EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

LDA boost classification: boosting by topics

La Lei*, Guo Qiao, Cao Qimin and Li Qitao

Abstract

AdaBoost is an efficacious classification algorithm especially in text categorization (TC) tasks. The methodology of setting up a classifier committee and voting on the documents for classification can achieve high categorization precision. However, traditional Vector Space Model can easily lead to *the curse of dimensionality* and *feature sparsity* problems; so it affects classification performance seriously. This article proposed a novel classification algorithm called LDABoost based on boosting ideology which uses Latent Dirichlet Allocation (LDA) to modeling the feature space. Instead of using words or phrase, LDABoost use latent topics as the features. In this way, the feature dimension is significantly reduced. Improved Naïve Bayes (NB) is designed as the weaker classifier which keeps the efficiency advantage of classic NB algorithm and has higher precision. Moreover, a two-stage iterative weighted method called Cute Integration in this article is proposed for improving the accuracy by integrating weak classifiers into strong classifier in a more rational way. Mutual Information is used as metrics of weights allocation. The voting information and the categorization decision made by basis classifiers are fully utilized for generating the strong classifier. Experimental results reveals LDABoost making categorization in a low-dimensional space, it has higher accuracy than traditional AdaBoost algorithms and many other classic classification algorithms. Moreover, its runtime consumption is lower than different versions of AdaBoost, TC algorithms based on support vector machine and Neural Networks.

Keywords: Latent Dirichlet Allocation, Topics, Boosting, Two-procedure iterative weighting, Text classification

1. Introduction

Text categorization (TC) has received unprecedented focus in recent years. A TC system can rescue people from tremendous amount of information in this era of information explosion. In addition, text classification is the foundation of many popular information processing technologies such as information retrieval, machine Q & A and sentiment analysis. Since a high percentage of information in the network is textual information [1], the precision of text classification will largely determines the ability of people in information utilization, in other words, the quality of our life.

The procedure of TC can be defined similar with other data classification tasks as the problem of approximating an unknown category assignment function $F{:}D \times C \rightarrow \{0, 1\}$, where D is a set of documents and C is the set of predefined categories:

$$F(d,c) = \begin{cases} 1, & d \in D \ \& \ d \ \textit{belong to the class} \ c \\ 0, & \textit{otherwise} \end{cases} \quad (1)$$

The approximating function $M{:}D \times C \rightarrow \{0, 1\}$ is called a classifier. The task is to build a classifier that produces results as close as possible to the true category assignment function F.

The first step of TC is feature selection. Feature selection is a process of choosing representative features such as words, phrases, concepts, etc., as the classification operand. Note that the most frequent words are not always the feature words. For instance, *corpus* is a very important word in a scientific literature retrieval system, but it would not be chosen in a corpus database system. An example of feature selection in a sports news classification system is shown in Figure 1.

Since feature selection is the basis of TC, it has aroused extensive attention from scholars. Feature representation models such as Bag-of-words, Vector Space Model (VSM), Probabilistic Latent Semantic Indexing [2], and Latent Dirichlet Allocation (LDA) [3] have been proposed for selecting features in document set.

In traditional Bag-of-words and VSM, words are selected as features. Word features tend to result in the curse of dimensionality and feature sparsity problems. Feature dimension of a middle-size document set may

* Correspondence: lalei1984@yahoo.com.cn
School of Automation, Beijing Institute of Technology, Beijing, China

