Knowledge-Based Systems 000 (2017) 1–12

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A new Centroid-Based Classification model for text categorization

Chuan Liu[a], Wenyong Wang[a,b,*], Guanghui Tu[a], Yu Xiang[a], Siyang Wang[c], Fengmao Lv[a]

[a] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[b] Shanghai Hefu Artificial Intelligence Technology (Group) Company Limited Hefu Institution of UESTC, Chengdu 611731, China
[c] Mathematics Department, University of California San Diego, La Jolla 92093, USA

ARTICLE INFO

Article history:
Received 13 January 2017
Revised 17 August 2017
Accepted 26 August 2017
Available online xxx

Keywords:
Text categorization
Centroid-Based Classifier
Machine learning
Gravitation Model

ABSTRACT

The automatic text categorization technique has gained significant attention among researchers because of the increasing availability of online text information. Therefore, many different learning approaches have been designed in the text categorization field. Among them, the widely used method is the Centroid-Based Classifier (CBC) due to its theoretical simplicity and computational efficiency. However, the classification accuracy of CBC greatly depends on the data distribution. Thus it leads to a misfit model and also has poor classification performance when the data distribution is highly skewed. In this paper, a new classification model named as Gravitation Model (GM) is proposed to solve the class-imbalanced classification problem. In the training phase, each class is weighted by a mass factor, which can be learned from the training data, to indicate data distribution of the corresponding class. In the testing phase, a new document will be assigned to a particular class with the max gravitational force. The performance comparisons with CBC and its variants based on the results of experiments conducted on twelve real datasets show that the proposed gravitation model consistently outperforms CBC together with the Class-Feature-Centroid Classifier (CFC). Also, it obtains the classification accuracy competitive to the DragPushing (DP) method while it maintains a more stable performance. Thus, the proposed gravitation model is proved to be less over-fitting and has higher learning ability than CBC model.

1. Introduction

Automatic Text Categorization (TC) (also known as text classification) is a task of assigning text documents to pre-defined categories. In recent years, TC has received more attention due to a large number of text documents available on the Internet, and it has been widely used in many applications such as text filtering [1] and web page recommendation [2].

Centroid-Based Classifier (CBC) [1,3–7] is one of the most popular TC methods. The basic idea of CBC is that an unlabeled sample should be assigned to a particular class if the similarity of this sample to the centroid of the class is the largest. Compared with other TC methods, CBC is efficient since its computational complexity is linear in the training phase, this merit is important for massive text classification task. Although it has been shown that CBC consistently outperforms other methods such as k-Nearest-Neighbors, Naive Bayes, and Decision Tree on a wide range of datasets [8], CBC often suffers from the model misfit [9,10] when the data is class-imbalanced distribution [11]. This class-imbalanced problem, that implies a high ratio of the number of instances in the majority class to the number of examples in the minority class, is common in many real problems in which the minority class is usually the one that has the highest interest from a learning task. The misfit with imbalanced datasets is that CBC is often biased towards the minority class and, therefore, there is a higher misclassification rate for the minority class and a lower recall rate for the majority class.

To solve the model misfit of CBC, numerous approaches have been proposed, such as Class-Feature-Centroid (CFC) [3], Generalized Cluster Centroid Based Classifier (GCCC) [12], DragPushing (DP) [6] and Large Margin DragPushing (LMDP) [7]. However, the existing improvements of CBC focus on the methods that aim to obtain good centroids in construction or training phase. Therefore, they can not solve the inherent disadvantages of CBC model.

In this paper, the authors believe that different classes should share different similarities to the unlabeled sample in the class-imbalanced dataset. Therefore, the proposed model motivated by Newton's law of universal gravitation concentrates on producing a Voronoi partition [13] that directly alleviates the class-imbalanced problem. Hence, the proposed method is to adjust the classification hyperplane instead of the location of centroids to improve the classification performance. In the proposed model, each class is

* Corresponding author at: School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China.
E-mail addresses: mrliuchuan@foxmail.com, wangwy@uestc.edu.cn (W. Wang).

http://dx.doi.org/10.1016/j.knosys.2017.08.020
0950-7051/

weighted by a mass factor to indicate the data distribution of the corresponding class, and the value of mass factor can be learned from the training set. Then, a new document will be assigned to a particular class with the max gravitational force. The proposed method is empirically evaluated by comparing it with three frequently-used centroid-based methods (i.e., CBC, CFC, and DP) based on twelve famous datasets in the field of text categorization. The experimental results demonstrate that the proposed method works well on the real-world datasets, and outstandingly outperforms the centroid-based approaches (e.g., CBC and CFC). Furthermore, it maintains a more stable performance than the state-of-the-art method.

The remainder of this paper is organized as follows: Section 2 summarizes the related work. Section 3 describes the original CBC. Three representative variants of CBC are presented in Section 4. The proposed gravitation model is introduced in Section 5. The performance of gravitation model is evaluated in Section 6. The paper closes with the conclusions and future works in Section 7.

2. Related work

A growing number of statistical classification methods and machine learning techniques which include Naive Bayes (NB) [14], K-Nearest Neighbor (KNN) [15,16], Neural Network (NN) [17–21], Decision Tree (DT) [22], Support Vector Machines (SVM) [23,24], CBC [1,3–7,25,26], graph-based approach [27] and classifier ensemble approaches [28], have been applied to text categorization in recent years. Naive Bayes is an effective probability model for text mining tasks. However, the performance is poor in case there are no sufficient training documents for each category [14]. KNN is a type of instance-based learning algorithm, but it suffers from several drawbacks [29] such as high storage requirement, low efficiency in classification response, and low noise tolerance. Thus, it is inefficient in online classification [15]. In a neural network based TC, many studies exploited deep learning methods in recent years. Word embeddings are used to represent dense and low-dimensional real-valued vectors. Additionally, the embeddings are capable of solving sparsity problem. For instance, Mikolov et al. represent word2vec in [19–21] and also doc2vec/paragraph vector in [30]. These models have the potential to overcome the weaknesses (i.e., ignoring the ordering/semantics of the words) of bag-of-words models. Decision tree performs well when there is a small number of features. However, it becomes difficult to create a text classifier for a large number of features [22]. SVM has been proved to be a state-of-the-art classifier in TC filed. However, the training of SVM is a very slow process and has become a bottleneck, since the Quadratic Programming (QP) problem that implies high training time complexity $O(n^3)$ and space complexity $O(n^2)$ needs to be solved [31]. Thus, the weakness with SVM is that large-scale training instances will lead to slow learning speed and large buffer memory requirement. To reduce the high computational complexity of SVM, a feasible approach is to reconstruct training set for SVM in the context of the reduction methods [31,32]. The graph-based approach for text classification have proven to achieve good performances as demonstrated in [27]. Also, many classifier ensemble approaches such as bagging [33], boosting [34], random subspace [35], and random forest [36] have been successfully applied to different areas and have achieved excellent performances. However, these approaches are accompanied by poor efficiency.

Several investigations [37,38] indicate that standard learning algorithms suffer from the significant loss of performance in the imbalanced dataset scenario in classification. Therefore, many solutions which can be categorized into three major groups [39] i.e., data sampling [40,41], cost-sensitive learning [42,43] and ensemble techniques [44–46] have been proposed to deal with this problem. Data sampling is to produce a more balanced class distribution that allow the learning algorithms to perform in a similar manner to standard classification. Thus, the main advantage of resampling techniques is that they are independent of the underlying learning algorithms, and it has proved empirically to be an useful solution that applying a preprocessing step to balance the class distribution. Resampling techniques can be mainly categorized into two families, i.e., undersampling methods and oversampling methods, and the main difference between them is dependent on the object that they deal with. Undersampling methods are based on data cleaning techniques to create a subset of the original dataset by eliminating the instances of majority class. A wide variety of undersampling methods such as Wilsonâs edited nearest neighbor (ENN) [47] rule, the one-sided selection (OSS) [48], the condensed nearest neighbor rule (CNN) [49] and the neighborhood cleaning rule [50] have been proposed. Conversely, the main idea of oversampling methods is to interpolate several minority class instances into the original dataset by replicating some instances or creating new instances from existing ones. Some representative works in this area include Synthetic Minority Oversampling TEchnique (SMOTE) [40], Borderline-SMOTE [51], Adaptive Synthetic Sampling [52], Safe-Level-SMOTE [53] and SPIDER2 [54] algorithms. Cost-sensitive learning considers higher costs for the misclassification of instances of the positive class (minority class) respecting the negative class (majority class), and therefore tries to minimize higher cost errors [42,43]. Thus, it is usually a solution that incorporates approaches at the data level, at the algorithmic level, or at both levels combined. For example, in the context of data level, the most popular method is resampling the training dataset according to the cost decision matrix by means of undersampling/oversampling [43] or assigning instance weights [55]. In the context of algorithmic level, the cost-sensitive algorithm based on the decision tree theory assigns the class label to the node that minimizes the classification cost [42,56]. Ensemble techniques construct several simple classifiers from the original data in order to aggregate their predictions when unknown instances are presented. To deal with the class imbalance problem, ensemble-based methods have brought along a growth of attention from researchers [44–46,57–59]. Also, ensemble techniques can be mainly categorized into two groups, i.e., cost-sensitive ensemble learning [57,60–62] and data preprocessing ensemble learning [44,58,63–66]. A complete taxonomy of ensemble techniques for learning with imbalanced classes has been proposed in well-known surveys [39,67].

CBC is relatively theoretically simple and computationally efficient compared with classifiers mentioned above. However, CBC is often plagued for inductive bias [10] or model misfit [9]. Thus, many researchers have proposed the improved methods of CBC, which can be mainly classified into three categories, i.e., choosing supervised term weighting, constructing good centroids in the initial phase, as well as adjusting the position of centroids during the training phase.

