, Feature selection, perceptron learning, and a usability case study for text categorization, in: N.J. Belkin, A.D. Narasimhalu, P. Willett (Eds.), Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval, ACM Press, New York, US, Philadelphia, US, 1997, pp. 67–73. Available from <http://www.acm.org/pubs/articles/proceedings/ir/258525/p67-ng/p67-ng.pdf>.

[16] E.-H. Han, G. Karypis, Centroid-based document classification: analysis and experimental results, in: Principles of Data Mining and Knowledge Discovery, 2000, pp. 424–431. Available from <citeseer.nj.nec.com/han00centroidbased.html>.

[17] V. Lertnattee, T. Theeramunkong, Improving centroid-based text classification using term-distribution-based weighting and feature selection, in: Proceedings of INTECH-01, 2nd International Conference on Intelligent Technologies, Bangkok, Thailand, 2001, pp. 349–355.

[18] T. Theeramunkong, V. Lertnattee, Improving centroid-based text classification using term-distribution-based weighting system and clustering, in: Proceedings of ISCIT-01, 2nd International Symphosium on Communication and Information Technology, Cheingmai, Thailand, 2001, pp. 1167–1182.

[19] R.E. Schapire, Y. Singer, A. Singhal, Boosting and Rocchio applied to text filtering, in: W.B. Croft, A. Moffat, C.J. van Rijsbergen, R. Wilkinson, J. Zobel (Eds.), Proceedings of SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval, ACM Press, New York, US, Melbourne, Australia, 1998, pp. 215–223. Available from <http://www.research.att.com/schapire/cgi-bin/uncompress-papers/SchapireSiSi98.ps>.

[20] W.W. Cohen, Learning to classify English text with ILP methods, in: L. De Raedt (Ed.), Advances in Inductive Logic Programming, IOS Press, Amsterdam, The Netherlands, 1995, pp. 124–143. Available from <http://www.research.whizbang.com/wcohen/postscript/ilp.ps>.

[21] H. Sorensen, M. McElligott, Psun: a profileing system for usenet news, in: Proceedings of CIKM-95, Intelligent Information Agents Workshop, Baltimore, 1995.

[22] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information Processing and Management 24 (5) (1988) 513–523.

[23] A. Singhal, G. Salton, C. Buckley, Length normalization in degraded text collections, Tech. Rep. TR95-1507, 1995. Available from <citeseer.nj.nec.com/singhal95length.html>.

[24] A. Singhal, C. Buckley, M. Mitra, Pivoted document length normalization, in: Research and Development in Information Retrieval, 1996, pp. 21–29. Available from <citeseer.nj.nec.com/singhal96pivoted.html>.

[25] K. Nigam, A.K. McCallum, S. Thrun, T.M. Mitchell, Learning to classify text from labeled and unlabeled documents, in: Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence, AAAI Press, Menlo Park, US, Madison, US, 1998, pp. 792–799, an extended version appears as [1]. Available from <http://www.cs.cmu.edu/knigam/papers/emcat-aaai98.ps>.

[26] G. Siolas, F. d'Alche Buc, Support vector machines based on a semantic kernel for text categorization, in: S.-I. Amari, C.L. Giles, M. Gori, V. Piuri (Eds.), Proceedings of IJCNN-00, 11th International Joint Conference on Neural Networks, vol. 5, IEEE Computer Society Press, Los Alamitos, US, Como, Italy, 2000, pp. 205–209. Available from <http://dlib.computer.org/conferen/ijcnn/0619/pdf/06193581.pdf>.

[27] K.H. Lee, J. Kay, B.H. Kang, U. Rosebrock, A comparative study on statistical machine learning algorithms and thresholding strategies for automatic text categorization, in: M. Ishizuka, A. Sattar (Eds.), Proceedings of PRICAI-02, 7th Pacific Rim International Conference on Artificial Intelligence, Springer Verlag, Heidelberg, Germany, Tokyo, Japan, 2002, pp. 444–453, published in the "Lecture Notes in Computer Science" series, number 2417. Available from <http://link.springer.de/link/service/series/0558/papers/2417/24170444.pdf>.

[28] A.K. McCallum, R. Rosenfeld, T.M. Mitchell, A.Y. Ng, Improving text classification by shrinkage in a hierarchy of classes, in: J.W. Shavlik (Ed.), Proceedings of ICML-98, 15th International Conference on Machine Learning, Morgan Kaufmann Publishers, San Francisco, US, Madison, US, 1998, pp. 359–367. Available from <http://www.cs.cmu.edu/mccallum/papers/hier-icml98.ps.gz>.

[29] M. Craven, D. DiPasquo, D. Freitag, A.K. McCallum, T.M. Mitchell, K. Nigam, S. Slattery, Learning to extract symbolic knowledge from the World Wide Web, in: Proceedings of AAAI-98, 15th Conference of the American Association for Artificial Intelligence, AAAI Press, Menlo Park, US, Madison, US, 1998, pp. 509–516. Available from <http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-11/www/wwkb/overview-aaai98.ps.gz>.

[30] F. Sebastiani, Machine learning in automated text categorization, ACM Computing Surveys 34 (1) (2002) 1–47. Available from <http://faure.iei.pi.cnr.it/fabrizio/Publications/ACMCS02.pdf>.

[31] G. Salton, Automatic Text Processing: The Transformation, Analysis, and Retreival of Information by Computer, Addison-Wesley, Reading, Pensylvania, 1989.

[32] J.J. Rocchio, Relevance feedback in information retrieval, in: G. Salton (Ed.), The SMART REtrieval System: Experiments in Automatic Document Processing, Prentice-Hall, Englewood Cliffs, New Jersey, 1971, pp. 313–323.

[33] S.E. Robertson, K. Sparck-Jones, Relevance weighting of search terms, Journal of American Society for Information Science 3 (27) (1976) 129–146.

[34] N. Fuhr, Models for retrieval with probabilistic indexing, Information Processing and Management 1 (25) (1989) 55–72.

[35] F. Debole, F. Sebastiani, Supervised term weighting for automated text categorization, in: Proceedings of SAC-03, 18th ACM Symposium on Applied Computing, ACM Press, New York, US, Melbourne, US, forthcoming. Available from <http://faure.iei.pi.cnr.it/fabrizio/Publications/SAC03b.pdf>.