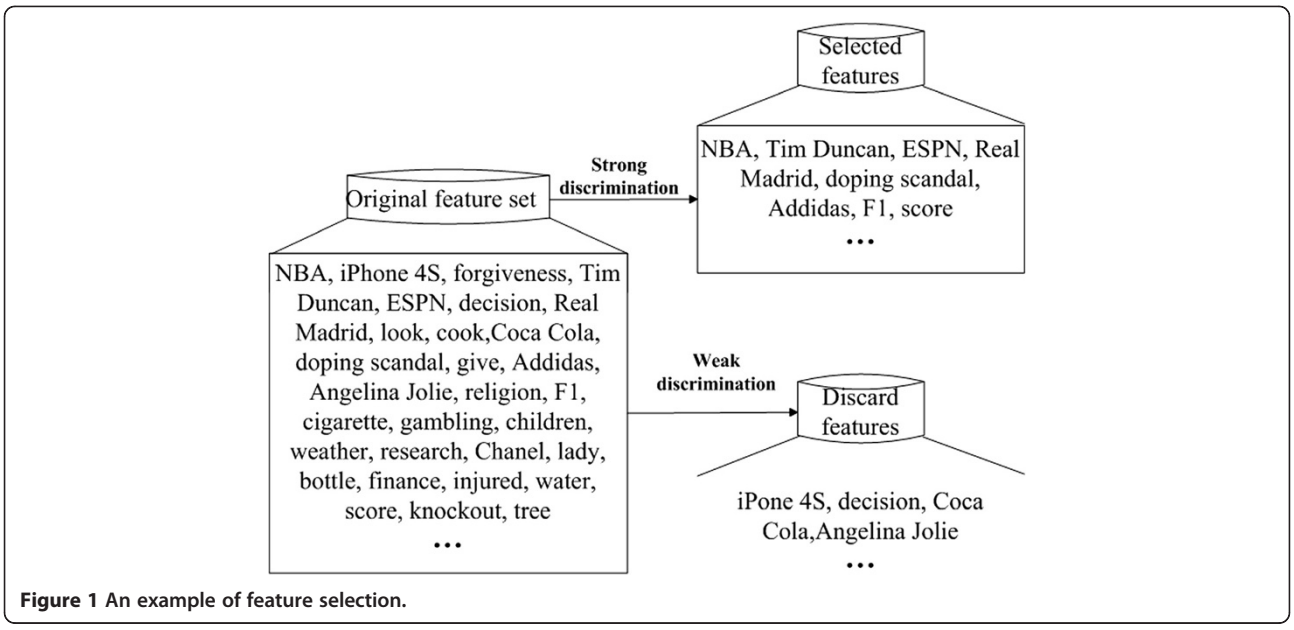

Figure 1 An example of feature selection.

reach 10^4 or 10^5 [4] and extremely increasing the computational and runtime complexity of the task. This is the so-called curse of dimensionality. Feature sparsity means the occurrence probability in a certain document of a feature which belonged to the document set is very low. In other words, in the vector space, most components of a text are zero-vectors. Feature sparsity would greatly reduce the accuracy of classification [5]. To solve problems above, some experts try to use non-continuous phrases [6], concepts [7], and topics [8] as features.

Another pivotal aspect of TC is a classification algorithm design. Although there are also considerable literatures in this area, support vector machine (SVM), Decision Tree, Neural Networks, Naïve Bayes (NB), Rocchio, and voting-based algorithm [9] are the most important methods. The core issue of categorization is kept balance between accuracy and efficiency. Some algorithms have quite good accuracy and high time cost at the same time, such as SVM. Light classification algorithm, for instance, NB, has low time consumption but the precision is not always ideal. Even more, neural networks and some other compromise solutions may lead to bad performance both in accuracy and efficiency. Voting-based categorization algorithms also known as classifier committees can adjust the number and professional level of "experts" in the committees to find a balance between performance and time-computational consumption.

Few researchers place dimension reduction and classification algorithm in the same framework to make a comprehensive consideration. Classification algorithm should be based on feature selection to further improving its performance. In another hand, feature dimension reduction should use classification algorithm to check its effectiveness.

The rest of this article is organized as follows. Section 2 reviews LDA and analyzes its application in text feature selection. Section 3 improves traditional NB as the weak classifier. In Section 4, a two-procedure iterative weighted method is proposed by introducing Mutual Information (MI) criterion in it to integrating a strong classifier. Section 5 then proposed LDABoost based on Sections 3 and 4 which is the first time that LDA is used together with Boosting algorithm to the best of the authors' knowledge as the final classification framework. The application of the novel classification method is presented and analyzed in Section 6. Finally, Section 7 summarizes the article.

2. Feature extraction by LDA

Strictly speaking, dimension reduction algorithms can be categorized into two groups: feature extraction and feature selection. In the former, new features of texts are combined from their original features through algebraic transformation. In the latter, subsets of features are selected directly. Feature extraction is mathematically efficient but with high computational overhead [10]. Feature selection is quite convenient to be implemented in real world. However, there is no theoretically guarantee in optimality for feature selection's solution. Probabilistic topic model-based dimension reduction algorithms attract more and more attention because it maintains the merit of feature extraction and to some extent overcome the high computational consumption problems.

2.1. LDA

LDA is a powerful probabilistic topic model. Its essence is a three-layer Bayesian network. It uses a structure more or less like the following former: category > latent topics > words. The schematic of LDA is shown in Figure 2 [11].

In Figure 2, K is the number of topics, M the number of documents, N_m the number of words in the mth document, ϕ_k the words distribution in topic k, θ_m the topic distribution in document m, ϕ_k and θ_m also the parameters of multinomial distribution which are used to *generating* topics and words, α and β are empirical parameters and usually they are symmetric.

ϕ_k and θ_m follow a Dirichlet allocation as

$$P_{Dir}(\mu|\alpha) = \frac{\Gamma(\alpha_0)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\prod_{k=1}^{K}\mu_k^{\alpha_k-1} \quad (2)$$

where $0 \le \mu_k \le 1$, $\sum_k \mu_k = 1$, $\alpha_0 = \sum_{k=1}^{k} \alpha_k$ and Γ is the Gamma function. Dirichlet distribution is the priori conjugate distribution of multinomial distribution.

LDA follows below steps to *generating* words [12]:

1. Topic sampling by $\varphi_k \sim P_{\mathrm{Dir}}(\beta)$, $k \in [1, K]$.
2. In document m, $m \in [1, M]$ make topic probability distribution sampling by $\theta_m \sim P_{\mathrm{Dir}}(\alpha)$.
3. Document length sampling by $N_m \sim$ Poisson (ξ).
4. Select a latent topic $z_{m,n} \sim$ Multinomial (θ) for nth word in document m, where $n \in [1, N]$.
5. Generate a word $w_{m,n} \sim$ Multinomial($\phi_{z_{m,n}}$).