As pointed in [68], term weighting is a critical factor that affects the performance of the classifier. Therefore, term weighting methods for TC have gained increasing attention in many literatures. For instance, Lan et al. [69] proposed a new supervised Relevance Frequency (RF) factor for term weighting. Based on the RF factor, Nguyen et al. [70] proposed two robust factors, i.e., Kullback Leibler (KL) divergence and Jensen Shannon (JS) divergence, to overcome the problem that RF factor does not penalize terms appearing equally often in both classes. Also, the term weighting comprised of term frequency factor, which relates more to the individual document, and collection frequency factor that is computed over the entire collection of documents has been proved to improve the performance of CBC in many literatures [69,71–74].

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020

The methods that aim to obtain good centroids of CBC in the construction phase are also introduced. For example, Guan et al. [3] designed a CFC classifier which combines inter-class term index with inner-class term index to obtain better prototype vectors than CBC. The experimental results in [3] show that CFC is better than the compared classifiers. However, they employed the testing samples to construct the centroid vectors, which is not reasonable since test samples are the unknown information. The experimental results in this paper based on cross-validation scheme indicate that CFC has a low generalization ability. Additionally, Lertnattee et al. [5] investigated the effectiveness of the frequently-used normalization functions and proposed a new type of class normalization method (i.e., term-length normalization) in the construction phase of centroid vectors. As it is argued that using only border instances rather than all instances to construct centroid vectors can obtain higher generalization accuracy, Wang et al. [11] proposed a Border-Instance-based Iteratively Adjusted Centroid Classifier (IACC-BI) that relies on the border instances found by some routines, e.g., 1-Nearest-and-1-Furthest-Neighbors strategy, to construct centroid vectors.

In addition, many researchers are also committed to improving the performance of CBC by adjusting the position of centroids during the training phase. For example, Wang et al. [4] proposed a centroid-based TC method called Smoothing Listwise Ranking Centroid Method (SLRCM) that uses a smoothing ranking loss function which pays enough attention to the position of the target class. Cataltepe et al. [25] provided an improvement of CBC, which starts from the centroids of each class and iteratively increases the weight of misclassified training instances on the centroid computation until the validation errors increasing. This improved algorithm results in smaller test error than CBC in seven out of nine datasets. Miao et al. [75] introduced a pairwise optimized Rocchio algorithm, which dynamically adjusts the position of prototype vectors between pairs of classes. Wu et al. [76] designed a refinement tree approach to correct model misfit, and applied it to improve Rocchio and Naive Bayes algorithm. Instead of improving the classification technique itself to make it more flexible, this approach successively refines the classification model based on the prediction errors of the training data. Also, Tan et al. [6] presented DP approach to iteratively adjust the prototype vectors, which takes the advantage of misclassified samples to drag the centroid of correct category and push the centroid of the incorrect category. On the basis of DP strategy, Tan et al. [7] further extended DP to Large Margin DragPushing (LMDP) that refines the centroid classifier model by utilizing not only training errors but also training margins. Tan et al. [77] also explored the use of Error-Correcting Output Codes (ECOC) to boost centroid-based classifier, and then applied Model-Refinement to decrease the bias introduced by ECOC. Pang et al. [12] proposed a Generalized Cluster Centroid-Based Classifier (GCCC) that exploits a clustering algorithm to integrate KNN and Rocchio algorithm. In fact, the skewed distribution of data is the main reason that leads to the misfit of CBC. Thus, the improvements that focus on the adjustment of centroid position can not solve the misfit problem of CBC model significantly.

3. The centroid-Based Classifier

In centroid-based text categorization, the documents are commonly represented by the Vector Space Model (VSM) [78], where all the training documents are used to generate a lexicon $\mathcal{F} = \{t_1, t_2, \ldots, t_{|\mathcal{F}|}\}$ ($t_k, k \in [1, |\mathcal{F}|]$ represents kth term in the lexicon), and each document is regarded as a vector in $|\mathcal{F}|$-*dimension* feature space. Then, the *Term Frequency and Inverse Document Frequency (TFIDF)* is usually employed to convert the textual document into a numeric vector, which uses the following formula [70]:

$$tfidf(t_k, \boldsymbol{d}_i) = tf(t_k, \boldsymbol{d}_i) \times \log \frac{|\ D\ |}{|\ D(t_k)\ |}, \tag{1}$$

where $tf(t_k, \boldsymbol{d}_i)$ is the frequency of term t_k being in document $\boldsymbol{d}_i$, $|D|$ denotes the number of all training documents, and $|D(t_k)|$ denotes the number of documents containing t_k in text collection D. Then, the normalization for the term weighting is performed as follows [69]:

$$w_{ki} = \frac{tfidf(t_k, \boldsymbol{d}_i)}{\sqrt{\sum_{z=1}^{|\mathcal{F}|} (tfidf(t_z, \boldsymbol{d}_i))^2}}, \tag{2}$$

where w_{ki} is the normalized weight of term t_k in document $\boldsymbol{d}_i$.

After giving the normalized representation of documents, class centroid $\boldsymbol{c}_j$ is obtained by summing up all document vectors in class C_j and then normalizing the result by its size. Hence, the formal description of a class centroid is

$$\boldsymbol{c}_j = \frac{\sum_{\boldsymbol{d}_i \in C_j} \boldsymbol{d}_i}{\| \sum_{\boldsymbol{d}_i \in C_j} \boldsymbol{d}_i \|_2}, \tag{3}$$

where $\|*\|_2$ denotes 2-norm. Thus, the similarity between an unlabeled document $\boldsymbol{d}$ and centroid $\boldsymbol{c}_j$ can be measured by the cosine function [11], which is given by

$$\cos(\boldsymbol{d}, \boldsymbol{c}_j) = \frac{\boldsymbol{d} \cdot \boldsymbol{c}_j}{\| \boldsymbol{d} \|_2 \times \| \boldsymbol{c}_j \|_2}, \tag{4}$$

where "$\cdot$" denotes the dot-product of the two vectors.

Finally, an unlabeled document $\boldsymbol{d}$ will be assigned to the class whose centroid vector is most similar to vector $\boldsymbol{d}$, that is, the class of $\boldsymbol{d}$ is given by

$$Class(\boldsymbol{d}) = \arg\max_{j} \cos(\boldsymbol{d}, \boldsymbol{c}_j). \tag{5}$$

4. The improved methods

In this section, three representative variants of CBC that are frequently-used as the baseline algorithms for the purpose of comparison are reviewed, and the disadvantages of these variants are also highlighted.

4.1. DragPushing method

DP method [6] is an effective refinement strategy to deal with misfit problem of CBC model. It takes the advantage of misclassified samples (i.e., training errors) to drag the centroid of the correct class towards misclassified samples and push the centroid of the incorrect class away from misclassified samples. Thus, if one document $\boldsymbol{d}$ labeled as class A is misclassified into class B, both centroid $\boldsymbol{c}_A$ and $\boldsymbol{c}_B$ should be modified by the following formulas:

$$\boldsymbol{c}_A := \boldsymbol{c}_A + \eta \times \boldsymbol{d}, \tag{6}$$

$$\boldsymbol{c}_B := \boldsymbol{c}_B - \eta \times \boldsymbol{d}, \tag{7}$$

where η denotes the learning rate which controls the step-size of updating operation. Formula (6) and (7) are respectively called "drag" and "push". After having executed the "drag" and "push" processes, the centroid of class A would share more similarity with document $\boldsymbol{d}$, while the similarity between document $\boldsymbol{d}$ and class B would be reduced.

However, if we update the centroids one by one, some centroids may be moved to and fro. In order to avoid this problem, the authors apply batch-update mode to adjust the centroids. The batch-update means that each refinement uses all misclassified samples

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020

to drag or push centroids. The batch-update formula for each centroid is as follows:

$$\boldsymbol{c}_A := \boldsymbol{c}_A + \eta \times \left(\sum_{\boldsymbol{d} \in \phi} \boldsymbol{d} - \sum_{\boldsymbol{d} \notin \varphi} \boldsymbol{d} \right), \tag{8}$$

where ϕ is the collection of samples, labeled as class A, that is misclassified into other classes, φ is the set of samples that do not belong to class A but misclassified into class A. In one iteration of refinement, it needs to classify all the training documents and make use of the misclassified samples to adjust the centroids by Formula (8).

4.2. LMDP method

DP method adopts only one criterion (i.e., training error) as its optimization objective, which can not guarantee the generalization ability. Therefore, motivated by the large margin of SVM, Large Margin DragPushing (LMDP) [7] method makes use of the margin conception to boost the generalization ability of DP. Since DP is on-line update mode, the margin definition in SVM can not be utilized directly. Therefore, the authors applied an alternative definition named as "hypothesis-margin" [79] for each document. The margin of a hypothesis regarding a document is the distance between the hypothesis and the closest hypothesis that assigns an alternative label to the given document. Thus, the hypothesis margin of document $\boldsymbol{d}$ for centroid-based classifier is defined as below:

$$HM(\boldsymbol{d}) = (\mathrm{Sim}(\boldsymbol{d}, \boldsymbol{c}_R) - \mathrm{Sim}(\boldsymbol{d}, \boldsymbol{c}_M)), \tag{9}$$

where $\boldsymbol{c}_R$ is the most similar centroid to document $\boldsymbol{d}$, and $\boldsymbol{c}_M$ denotes the centroid that shares the second similarity with document $\boldsymbol{d}$.

Then, corresponding to the definition of training error, a generalized error called "*margin error*" was introduced by the authors. In the definition of margin error, if the hypothesis-margin of sample $\boldsymbol{d}$ is bigger than zero but smaller than a small positive constant that is called "*min margin*" denoted by θ, it should be employed to adjust the centroids. As a result, the batch-update formula combining training error and margin error can be written down as follows:

$$\boldsymbol{c}_A := \boldsymbol{c}_A + \eta \times \left(\sum_{\substack{\boldsymbol{d} \in \text{class A} \\ HM(\boldsymbol{d}) < 0}} \boldsymbol{d} - \sum_{\substack{\boldsymbol{d} \notin \text{class A} \\ HM(\boldsymbol{d}) < 0}} \boldsymbol{d} + \lambda \times \left(\sum_{\substack{\boldsymbol{d} \in \text{class A} \\ 0 < HM(\boldsymbol{d}) < \theta}} \boldsymbol{d} - \sum_{\substack{\boldsymbol{d} \notin \text{class A} \\ 0 < HM(\boldsymbol{d}) < \theta}} \boldsymbol{d} \right) \right), \tag{10}$$

where λ is a constant parameter to balance training error with margin error. It is evident to see that not only the misclassified documents, but also small hypothesis-margin documents are employed by LMDP method to improve the performance of CBC.

4.3. CFC method

Different from the previous improvements that focus on iteratively updating centroids in the training phase, CFC [3] classifier tries to obtain good centroids during the construction phase to improve the accuracy of classification. In comparison with CBC to create prototype vectors, CFC pays more attention on term distribution of the corpus to improve the quality of centroid. Thus, the weight in CFC for term t_i of centroid $\boldsymbol{c}_j$ is calculated as:

$$w_{ij} = b^{\frac{DF_{t_i}^j}{|C_j|}} \times \log\left(\frac{|C|}{CF_{t_i}}\right), \tag{11}$$

where $DF_{t_i}^j$ is the frequency of documents that contain term t_i in class C_j, $|C_j|$ is the number of documents in class C_j, $|C|$ is the number of classes, CF_{t_i} is the number of classes containing term t_i, and b is a constant larger than one as suggested in reference [3].

Apparently, the formula above is comprised of two components, the first component $b^{\frac{DF_{t_i}^j}{|C_j|}}$ $(b > 1)$ is called the inner-class term index, and the second one $\log(\frac{|C|}{CF_{t_i}})$ represents the inter-class term index. If a term t_i appears many times in documents of category C_j, and rarely appears in documents of other categories, both inner-class and inter-class term indices would gain high values. Thus, the weight of term t_i in centroid $\boldsymbol{c}_j$, denoted by w_{ij}, would finally obtain a high value.

It is worth mentioning that CFC adopts denormalized prototype vector for prediction since normalized prototype vector decreases its discriminative capability, and the experiments conducted in literature [3] show that the performance of CFC is generally good when the constant parameter b is between 1 and e.

4.4. The analyses of the variants

Regarding CFC, Nguyen et al. [70] argued that CFC is inherently biased toward the abundant classes. Thus, it invariably leads to poor performance on the class-imbalanced dataset. Also, the results of experiments performed in this paper based on five-fold cross-validation scheme indicate that CFC has a low generalization ability.

Though DP (or LMDP) method performs well in many cases, it is also possible to obtain poor accuracy in some particular situations. For example, class A, B and D are distributed as shown in Fig. 1, where $\boldsymbol{c}_A$, $\boldsymbol{c}_B$ and $\boldsymbol{c}_D$ are the original centroids of class A, B and D, respectively. If DP (or LMDP) is adopted to refine the centroids, as a result, the centroids of class B and D will be moved out of their geometry regions (i.e., $\boldsymbol{c}'_B$ and $\boldsymbol{c}'_D$). Supposing there is a class E that is located beside the left side of class B, and the refined centroid of class B (i.e., $\boldsymbol{c}'_B$) exactly falls into class E, hence some instances would be inevitably misclassified no matter how centroids are moving. In this case, DP (or LMDP) is no longer suitable for such class-imbalanced dataset.

5. The proposed Gravitation Model

In this section, a new Centroid-Based Classification Model, i.e., Gravitation Model (GM), is introduced to easily overcome the inherent shortcomings (or biases) of CBC in the class-imbalanced dataset. The proposed model concentrates on the adjustment of classification hyperplane to reduce the biases inherent in CBC, and it is entirely different from the previous works [3–7,11,25] which obtain the centroids with good initial term weights in construction phase or modify the position of centroids during training phase. In the rest of this section, the motivation of GM is first illustrated and followed by a formal definition, then a detailed algorithm description of GM is given.

5.1. The motivation of Gravitation Model

GM is motivated by Newton's law of universal gravitation which states that a particle attracts every other particle in the universe by a force. The magnitude of the attractive force F between two objects is as follows:

$$F = \frac{GMm}{r^2}, \tag{12}$$

where G denotes the gravitational constant, M is the mass of the first object, m is the mass of the second object and r is the distance between the two objects. This formula indicates that the force is proportional to the product of masses of the two objects and inversely proportional to the square of the distance between them.

Suppose the mass of object A is M_A, the mass of object B is M_B, and the mass of object S, which located between object A and B,

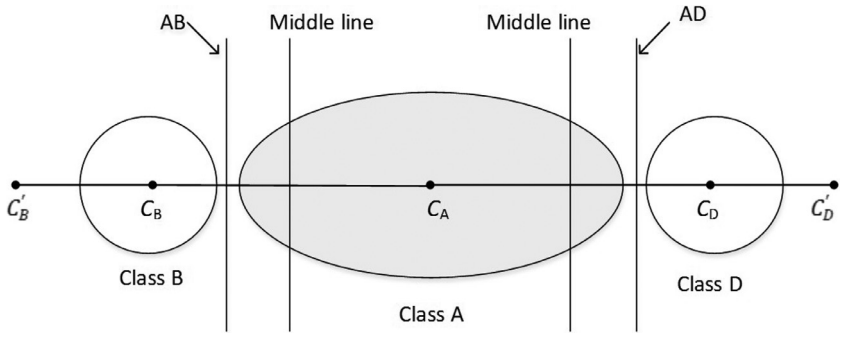

Fig. 1. Outline of refined centroids of class B and D.

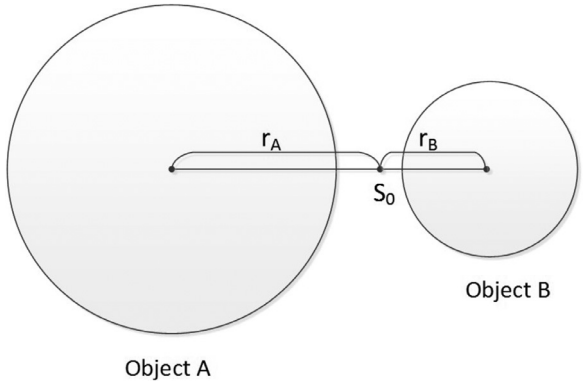

Fig. 2. Gravitation equilibrium point S_0 between object A and B.

is m ($M_A > M_B \gg m$). Moreover, the distance from object S to object A and B are r_A and r_B, respectively. As a result, the attractive force between object S and object A (or B) is F_A (or F_B). According to the law of universal gravitation, there is a Lagrange point (i.e., S_0) lying on the line connecting the object A and B, where the gravitational attraction of M_A to object S counteracts M_B's gravitational attraction to object S, that is $F_A = F_B$, as shown in Fig. 2. If object S lies on the left of point S_0, then object A's attractive force will be greater than the object B's gravitational pull on object S (i.e., $F_A > F_B$), thus object S will move toward object A. On the contrary (i.e., $F_A < F_B$), object S will move to object B.

5.2. The definition of Gravitation Model

Similar to the universal gravitation, object A (or B) can be seen as class A (or B), and object S can be seen as the unknown sample $\boldsymbol{d}$ in the vector space. Hence, the label of sample $\boldsymbol{d}$ will be determined by the attractive forces from class A and class B. For example, if the sample $\boldsymbol{d}$ locates on the left of Lagrange point S_0, then the sample $\boldsymbol{d}$ will be classified into category A due to $F_A > F_B$. Conversely, if $F_A < F_B$, the sample $\boldsymbol{d}$ will be classified into category B. Therefore, the classification hyperplane in GM is no longer the hyperplane that class A and B share the same similarity with the unknown sample $\boldsymbol{d}$ as in original CBC but Lagrange hyperplane that class A and B share the same attractive force. In the vector space, Lagrange hyperplane between two classes is defined as follows:

$$\frac{M_A}{r_A^2} = \frac{M_B}{r_B^2}, \tag{13}$$

where r_A (or r_B) is the distance from sample $\boldsymbol{d}$ to the centroid of class A (or class B). In fact, $\frac{1}{r_A^2}$(or $\frac{1}{r_B^2}$) can be regarded as the measure of similarity between sample $\boldsymbol{d}$ and centroid of class A (or class B). Therefore, the formula above can be rewritten as below:

$$M_A * Sim(\boldsymbol{d}, \boldsymbol{c}_A) = M_B * Sim(\boldsymbol{d}, \boldsymbol{c}_B), \tag{14}$$

where $Sim(\boldsymbol{d}, \boldsymbol{c}_A)$(or $Sim(\boldsymbol{d}, \boldsymbol{c}_B)$) is the measure of similarity between sample $\boldsymbol{d}$ and the centroid of class A(or B), and M_A(or M_B) is unknown mass factor which can be learned from the training set. The objective of learning mass factor is to obtain the best Lagrange hyperplane so that the classification error (i.e., 1-accuracy) is the lowest. In testing phase, GM first calculates the attractive force between the unidentified sample $\boldsymbol{d}$ and the centroid of each class, which uses the following formula:

$$F(\boldsymbol{d}, \boldsymbol{c}_j) = M_j * Sim(\boldsymbol{d}, \boldsymbol{c}_j), \tag{15}$$

where M_j, $j \in [1, |C|]$ denotes mass factor of class j. Then, the class of document $\boldsymbol{d}$ is directly determined by assigning it to the class with the max attractive force, that is

$$Class(\boldsymbol{d}) = \arg\max_j F(\boldsymbol{d}, \boldsymbol{c}_j). \tag{16}$$

Obviously, the mass factors introduced by GM are used to indicate the class distribution in the training phase, and then they are applied to determine the category of unidentified samples in the testing phase. In this framework, learning mass factors leads to a Voronoi partition [13] that directly alleviates the class-imbalanced problem.