In LDA, we assume that words are generated by topics and those topics are infinitely exchangeable within a document. Therefore, the joint probability of topics and words is

$$P(w, z) = \int P(\theta)(\prod_{n=1}^{N} P(z_n|\theta)P(w_n|z_n)d\theta \quad (3)$$

Follow above steps, LDA model aggregates semantically similar words as latent topic. If we make topic selection according to function (2) using them as text features, the feature dimension will greatly be reduced.

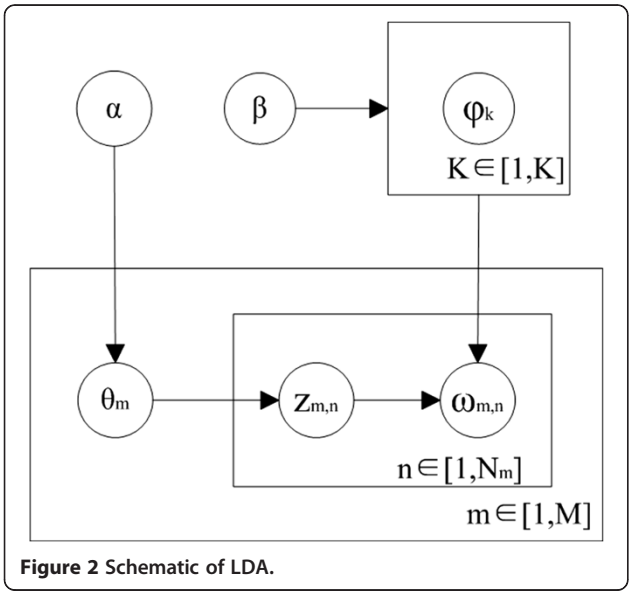

Figure 2 Schematic of LDA.

2.2. Parameter estimation in LDA

Obviously, neither Equation (1) nor Equation (2) can be calculated directly. Therefore, the topic selection problem translated into parameter estimation problem. In LDA, parameters can be estimated by Maximum Entropy, Variational Bayesian Inference [13], Expectation-Propagation [14], Gibbs sampling, etc.

Gibbs sampling is a special case of Markov Chain Monte Carlo, it samples for a component of the joint distribution and keep the value of other components in a time. For the situation of high-dimensional joint distribution, this strategy simplified steps of the algorithm.

Heinrich [15] designed a Collapsed Gibbs Sampling (CGS) algorithm to avoid the estimation of parameters ϕ_k and θ_m by using integration. CGS samples topic z for each word w. Once the topic of w is identified, ϕ_k and θ_m can be calculated by frequency statistics. As the analysis above, parameter estimation problem translate into calculate the conditional probability of topic sequence in the condition that word sequence is known as

$$P(z|w) = \frac{P(w, z)}{\sum_z (w, z)} \quad (4)$$

Where w is a vector constitute by the words end-to-end. Because the sequence of z is usually very long, the possible value growth exponentially with the length of the vector and difficult to be calculated directly. Fortunately, CGS can decompose the problem into several sub-problems, samples a topic in each time. The final sampling function is

$$P(z_i = k|z_{\to i}, w) \propto \frac{n_{k,\to i}^{t} + \beta_t}{\sum_{t=1}^{V} n_{k,\to i}^{(t)} + \beta_t} \cdot \frac{n_{m,\to i}^{(t)} + \alpha_k}{\left[\sum_{k=1}^{K} n_m^{(k)} + \alpha_k\right] - 1} \quad (5)$$

Assume $w_i = t$, where z_i represents the topic variation of ith word, $\to i$ means exclude element i, n_k^t is the occurrence time of word t in topic k, βt is the priori of Dirichlet distribution, $n_{m(k)}$is the frequency of topic k in document m, αk is the Dirichlet priori of topic k.

Since we get the topic k of word w, parameters ϕ_k and θ_m can be computed as:

$$\phi_{k,i} = \frac{n_k^{(t)} + \beta_t}{\sum_k^K n_k^{(t)} + \beta_t} \tag{6}$$

$$\theta_{m,k} = \frac{n_m^{(t)} + \alpha_k}{\sum_k^K n_m^{(t)} + \alpha_k} \tag{7}$$

LDA builds a statistic model for document set, texts, categories, topics, and words. Using sampling algorithms such as Gibbs sampling can estimate the model's parameters and achieve document representation in feature space.

2.3. Dimension reduction based on LDA

Reasonable feature selection and feature extraction approaches should make documents of the same category have much shorter distance in feature space and documents from different categories have much longer distance. In other words, categorization results based on selected features should have maximum within-class similarity and minimum between-classes similarity.

Feature distance can be measured in different space systems such as Euclidean distance, Manhattan distance, Minkowski distance, Chebyshev distance, and so on. Euclidean distance is probably the most popular distance metrics. However, in classification problems especially in TC problem, Mahalanobis distance is the most effective ranging standard [16]. The definition of Mahalanobis distance as follows

$$D_M(x) = \sqrt{(x-\mu)^T \sum^{-1} (x-\mu)} \tag{8}$$

Where $x = (x_1, x_2, \ldots, x_n)^T$ is a multi-variable feature vector, the mean of x is $\mu = (\mu_1, \mu_2, \ldots, \mu_n)^T$. Different from Euclidean distance, Mahalanobis distance can reflect the relationship between various of the feature. In addition, it takes features' characteristics of scale-invariant into account. Therefore, Mahalanobis distance is used to measure the distance of topics and as the reference of classification.