As shown in Fig. 1, for an abundant (majority) class (i.e., C_A), its samples tend to be assigned to its adjacent categories incorrectly due to its wide distribution range in the vector space, which reduces the recall (see Eq. (21)) of this category. In contrast, regarding a rare (minority) class (i.e., C_B or C_D), the samples of its neighbor categories are prone to be classified to this category because of its narrow distribution range, thereby reducing its precision (see Eq. (20)). This is the main reason that the original CBC model has the poor performance on class-imbalanced data. However, this problem can be easily solved by adding a mass factor since the region a centroid covers would be expanded or contracted equally in all directions in GM.

5.3. Learning mass factor M

The key to GM is to determine the inherent mass factor M of each class so that the model fits the training data well. The idea of learning mass factors is to use the misclassified samples to guide the update of factor M_i, $i \in [1, |C|]$ until the number of misclassified samples falls to zero or a small constant. The way of learning factor M has two typical modes i.e., stochastic learning and batch-update learning. Since the time complexity of stochastic learning and batch-update learning are linear, GM is still a linear model. The learning ways are concisely introduced as follows.

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020

5.3.1. Stochastic learning algorithm

This algorithm updates the corresponding two mass factors for each misclassified sample. For instance, if document $\boldsymbol{d}_i$ that belongs to class C_i is misclassified into class C_j. To classify $\boldsymbol{d}_i$ correctly, it is needed to increase the attractive force of class C_i on sample $\boldsymbol{d}_i$ while correspondingly decrease the force between class C_j and sample $\boldsymbol{d}_i$. Thus, the mass factor M_i should be enlarged while M_j should be reduced. The concrete updating formulas for M_i and M_j are as follows:

$$M_i := M_i + \xi, \tag{17}$$

$$M_j := M_j - \xi, \tag{18}$$

where ξ is a constant value given by the user, which is used to control the updating strength.

The pseudo code of stochastic learning algorithm is shown in Algorithm 1, where the mass factors are initialized to one, then

Algorithm 1 Stochastic Learning Mass (SLA).

Require:
X : the whole training set with P classes and N samples, where $\boldsymbol{d}_j^n$ repesents the nth sample labeled as class C_j;
$\boldsymbol{c}_j$: the centroid of class C_j;
ξ : the step strength of updating mass factors;
ε : the threshold of classification error rate;
Mlter: the maximum number of iterations.
Ensure:
M_j :the mass factor of class C_j $(1 \le j \le P)$.

```
for j = 1; j ≤ P do
    M_j ← 1;
end for
R^new ← 1;
Repeat ~Iter++;
error ← 0;
R^old ← R^new;
for n = 1; n ≤ N do
    ĵ ← arg max_{i=1,···,P} M_i * Sim(d_j^n, c_i);
    if j ≠ ĵ then
        M_j ← M_j + ξ;
        M_ĵ ← M_ĵ − ξ;
        error++;
    end if
end for
R^new ← error/N;
Until ε > (R^old − R^new)/R^old OR Iter > Mlter;
Return M_j~(1 ≤ j ≤ P);
```

the GM is used to classify the training samples one by one. For a misclassified sample, the corresponding mass factors are updated (i.e., expanding or contracting the region of class over which it covers), until the difference of classification error between two adjacent iterations is less than the given threshold (denoted by ε), or the number of iterations reaches the maximum.

5.3.2. Batch-update learning algorithm

It is different from stochastic learning algorithm where each misclassified document is followed by an updating step of corresponding mass factors, the batch-update learning algorithm updates the mass factors after classifying all the training samples. Hence, the formula for updating the mass factor M_j of the class C_j is as follows:

$$M_j := M_j + (\beta_{j1} - \beta_{j2}) * \xi, \tag{19}$$

where β_{j1} is the number of documents labeled as class C_j but misclassified into other classes, and β_{j2} refers to the number of documents misclassified into class C_j.

6. Experiments

In this section, the performance of GM is evaluated across a range of text collections by comparing it with CBC, CFC and DP. SVM is not included in the experiments because of its time complexity. Our previous work [80] can be referred to for the performance of SVM in text categorization.

The experiments can be divided into two phases. In the first phase, the experiments are performed on twelve real datasets typically used in the context of TC. In the second phase, the binary classification experiments are carried out on twenty artificially generated datasets to further explore the empirical merit of GM on highly class-imbalanced data. Finally, the inherent advantages of GM compared to CBC are analyzed in a real dataset. The source codes associated with this paper are available on our Web site[1].

6.1. Datasets

The benchmark collections used in the experiments are *20-Newsgroups*[2], *Reuters-21578*[3] and other *ten datasets* from Karypis Lab[4]. They are concisely introduced as follows:

Newsgroups — it is a benchmark corpus typically used in the research field of text categorization (or text clustering). This corpus consists of 19,997 articles that are organized into 20 different categories, and it is highly balanced since each category has nearly 1000 texts.

Reuters — it was originally gathered by Carnegie Group, Inc. and Reuters, Ltd. There are several versions such as Apte' split 90 categories and Apte' split 115 categories used in literatures. The version of Apte' split 90 categories which contains 11,406 texts for 90 categories are used in the experiments. The instance distribution over the 90 categories is highly imbalanced.

Datasets from Karypis Lab — ten datasets including *cacmcisi, classic, fbis, hitech, new3, ohscal, re0, re1, tr12,* and *tr23* are picked out from Karypis' dataset. To guarantee their diversity, these collections come from different sources such as Web ACE project, Broadcast of TREC, medical science, etc. The detail description of each collection can be referred to literature [81].

Before the experiments, several preprocessing steps are applied on these datasets. Firstly, the samples that have multiple labels are deleted since the experiments focus on single-label TC task, and the categories whose samples are less than ten are also deleted. Secondly, a stop word list is used to remove common words, and the Porters stemming algorithm [82] is adopted to compute the root of each word. Then, the terms that occur less than four times in the whole dataset are removed to reduce the dimension of feature space. Also, the term-weighting scheme i.e., TFIDF [70] is used to map the terms of each document into a compact vector representation. The general characteristics of datasets are listed in Table 1 after the pre-processing steps above.

6.2. The performance metrics

To evaluate classification performance, some indices such as precision, recall, $macroF_1$, and $microF_1$ are adopted as performance metrics. Suppose the number of instances classified correctly into

[1] https://github.com/liuchuan-uestc/GM.
[2] http://www.ai.mit.edu/people/jrennie/20Newsgroups/.
[3] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html.
[4] http://glaros.dtc.umn.edu/gkhome/fetch/sw/cluto/datasets.tar.gz.

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020

Table 1
Summary of datasets used in the experiments.

Dataset	Classes	Instances	Features
Newsgroups	20	13,128	29,949
Reuters	44	11,195	8843
cacmcisi	2	4663	2601
classic	4	7094	6562
fbis	17	2463	2000
hitech	6	2301	10,287
new3	44	9558	29,005
ohscal	10	11,162	10,831
re0	13	1504	2560
re1	25	1657	3383
tr12	8	313	5329
tr23	6	204	5466

class C_m is a, classified incorrectly into C_m is b, and the number of instances in C_m classified into other classes is c. The precision (p_m) and recall (r_m) typically used to evaluate the classification performance of class C_m are respectively defined as follows:

$$p_m = \frac{a}{a+b}, \tag{20}$$

$$r_m = \frac{a}{a+c}. \tag{21}$$

Usually, there is an inverse relationship between precision and recall. It is therefore possible to increase one at the cost of reducing the other. Thus, F_1 introduced to measure the average performance for each class is a weighted combination of precision and recall, which is defined as below:

$$F_1(r, p) = \frac{2pr}{p+r}. \tag{22}$$

Thus, $macroF_1$ is computed by averaging F_1 of all categories, while $microF_1$ is calculated by averaging the precision and recall of all instances. Since $macroF_1$ gives the weight to all categories equally, it will mainly be influenced by the performance of rare categories. In contrast, $microF_1$ treats all instances equally, it will be dominated by the performance of common categories.

6.3. Experiments design

The five-fold cross-validation scheme is applied to evaluate the performance of algorithms. That is, each dataset is randomly partitioned into five subsets. Each subset is selected for testing and the remaining four subsets are utilized for training. Finally, the average performance of five testing results is calculated to decrease the variance of the estimator.

With respect to CFC, the denormalized prototype vector is used for prediction as it was proved to be better than the normalized prototype vector as presented in literature [3], and the parameter b is fixed to $e - 1.5$ (≈ 1.218). Moreover, cross-validation scheme was not applied in the evaluation of CFC in literature [3] (CFC was trained by the whole dataset and tested by selecting three quarters of all samples[5]). Thus, in our experiments, CFC is investigated in two ways i.e., five-fold cross-validation and the method as in literature [3] to give more reasonable results.

Regarding DP method, the learning rate α is set to 0.01, and the maximum iteration *MaxIter* equals 50. For GM, the stochastic learning algorithm is adopted to learn the mass factors. The parameters ξ and ε are set to 0.0001, and the maximum iteration *MaxIter* is also set to 50 in all corpora.

[5] See the source code of CFC posted by authors on https://github.com/jzhou77/CFC.

Table 2
The comparison of different classifiers in $microF_1$.

Datasets	GM	CBC	CFC-CV	DP	CFC
Newsgroups	0.887	0.886	0.773	**0.899**	0.968
Reuters	0.874	0.783	0.705	**0.893**	0.952
cacmcisi	0.927	0.875	0.910	**0.944**	0.998
classic	0.930	0.884	0.811	**0.948**	0.887
fbis	**0.820**	0.794	0.719	0.800	0.901
hitech	**0.710**	0.704	0.532	0.690	0.969
new3	0.799	0.779	0.570	**0.821**	0.984
ohscal	**0.773**	0.761	0.549	**0.773**	0.934
re0	0.805	0.762	0.621	**0.815**	0.919
re1	**0.853**	0.815	0.680	0.849	0.978
tr12	**0.908**	0.899	0.674	**0.908**	1.000
tr23	0.814	0.790	0.637	**0.859**	0.987

Table 3
The comparison of different classifiers in $macroF_1$.

Datasets	GM	CBC	CFC-CV	DP	CFC
Newsgroups	0.883	0.883	0.765	**0.899**	0.972
Reuters	**0.780**	0.716	0.571	0.764	0.963
cacmcisi	0.921	0.882	0.912	**0.938**	0.998
classic	0.936	0.908	0.856	**0.951**	0.912
fbis	0.780	0.764	0.654	**0.782**	0.907
hitech	**0.677**	0.670	0.473	0.657	0.971
new3	0.809	0.797	0.603	**0.832**	0.988
ohscal	**0.763**	0.758	0.535	**0.763**	0.933
re0	0.794	0.782	0.440	**0.800**	0.932
re1	**0.787**	**0.787**	0.539	0.784	0.978
tr12	**0.908**	0.900	0.658	0.906	1.000
tr23	0.807	0.793	0.581	**0.847**	0.992

For concise of reference, "CFC" is used to denote the original CFC as in literature [3], while "CFC-CV" denotes CFC with five-fold cross-validation evaluation method. Finally, the distance between the sample $\boldsymbol{d}_i$ and the centroid $\boldsymbol{c}_j$ is defined as follows [83]:

$$dis_{ij} = 1 - \cos(\boldsymbol{d}_i, \boldsymbol{c}_j), \tag{23}$$

where $dis_{ij} \in [0, 1]$.

6.4. Multi-class experiments

The performance comparisons in $microF_1$ and $macroF_1$ are listed in Tables 2 and 3. The max value in each row has been highlighted in bold without considering the performance of CFC since the evaluation method of CFC is different from other algorithms. From Tables 2 and 3, the following conclusions can be drawn.

GM consistently outperforms CBC in $microF_1$ and $macroF_1$. The $microF_1$ of GM is respectively 9.02%, 5.2%, 4.6%, 4.3%, and 3.8% higher than CBC on *Reuters, cacmcisi, classic, re0*, and *re1*, and is slightly better than CBC on the remaining datasets. Meanwhile, the $macroF_1$ of GM respectively beats CBC by 6.48%, 3.9%, and 2.8% on *Reuters, cacmcisi*, and *classic*. In the rest of datasets, the $macroF_1$ of GM is also obtained small increase compared with CBC.

CFC has the best performance while CFC-CV has the worst performance compared with other algorithms on most of the datasets. The difference in performance between CFC and CFC-CV is due to the reason that CFC uses the whole dataset to calculate prototype vectors while CFC-CV only adopts the training set. The $microF_1$ and $macroF_1$ of CFC-CV are lower than CBC on all datasets except *cacmcisi* which includes only two classes. Therefore, we hypothesize that CFC-CV is good at learning datasets with a small number of classes. On the contrary, the $microF_1$ and $macroF_1$ of CFC have an overwhelming advantage compared with others on eleven datasets. Therefore, this is also an indication that CFC has a low generalization ability (or overfitting) on training data.

DP is the best centroid-based classifier, and GM has the approximate performance as DP on most of the datasets. As shown in

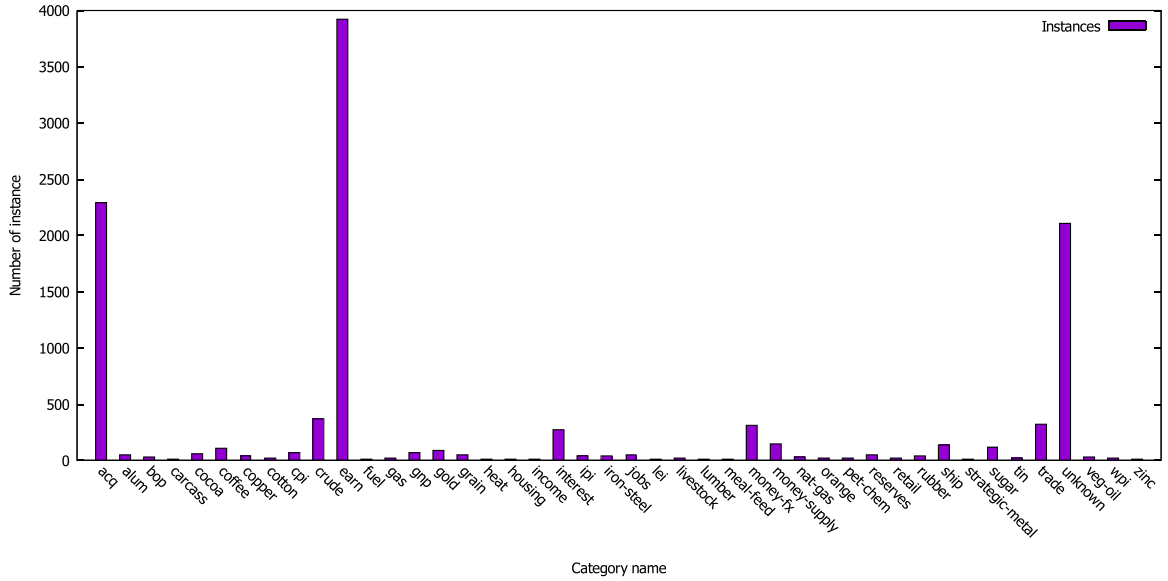

Fig. 3. The instance number of each category on *Reuters*.

Table 4
The summary of artificially generated class-imbalanced datasets.

Datasets	Positive	Negative	Name of Positive Class	Ratio
D1	1000	12,128	comp.graphics	1:12
D2	1000	12,128	comp.os.ms-windows.misc	1:12
D3	409	12,719	comp.sys.ibm.pc.hardware	1:31
D4	998	12,130	comp.sys.mac.hardware	1:12
D5	1000	12,128	comp.windows.x	1:12
D6	863	12,265	rec.autos	1:14
D7	142	12,986	rec.sport.hockey	1:91
D8	527	12,601	talk.politics.guns	1:23
D9	1000	12,128	talk.politics.misc	1:12
D10	1000	12,128	talk.religion.misc	1:12
D11	1159	10,003	Antibodies	1:8
D12	709	10,453	Carcinoma	1:14
D13	764	10,398	DNA	1:13
D14	1001	10,161	In-Vitro	1:10
D15	864	10,298	Molecular-Sequence-Data	1:11
D16	1621	9541	Pregnancy	1:5
D17	1037	10,125	Prognosis	1:9
D18	1297	9865	Receptors	1:7
D19	1450	9712	Risk-Factors	1:6
D20	1260	9902	Tomography	1:7

Table 5
The comparison of different classifiers on the imbalanced datasets in $microF_1$.

Datasets	GM	CBC	CFC-CV	DP
D1	0.968	0.924	0.939	**0.979**
D2	0.964	0.945	0.938	**0.974**
D3	0.961	0.902	0.971	**0.978**
D4	0.971	0.949	0.951	**0.983**
D5	0.929	0.883	0.955	**0.955**
D6	0.984	0.974	0.964	**0.988**
D7	0.984	0.974	0.964	**0.988**
D8	0.966	0.934	0.969	**0.977**
D9	**0.941**	0.889	0.929	0.939
D10	0.940	0.885	0.925	**0.947**
D11	**0.940**	0.906	0.885	0.935
D12	0.958	0.928	0.942	**0.965**
D13	0.941	0.910	0.923	**0.947**
D14	0.935	0.866	0.906	**0.935**
D15	0.956	0.913	0.934	**0.962**
D16	0.962	0.960	0.895	**0.967**
D17	**0.926**	0.824	0.909	0.923
D18	0.933	0.867	0.899	**0.947**
D19	**0.937**	0.884	0.879	0.936
D20	0.958	0.933	0.912	**0.966**

Table 2, GM and DP obtain the same $microF_1$ that is the best performance on *ohscal* and *tr12*. Except for CFC, GM obtains the best performance in five out of twelve datasets, while DP achieves in nine out of twelve datasets. Also, a similar trend can be seen from Table 3, where GM remains the leader with five datasets, and DP takes the crown in eight datasets. From Tables 2 and 3, it should be appreciated that GM is always a very close runner-up when DP is the winner. However, it is also worth noting that DP has poorer performance than CBC in some of the datasets such as *hitech* and *re1*, which does not happen on GM. Thus, DP can not reduce the inherent bias of CBC in some particular situations. GM undoubtedly proves itself to be more stable than DP.

6.5. Binary class experiments

To further verify the performance of GM on the class-imbalanced dataset, twenty highly imbalanced datasets areartificially generated by using *Newsgroups* and *ohscal*. The first ten of them are produced by *Newsgroups* and the remaining ten are generated by *ohscal*. As a common method in practice [70], the method used to generate these datasets is keeping one class as a concrete class and the rest of categories as negative. The details of datasets are shown in Table 4, where the "*Ratio*" denotes "*Positive:Neagtive*" for each dataset. It is clear to see that these datasets have a skewed class distribution with ratio varied from 1:5 to 1:91.