Use topic as feature will undoubtedly increase the distance of features and reduce the between-classes similarity of texts. The principle of topic features is shown in Figure 3.

As show in the figure, LDA can decrease the probability of misclassification caused by confusing words. Furthermore, science plenty of words converging into a topic, LDA significantly reduces the dimensionality of feature space. Topics in feature space are quite similar with cluster headers in ad hoc networks. In ad hoc networks, using cluster headers as representation of the web can greatly deduces the complexity of network topology. Similarly, use topics to representing documents can benefit categorization.

The workflow of dimension reduction based on LDA is as follows:

1. Input training document set.
2. Preprocessing. Such as word segmentation and Part-of-Speech tagging.
3. Preprocessing. Check the stop words list and remove them out of the document set.
4. Set values for empirical parameters.
5. Call LDA. Synthesize words into latent topics.
6. Calculate Mahalanobis distance of topics and select high weight topics as the feature topics.

Hitherto, a document feature extraction method is proposed. It based on LDA model and can significantly reduce the dimension of feature space by selecting topics as document features. Using the low-dimensional feature set as the foundation can greatly improve the accuracy of TC, moreover, decrease its time and computational consumption.

3. Classifier design based on NB

Theoretically, once weak classifiers are more accurate than guess randomly (1/2 in two-class tasks or $1/n$ in multi-class tasks), AdaBoost can integrate them into a strong classifier whose precision could infinitely close to the true category distribution [17]. However, when the precision of weak classifiers is lower, more weak classifiers are needed to construct a strong classifier. Too many weak classifiers in the system sometimes increase its complexity and computational consumption to intolerable level. In another hand, boosting algorithms which use complex base learners based on SVM [18], Neural Networks [19], etc., can certainly achieve higher accuracy but lead to some new problems because they are over sophisticated and thus contrary to the ideology of Boosting algorithm.

Boosting algorithm proposed in this article uses topics supported by LDA as its feature set. According to the analysis in Section 2, topic feature set has parlous lower dimension and features in it have higher discrimination. Therefore, weak classifier based on simple algorithm such as NB can achieve an ideal precision with really low runtime cost.

3.1. NB classification

The basic idea of NB is calculates the priori probability of an object, then using Bayesian formula to calculate its posterior probability. Finally, use the posterior probability as the probability of which category the new text should belong to.

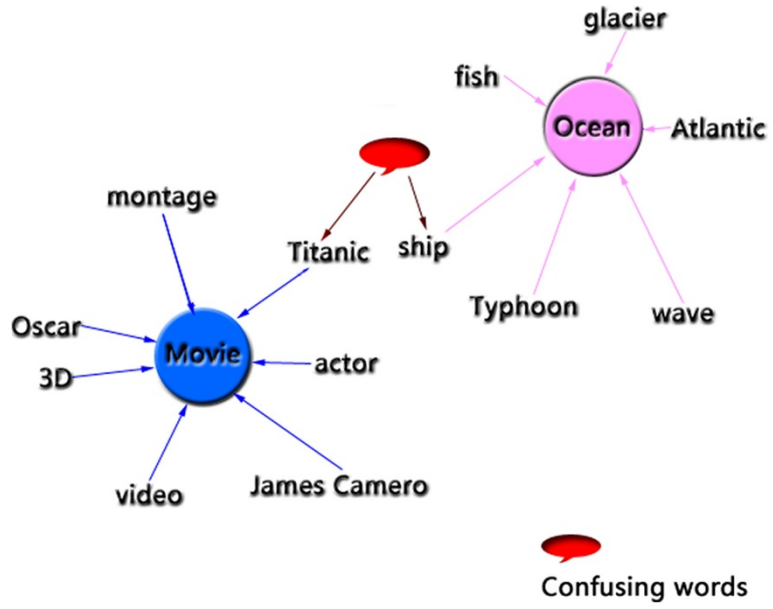

Figure 3 The principle of topic features.

In the training document set, priori probability vector $X = (x_1, x_2, \ldots, x_n)$ of weather topic features belong to some class can be calculated as:

$$x_k = P(z_k|c_j) = \frac{\sum_{l=1}^{D} N(z_k, d_l)}{|V| + \sum_{s=1}^{V}\sum_{l=1}^{D} N(z_s, d_l)} \tag{9}$$

Where $N(z_k, d_l)$ is the frequency of kth topic in the lth document. $|V|$ is the sum of topics, c_j the jth category, and D the sum of documents which belong to it.

In the test document set, the solution function of posterior probability is:

$$P(c_j|d_l) = \frac{P(c_j)\prod_{k=1}^{n} P(z_k|c_j)}{\sum_{r=1}^{C} P(c_r)\prod_{k=1}^{n} P(z_k|c_r)} \tag{10}$$

$P(c_j)$ can be calculated as:

$$P(c_j) = \frac{traing\ test\ belong\ to\ category\ c_j}{sum\ of\ training\ documents} \tag{11}$$

Where C is the sum of categories, n the number of feature topics in document d_l.

The posterior $P(c_j|d_l)$ of a document in different category condition has the same denominator $\sum_{r=1}^{C} P(c_r)\prod_{k=1}^{n} P(z_k|c_r)$.