The experimental results for the generated datasets are shown in Tables 5 and 6. It is easy to see that CBC is better than CFC-CV in $macroF_1$ on all datasets and in $microF_1$ on seven out of twenty datasets, which also indicates that CFC-CV is not good at skewed distribution data in TC. Besides, GM outperforms CBC in $microF_1$ on all datasets and in $macroF_1$ on seventeen datasets, while DP only beats CBC in $macroF_1$ on fourteen datasets. This observation also provides the evidence that DP is not stable as GM.

6.6. GM performance analyses

To deep understand the rationale of GM, a more detailed and visualized comparison between GM and CBC on *Reuters* is presented in this section. Fig. 3 shows the instance number of each

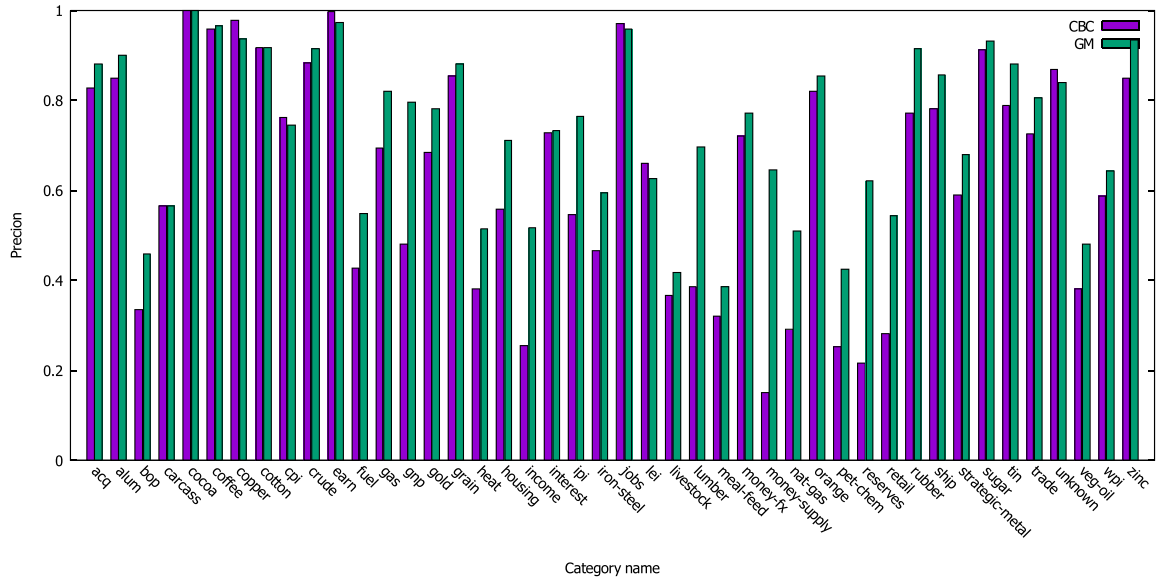

Fig. 4. The precision of each category using CBC and GM on *Reuters*.

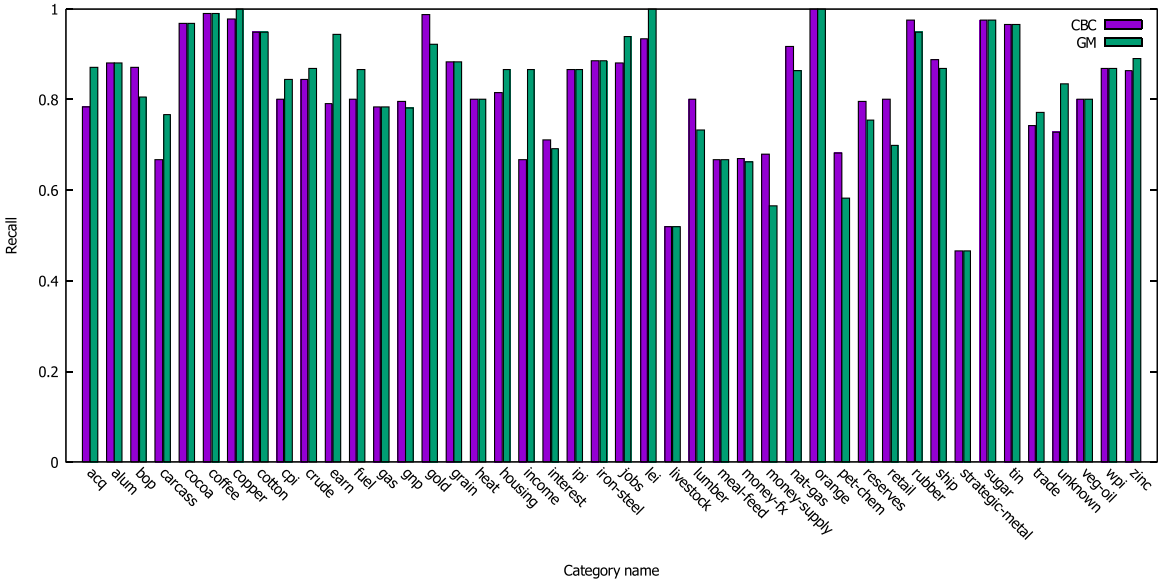

Fig. 5. The recall of each category using CBC and GM on *Reuters*.

category in *Reuters*. It can be seen that the instance distributions of categories are quite uneven, thereby leading to different geometry distributions of categories in vector space. Figs. 4 and 5 show the precision and recall of each category using GM and CBC on *Reuters*, in which the precision and recall are the average results in five-fold cross-validation scheme. With respect to CBC, the precisions of rare categories are generally lower than that of abundant categories, while the recalls of abundant categories are generally lower than that of rare categories. The low precisions of rare categories and low recalls of abundant categories largely contribute to the poor performance of CBC. However, regarding GM, as shown in Fig. 4, the precisions of rare categories, such as "*gnp*", "*income*", "*lumber*", "*money-supply*" and "*reserves*", get a significant improvement comparing with CBC. Similarly, as illustrated in Fig. 5, the recalls of abundant categories, such as "*acq*", "*earn*" and "*unknown*", also increase obviously. As a result, GM improves the precisions of rare categories and the recalls of abundant categories.

The distance between each sample and its corresponding class centroid is also calculated by Formula (23) to give a deep insight on *Reuters*. Thus, for each category, the mean, standard variance and maximum value of the distances are shown in Fig. 6. Apparently, the mean, standard variance, and maximum value fluctuate significantly between different categories. Therefore, the conclusion can be drawn is that the instance densities for different categories are rather unequal. In fact, the maximum distance of sample to its corresponding class centroid can be seen as the class radius which indicates the distribution range of each category in the vector space. Thus, from Figs. 3 and 6, it can be found out that the more instances a category contains, the larger its radius is.

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020

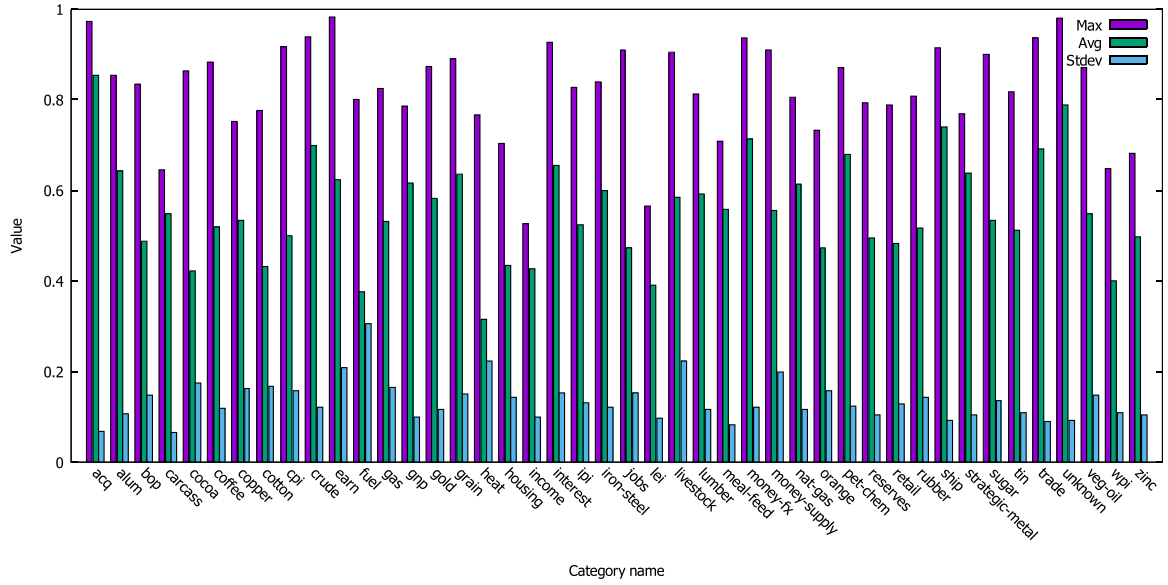

Fig. 6. The mean, standard variance and maximum value of the distances for each category on *Reuters*.