Therefore, NB TC finally calculates function below

$$P(c_j|d_l) = P(c_j)\prod_{k=1}^{n} P(z_k|c_j) \tag{12}$$

As shown in Equation (12), NB is quite a light classification method.

3.2. Multi-level NB

Features do not have weight in original NB, they are believed to have equal contribution for classification. However, this assumption is seldom suitable in TC. Latent topics from headlines, abstracts, and key words always have significant importance for TC. In addition, first and last paragraph of the document usually summarize the article and therefore may contain much more information for classification. Features selected from other parts of the document sometimes give lower benefit for categorization.

Therefore, topic features can be divided into several levels according to their position in documents. Give different weight for different level so that features from different levels can play different roles in categorization.

The number k of levels can be set by empirical values. However, empirical values need human experience and thus increase labor costs. Actually, k can be adjusted adaptively by sampling and comparing the relative entropy of features in different level. When the relative entropy of two levels is

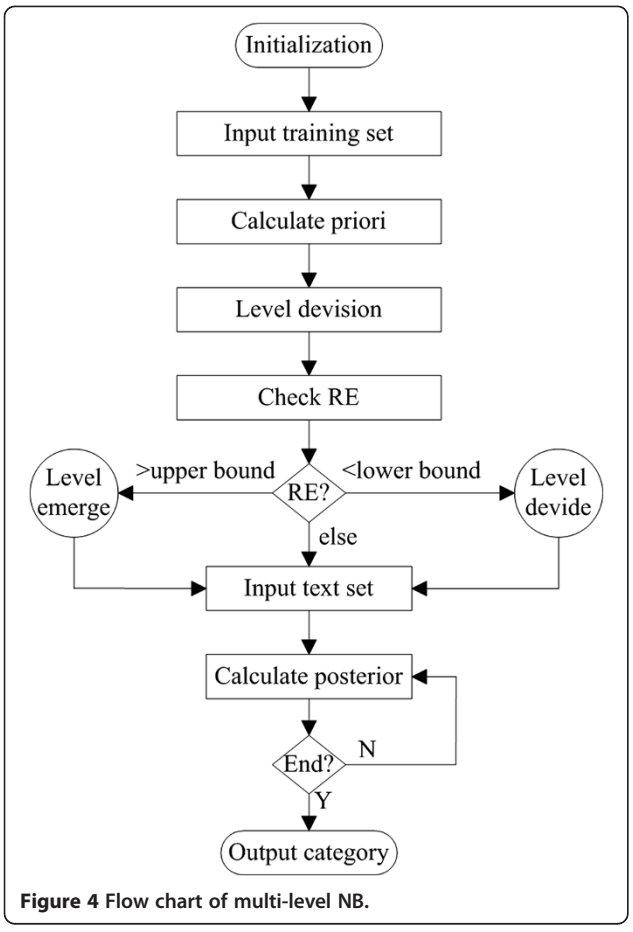

Figure 4 Flow chart of multi-level NB.

lower than system's lower bound, emerge the levels, when it is higher than upper bound, split them into more levels. The flow chart of multi-level NB is shown in Figure 4.

Following steps in Figure 4, a multi-level NB categorization algorithm is constructed. It uses topics extracted by LDA instead of feature words in traditional VSMs to improving its classification ability and maintaining the runtime consumption. Furthermore, a multi-level strategy is introduced in NB to ensure it use topics in a more effective way.

4. Cute integration (CI): the way strong classifier generated

Whether strong classifier has a good performance depends largely on how weak classifiers are combined. To build a powerful strong classifier, basis classifiers which have higher precision must take more responsibility in categorization process. Therefore, categorization system should distinguish between the performances of weak classifiers and give them different weights according to their capabilities. Moreover, ambiguous texts should be identified and pay more attention on them by allocating them higher weights. Using these weights, Boosting algorithms can integrate weak classifiers as the strong classifier in a more efficient way and achieve excellent performance.

4.1. Weighting mechanism of classic AdaBoost review

AdaBoost is a very classic boosting algorithm. It is widely used in classification problems. Reviewing its strategy is helpful for new algorithm design. Original AdaBoost algorithm uses a linear weighting way to generate strong classifier. In AdaBoost, strong classifier is defined as

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{13}$$

$$H(x) = sign(f(x)) \tag{14}$$

where $h_t(x)$ is a basis classifier, αt is a coefficient, and $H(x)$ the final strong classifier.

Given the training documents and category labels(x_1, y_1), $(x_2, y_2), \ldots, (x_m, y_m), x_i \in X$, and $y_i = \pm 1$. The strong classifier can be constructed as [20]

1. Initialize weight $D_1(i) = 1/m$, for $t = 1, 2, \ldots, T$.
2. Select a weak classifier with the smallest weighted error:

$$h_t = \arg\min_{h_j \in H} \ \varepsilon_j = \sum_{i=1}^{m} D_t(i)\left(y_i \neq h_j(x_i)\right) \tag{15}$$

   Where ε_j is the error rate.
3. Prerequisite: $\varepsilon_t < 1/2$, otherwise stop.
4. Upper bounded ε_t by $\varepsilon_t(H) \leq \prod_{t=1}^{T} Z_t$, where Z_t is a normalization factor.
5. Select α_t to greedily minimize $Z_t(\alpha)$ in each step.
6. Optimizing:

   Where $r_t = \sum_{i=1}^{m} D_t(i) h_t(x_i) y_i$ by using the constraint $Z_t = 2\sqrt{\varepsilon_t(1-\varepsilon_t)} \leq 1$.
7. Reweighting as

$$\alpha_t = \frac{1}{2}\log\left(\frac{1+r_t}{1-r_t}\right) \tag{16}$$

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t} = \frac{\exp(-y_i \sum_{q=1}^{t} \alpha_q h_q(x_i))}{{}^{m}\prod_{q=1}^{t} Z_q} \tag{17}$$

$$\exp(-\alpha_t y_i h_t(x_i)) \begin{cases} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{cases} \tag{18}$$

Above steps demonstrated that AdaBoost gives classifiers which have better classification performance higher weights automatically, especially by step 7. In this way,

AdaBoost can be implemented simply. The process of its feature selection is on a large set of features. Furthermore, it has good generalization ability. The work step of AdaBoost is shown in Figure 5.

In the above algorithm, the definition of *better classification performance* is not reasonable. Only using the classification error subset of former classifiers to training later classifiers is not enough. We called the documents which are classified incorrectly *difficult* document. The later classifiers will be evaluated whether they have the ability to rightly classifying difficult documents. However, the former classifiers have not been trained by the error subset of later classifiers.

This classifiers' training mechanism has overlooked two basic questions. First, if the document subset R_i which be classified rightly by the classifier i is also easy for classifier $i + 1$. Second, if the documents be classified incorrectly by the classifier j is also difficult for classifier $j - 1$.

The negligence of above questions makes the weights allocation strategy have no comprehensive consideration of training samples. In addition, in this situation training set could not be fully utilized to generating a more powerful strong classifier.

4.2. Two-procedure weighting method

In order to solve the above two questions, this article proposed a two-procedure weighting method. The basic idea of this weighting method takes a plus weighting step into training procedure. The additional step can be seen as the inverse process of the original iteration in AdaBoost. It uses the last document set to training the first weak classifier. It follows this order until the last base learner is trained by the first training set. Using weights in the two procedures to generating a final weight will increase the credibility of weak classifier's weight. In this way, the algorithm defines *powerful* for base classifiers by using not only the former part, but also the later part of the training sets. The work step of two-procedure weighting method is shown in Figure 6.

Two-procedure weighting algorithm can achieve weight allocation steps shown in Figure 6 following steps below.

1. Begin: initialize documents weights $w_d(i)$ and weak classifier weights $w_c(j)$.
2. Training first classifier C_1 with first sample documents subset D_1, mark the set of documents which be misclassified by C_1 in D_1 as E_1.
3. Loop: training C_i with D_i and E_{i-1}
4. Calculation: calculating weights of base classifiers according to first round of loops (trainings).
5. Reverse iterative: training c_1 with D_n.
6. Loop: training c_i with D_i and E_{n-i}.
7. Calculation: calculating weights of weak classifiers according to second round of loops (trainings).
8. Calculate final weights of base classifiers according to steps 4 and 7.
9. Cascade: combine base classifiers according to their final weights and construct strong classifier.
10. End.

Above steps ensure the full use of training sets and generate weight in each procedure.

4.3. Judgment for measuring the error

Most previous boosting-based algorithm only records the number of incorrectly classified documents. However,