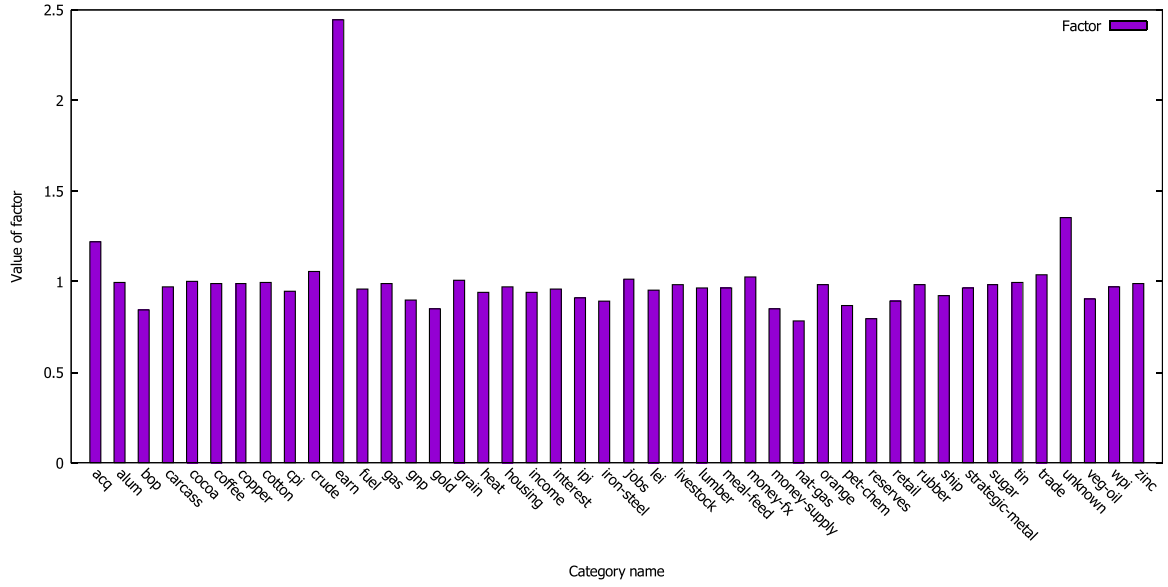

Fig. 7. The value of mass factor for each category on *Reuters*.

As the previous analysis, the category with a big radius such as "*acq*", "*earn*" and "*unknown*" may obtain low recall in CBC where the geometry distribution of each category is not considered. However, GM successfully solves this problem by giving each class a mass factor to indicate its distribution in vector space. The learned mass factor of each category on *Reuters* is shown in Fig. 7, where the top three factors are occupied by the abundant categories i.e., "*acq*", "*earn*" and "*unknown*". Apparently, the greater the mass factor of a category, the higher recall of this category. Also, the variation trend of mass factor is highly correlated with that of the radius. This phenomenon is expected since the gravitational force should be increased for the abundant category whose margin samples are far from the centroid in order to recall its samples as many as possible. Thus, most of the samples, that belong to abundant categories such as "*acq*", "*earn*" and "*unknown*" but misclassified into rare categories such as "*gnp*", "*income*", "*lumber*", "*money-supply*" and "*reserves*" in CBC model, are correctly classified in GM by increasing the mass factors of abundant categories.

7. Conclusions and future works

This paper addresses the problem of imbalanced classes in supervised document classification. The proposed model is to augment the vanilla CBC model by learning different weighted factors for different classes, thereby leading to a Voronoi partition that directly alleviates the class-imbalanced problem. A simple heuristic can be used to optimize the classification accuracy over these factors by effectively expanding or contracting the regions over which they cover. This paper reports the empirical merit of the proposed approach by comparing it with CBC and its variants across a range of datasets to show that the proposed method does improve on the base centroid method, while performing similarly to another approach (i.e., DP).

Also, there are some possible extensions of the proposed approach. For instance, as the definition of universal gravitation, the distance between objects is a significant factor for calculating the gravitational force. Thus, it is of significance to choose good initial

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020

Table 6
The comparison of different classifiers on the imbalanced datasets in $macroF_1$.

Datasets	GM	CBC	CFC-CV	DP
D1	0.902	0.848	0.748	**0.926**
D2	0.909	0.890	0.741	**0.917**
D3	0.803	0.791	0.688	**0.823**
D4	0.922	0.890	0.796	**0.942**
D5	**0.854**	0.844	0.827	0.824
D6	0.947	0.924	0.832	**0.955**
D7	0.883	0.852	0.801	**0.904**
D8	0.868	0.857	0.753	**0.869**
D9	0.830	**0.831**	0.719	0.801
D10	0.823	0.824	0.706	**0.867**
D11	**0.838**	0.819	0.643	0.804
D12	0.828	0.821	0.692	**0.854**
D13	0.775	**0.785**	0.607	0.746
D14	**0.796**	0.774	0.648	0.773
D15	0.855	0.824	0.720	**0.864**
D16	0.924	0.922	0.757	**0.931**
D17	**0.777**	0.750	0.626	0.717
D18	0.837	0.802	0.690	**0.877**
D19	**0.858**	0.825	0.654	0.852
D20	0.897	0.873	0.735	**0.912**

centroid vectors before learning the mass factors in GM. Therefore, a simple extension of GM comes from combining GM with the existing variants of CBC to achieve higher accuracy. Also, some other mass models could be explored, e.g., using the number of samples in a category as the mass of a class.

Acknowledgments

The authors would like to thank Yehao Deng for his suggestions in the revision of this paper and also thank the anonymous referees for the valuable comments and suggestions which help us to improve this paper. This work has been supported by MoE-CMCC (Ministry of Education of China - China Mobile Communications Corporation) Joint Science Fund under grant MCM20130661.

References

[1] H. Takçı, T. Güngör, A high performance Centroid-Based Classification approach for language identification, Pattern Recognit. Lett. 33 (16) (2012) 2077–2084.
[2] J. Xuan, X. Luo, G. Zhang, J. Lu, Z. Xu, Uncertainty analysis for the keyword system of web events, IEEE Trans. Syst., Man, Cybern. 46 (6) (2016) 829–842.
[3] H. Guan, J. Zhou, M. Guo, A Class-Feature-Centroid classifier for text categorization, in: Proceedings of the 18th International Conference on World Wide Web, ACM, 2009, pp. 201–210.
[4] W. Dandan, C. Qingcai, W. Xiaolong, A framework of Centroid-Based methods for text categorization, IEICE Trans. Inf. Syst. 97 (2) (2014) 245–254.
[5] V. Lertnattee, T. Theeramunkong, Class normalization in centroid-based text categorization, Inf. Sci. (Ny) 176 (12) (2006) 1712–1738.
[6] S. Tan, An improved centroid classifier for text categorization, Expert Syst. Appl. 35 (1) (2008) 279–285.
[7] S. Tan, Large margin dragpushing strategy for centroid text categorization, Expert Syst. Appl. 33 (1) (2007) 215–220.
[8] E.-H.S. Han, G. Karypis, Centroid-Based document classification: analysis and experimental results, in: European Conference on Principles of Data Mining and Knowledge Discovery, Springer, 2000, pp. 424–431.
[9] Y. Liu, Y. Yang, J. Carbonell, Boosting to correct inductive bias in text classification, in: Proceedings of the Eleventh International Conference on Information and Knowledge Management, ACM, 2002, pp. 348–355.
[10] S. Chakrabarti, S. Roy, M.V. Soundalgekar, Fast and accurate text classification via multiple linear discriminant projections, VLDB J. 12 (2) (2003) 170–185.
[11] D. Wang, J. Wu, H. Zhang, K. Xu, M. Lin, Towards enhancing centroid classifier for text classification border-instance approach, Neurocomputing 101 (2013) 299–308.
[12] G. Pang, S. Jiang, A generalized cluster centroid based classifier for text categorization, Inf. Process Manag. 49 (2) (2013) 576–586.
[13] F. Aurenhammer, Voronoi diagrams survey of a fundamental geometric data structure, ACM Comput. Surv. (CSUR) 23 (3) (1991) 345–405.
[14] A.M. Kibriya, E. Frank, B. Pfahringer, G. Holmes, Multinomial Naive Bayes for text categorization revisited, in: AI 2004: Advances in Artificial Intelligence, Springer, 2004, pp. 488–499.
[15] G. Guo, H. Wang, D. Bell, Y. Bi, K. Greer, Using knn model for automatic text categorization, Soft. Comput. 10 (5) (2006) 423–430.
[16] Z. Yu, H. Chen, J. Liu, J. You, H. Leung, G. Han, Hybrid-nearest neighbor classifier, IEEE Trans. Cybern. 46 (6) (2016) 1263–1275.
[17] R. Chau, C. Yeh, K.A. Smith, A neural network model for hierarchical multilingual text categorization, in: Advances in Neural Networks–ISNN 2005, Springer, 2005, pp. 238–245.
[18] B. Agarwal, N. Mittal, Text classification using machine learning methods-a survey, in: Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS 2012), December 28–30, 2012, Springer, 2014, pp. 701–709.
[19] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, **, 2013 arXiv Preprint arXiv:1301.3781.
[20] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
[21] T. Mikolov, W.-t. Yih, G. Zweig, Linguistic regularities in continuous space word representations., in: Hlt-naacl, vol.13, 2013, pp. 746–751.
[22] S.L. Lam, D.L. Lee, Feature reduction for neural network based text categorization, in: Database Systems for Advanced Applications, 1999. Proceedings., 6th International Conference on, IEEE, 1999, pp. 195–202.
[23] R.-C. Chen, C.-H. Hsieh, Web page classification based on a support vector machine using a weighted vote schema, Expert Syst. Appl. 31 (2) (2006) 427–435.
[24] B.-F. Zhang, J.-S. Su, X. Xu, A class-incremental learning method for multi-class support vector machines in text classification, in: Machine Learning and Cybernetics, 2006 International Conference on, IEEE, 2006, pp. 2581–2585.
[25] Z. Cataltepe, E. Aygun, An improvement of centroid-based classification algorithm for text classification, in: Data Engineering Workshop, 2007 IEEE 23rd International Conference on, IEEE, 2007, pp. 952–956.
[26] C. Liu, W. Wang, Q. Zhao, X. Shen, M. Konan, A new feature selection method based on a validity index of feature subset, Pattern Recognit. Lett. 92 (2017) 1–8.
[27] F. Colace, M. De Santo, L. Greco, P. Napoletano, Text classification using a few labeled examples, Comput. Human Behav. 30 (2014) 689–697.
[28] Z. Yu, L. Li, J. Liu, G. Han, Hybrid adaptive classifier ensemble, IEEE Trans. Cybern. 45 (2) (2015) 177–190.
[29] S. Garcia, J. Derrac, J. Cano, F. Herrera, Prototype selection for nearest neighbor classification: taxonomy and empirical study, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 417–435.
[30] Q.V. Le, T. Mikolov, Distributed representations of sentences and documents., in: ICML, vol.14, 2014, pp. 1188–1196.
[31] C. Liu, W. Wang, M. Wang, F. Lv, M. Konan, An efficient instance selection algorithm to reconstruct training set for support vector machine, Knowl. Based Syst. 1 (1) (2016) 1–24.
[32] X. Yang, J. Lu, G. Zhang, Adaptive pruning algorithm for least squares support vector machine classifier, Soft. Comput. 14 (7) (2010) 667–680.
[33] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[34] Y. Freund, R.E. Schapire, A desicion-theoretic generalization of on-line learning and an application to boosting, in: European Conference on Computational Learning Theory, Springer, 1995, pp. 23–37.
[35] T.K. Ho, The random subspace method for constructing decision forests, IEEE Trans. Pattern Anal. Mach. Intell. 20 (8) (1998) 832–844.
[36] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[37] V. García, J.S. Sánchez, R.A. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowl. Based Syst. 25 (1) (2012) 13–21.
[38] A. Orriols-Puig, E. Bernadó-Mansilla, Evolutionary rule-based systems for imbalanced data sets, Soft Comput.-A Fusion Foundations, Methodologies Appl. 13 (3) (2009) 213–225.
[39] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics, Inf. Sci. (Ny) 250 (2013) 113–141.
[40] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[41] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM Sigkdd Explorations Newsletter 6 (1) (2004) 20–29.
[42] P. Domingos, Metacost: a general method for making classifiers cost-sensitive, in: Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 1999, pp. 155–164.
[43] B. Zadrozny, J. Langford, N. Abe, Cost-sensitive learning by cost-proportionate example weighting, in: Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, IEEE, 2003, pp. 435–442.
[44] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Napolitano, Rusboost: a hybrid approach to alleviating class imbalance, IEEE Trans. Syst., Man, Cybern.-Part A 40 (1) (2010) 185–197.
[45] S. Wang, X. Yao, Relationships between diversity of classification ensembles and single-class performance measures, IEEE Trans. Knowl. Data Eng. 25 (1) (2013) 206–219.
[46] L.I. Kuncheva, J.J. Rodríguez, A weighted voting framework for classifiers ensembles, Knowl. Inf. Syst. 38 (2) (2014) 259–275.
[47] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Trans. Syst. Man. Cybern. 2 (3) (1972) 408–421.
[48] M. Kubat, S. Matwin, et al., Addressing the curse of imbalanced training sets: one-sided selection, in: ICML, vol.97, Nashville, USA, 1997, pp. 179–186.
[49] P. Hart, The condensed nearest neighbor rule, IEEE Trans. Inf. Theory 14 (3) (1968) 515–516.