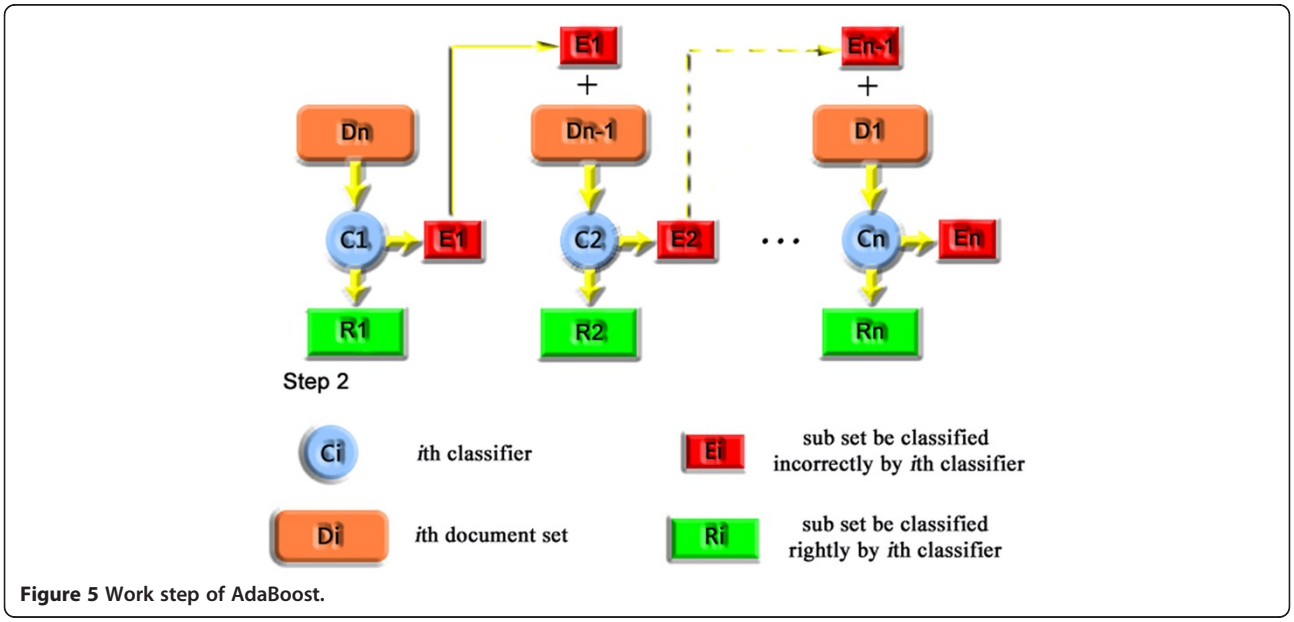

Figure 5 Work step of AdaBoost.

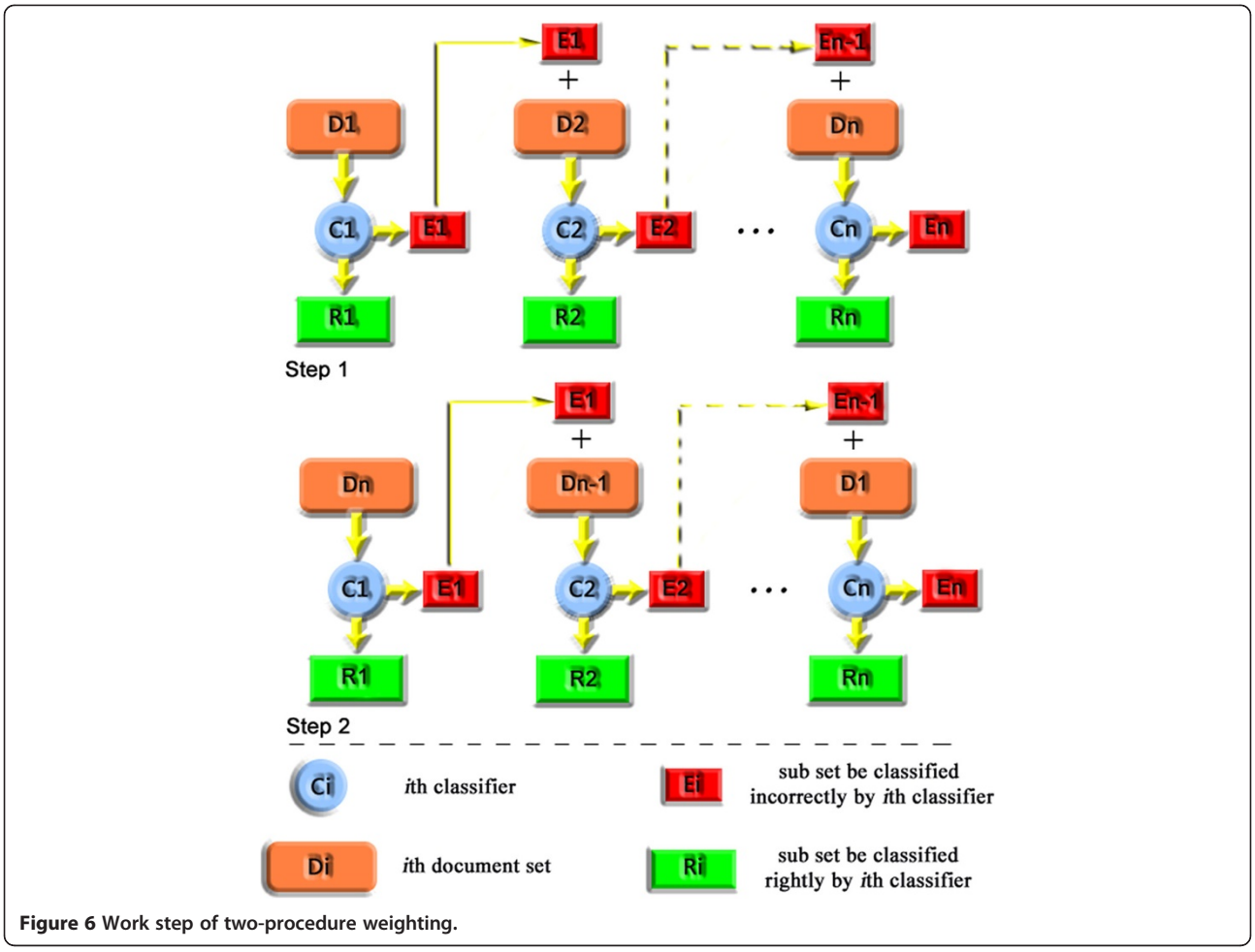

Figure 6 Work step of two-procedure weighting.

error numbers sometime cannot faithfully reflect the performance of weak classifiers because the severity of the error is not always the same.

Image the situation in Figure 3: make misclassification that put a film review about *Titanic* in the Ocean category is not as serious as put an Oscar Academy Awards in the Ocean category. In order to improve system's ability of distinguish between base classifiers' performance, some judgment should be used to evaluating the severity of errors.

Distance between the category which a document should belong to and the category which the document be classified incorrectly probably is the most intuitive reference to determine how serious an error is. However, the distance between text categories could not be measured directly like what scientist has done in physical world. In this article, we use MI as the judgment.