[50] J. Laurikkala, Improving identification of difficult small classes by balancing class distribution, in: Conference on Artificial Intelligence in Medicine in Europe, Springer, 2001, pp. 63–66.
[51] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-smote: a new over-sampling method in imbalanced data sets learning, Adv. Intell. Comput. 3644 (5) (2005) 878–887.
[52] H. He, Y. Bai, E.A. Garcia, S. Li, Adasyn: adaptive synthetic sampling approach for imbalanced learning, in: Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, IEEE, 2008, pp. 1322–1328.
[53] C. Bunkhumpornpat, K. Sinapiromsaran, C. Lursinsap, Safe-level-smote: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem, Adv. Knowl. Discov Data Mining 5476 (2009) 475–482.
[54] J. Stefanowski, S. Wilk, Selective pre-processing of imbalanced data for improving classification performance, in: International Conference on Data Warehousing and Knowledge Discovery, Springer, 2008, pp. 283–292.
[55] K.M. Ting, An instance-weighting method to induce cost-sensitive trees, IEEE Trans. Knowl. Data Eng. 14 (3) (2002) 659–665.
[56] B. Zadrozny, C. Elkan, Learning and making decisions when costs and probabilities are both unknown, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2001, pp. 204–213.
[57] Y. Sun, M.S. Kamel, A.K. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recognit. 40 (12) (2007) 3358–3378.
[58] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst., Man, Cybern., Part B (Cybern.) 39 (2) (2009) 539–550.
[59] J. Van Hulse, T.M. Khoshgoftaar, A. Napolitano, An empirical comparison of repetitive undersampling techniques, in: Information Reuse & Integration, 2009. IRI'09. IEEE International Conference on, IEEE, 2009, pp. 29–34.
[60] W. Fan, S.J. Stolfo, J. Zhang, P.K. Chan, Adacost: misclassification cost-sensitive boosting, in: Icml, 1999, pp. 97–105.
[61] K.M. Ting, A comparative study of cost-sensitive boosting algorithms, in: In Proceedings of the 17th International Conference on Machine Learning, Citeseer, 2000.
[62] M.V. Joshi, V. Kumar, R.C. Agarwal, Evaluating boosting algorithms to classify rare classes: comparison and improvements, in: Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on, IEEE, 2001, pp. 257–264.
[63] S. Hu, Y. Liang, L. Ma, Y. He, Msmote: improving classification performance when training data is imbalanced, in: Computer Science and Engineering, 2009. WCSE'09. Second International Workshop on, vol.2, IEEE, 2009, pp. 13–17.
[64] S. Wang, X. Yao, Diversity analysis on imbalanced data sets by using ensemble models, in: Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on, IEEE, 2009, pp. 324–331.
[65] S. Hido, H. Kashima, Y. Takahashi, Roughly balanced bagging for imbalanced data, Stat. Anal. Data Mining 2 (5–6) (2009) 412–426.
[66] J. Błaszczyński, M. Deckert, J. Stefanowski, S. Wilk, Integrating selective pre-processing of imbalanced data with ivotes ensemble, in: International Conference on Rough Sets and Current Trends in Computing, Springer, 2010, pp. 148–157.
[67] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Trans. Syst., Man, Cybern., Part C (Appl. Rev.) 42 (4) (2012) 463–484.
[68] E. Leopold, J. Kindermann, Text categorization with support vector machines. how to represent texts in input space? Mach. Learn. 46 (1–3) (2002) 423–444.
[69] M. Lan, C.L. Tan, J. Su, Y. Lu, Supervised and traditional term weighting methods for automatic text categorization, Pattern Anal. Mach. Intell., IEEE Trans. 31 (4) (2009) 721–735.
[70] T.T. Nguyen, K. Chang, S.C. Hui, Supervised term weighting centroid-based classifiers for text categorization, Knowl. Inf. Syst. 35 (1) (2013) 61–85.
[71] V. Lertnattee, C. Leuviphan, et al., Using class frequency for improving centroid-based text classification, ACEEE Int. J. Inf. Technol. 2 (02) (2012) 62–66.
[72] S. Robertson, Understanding inverse document frequency: on theoretical arguments for idf, J. Doc. 60 (5) (2004) 503–520.
[73] M. Lan, S.-Y. Sung, H.-B. Low, C.-L. Tan, A comparative study on term weighting schemes for text categorization, in: Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on, vol.1, 2005, pp. 546–551.
[74] V. Lertnattee, T. Theeramunkong, Analysis of inverse class frequency in centroid-based text classification, in: Communications and Information Technology, 2004. ISCIT 2004. IEEE International Symposium on, vol.2, IEEE, 2004, pp. 1171–1176.
[75] Y.-Q. Miao, M. Kamel, Pairwise optimized rocchio algorithm for text categorization, Pattern Recognit Lett 32 (2) (2011) 375–382.
[76] H. Wu, T.H. Phang, B. Liu, X. Li, A refinement approach to handling model misfit in text categorization, in: Proceedings of the eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2002, pp. 207–216.
[77] S. Tan, Using error-correcting output codes with model-refinement to boost centroid text classifier, in: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, Association for Computational Linguistics, 2007, pp. 81–84.
[78] G. Salton, A. Wong, C.-S. Yang, A vector space model for automatic indexing, Commun. ACM 18 (11) (1975) 613–620.
[79] K. Crammer, R. Gilad-Bachrach, A. Navot, N. Tishby, Margin analysis of the lvq algorithm, in: Advances in Neural Information Processing Systems, 2002, pp. 462–469.
[80] C. Liu, W. Wang, M. Konan, S. Wang, L. Huang, Y. Tang, X. Zhang, A new validity index of feature subset for evaluating the dimensionality reduction algorithms, Knowl. Based Syst. 121 (2017) 83–98.
[81] Y. Zhao, G. Karypis, D.-Z. Du, Criterion functions for document clustering, Technical Report, 2005. Technical Report
[82] M.F. Porter, An algorithm for suffix stripping, Program 14 (3) (1980) 130–137.
[83] Z. Yu, J. You, L. Li, H.-S. Wong, G. Han, Representative distance: a new similarity measure for class discovery from gene expression data, IEEE Trans. Nanobiosci. 11 (4) (2012) 341–351.

Please cite this article as: C. Liu et al., A new Centroid-Based Classification model for text categorization, Knowledge-Based Systems (2017), http://dx.doi.org/10.1016/j.knosys.2017.08.020