According to entropy theory, assume X and Y are a pair of discrete random variable where X, $Y \sim P(x, y)$, the joint entropy of X and Y defined as

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} P(x,y)\log p(x,y) \tag{19}$$

By using the chain rule of entropy, above function can be translated into:

$$\begin{aligned} H(X,Y) &= H(X) + H(Y|X) \\ &= H(Y) + H(X|Y) \end{aligned} \tag{20}$$

Therefore,

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \end{aligned} \tag{21}$$

$I(X; Y)$ is the MI of X and Y. The sketch map of MI is shown in Figure 7.

As shown in Figure 7, greater MI of two categories means they contain more similar information, thereby the distance between them is shorter. Obviously, it is less serious to misclassifying a document to a category which has large MI with its true category. Assume C_i is the true class of document i, C_i' is the error class of i. We can use $I(C_i; C_i')$ as the severity judgment of classification error.

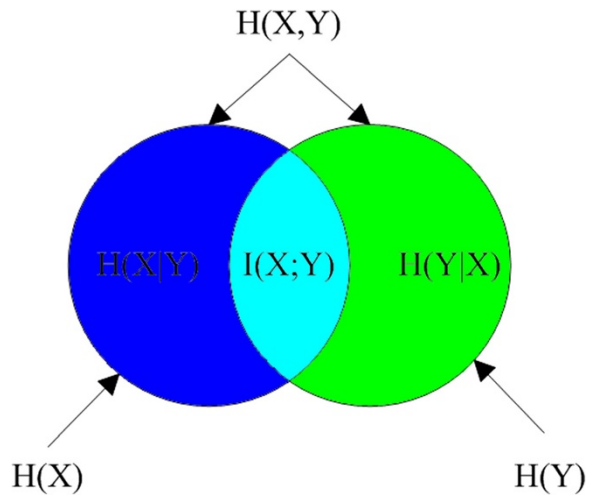

Figure 7 Sketch map of MI.

Assume $D = (d_1, d_2, \ldots, d_m)$ is the document set of category C, $D' = (d'_1, d'_2, \ldots, d'_n)$ is the document set of category C', the MI of them can be calculated as

$$\begin{aligned} I(D;D') &= H(D) - H(D|D') \\ &= H(D) + H(D') - H(D,D') \end{aligned} \tag{22}$$

Using the knowledge of entropy theory, Equation (22) can be solved as:

$$I(D,D') = \sum_{i=1}^{n}\sum_{j=1}^{m} P(d_i,d_j') \log \frac{P(d_i,d_j')}{P(d_i)P(d_j')} \tag{23}$$

If we take the error time t into account, it is easy to learn each misclassification corresponds to two categories, in other words, corresponds to a MI value. We can use the following function as the weight definition of classifier i.

$$w_i = \frac{1}{\prod_{j=1}^{t} I(C_j, C_j')} \tag{24}$$

4.4. CI algorithm: strong classifier construction

Strong classifier can be generated by integrating weak classifiers based on the strategies proposed in Sections 4.2 and 4.3. The strong classifier construction algorithm in this article called CI.

Using Equation (24) directly is the simplest but not the best way to weighting classifiers. Note that some basis classifiers may have a very high weight both in the first and second procedures. It means these classifiers have global high categorization ability and should play a more important role in classification process instead of using the average weight simply. In this case, an upper bound value is set as the final weight of significantly powerful classifiers. In another hand, some classifiers may have a very low weight in both two iterative loops. The utility of these classifiers must be limited by using a lower bound value to enhance system's accuracy.

Moreover, some weak classifiers may have a very high weight in one procedure but a very low weight in another iterative step. The system should consider the weak classifiers as noise-oversensitive and deduce its weight. In this article, we use $\min(w_j, w_j')$ as the final weight of noise-oversensitive classifier.

The runtime complexity of MI calculation is $O(m \bullet n)$ [21]. Therefore, the time consumption of CI algorithm is $O(m \bullet n^2)$, where m the number of base classifiers and n the number of training documents.

As analysis above, the computational complexity is proportional to the number of weak classifiers. In addition, when the number of classification objects increase, the time consumption would increase quadratic. Therefore, the algorithms avoid index explosion problem and have an acceptable runtime complexity. In addition, there is no condition missing and the weight's value of every classifier is non-infinite. Therefore, CI algorithm is convergence.

The pseudocode of strong classifier generation algorithm CI is shown in Figure 8.

In the figure, E_i is the error set of the ith basis classifier, w_i the weight of the ith classifier in the first weighting procedure, w_i' the weight of the ith classifier in the second weighting step, α the lower threshold of weight, w_{MIN} the lower bound, β the upper threshold of weight, w_{MAX} the upper bound, T the upper threshold of the difference between w_i and w_i', and W the final weight of the ith classifier.

Hitherto, the categorization performance of base classifiers could be measured accurately with a low time and computational overhead. The evaluation could be used for generating strong classifier in most reasonable way. Furthermore, the usage effectiveness of the training set is maximized by the CI. Theoretically, above algorithm should have better precision and higher efficiency than other boosting algorithms.

5. The final form of LDABoost

Combining works in previous sections together we can get the final framework of the novel TC system. It called LDABoost in this article.

Feature dimensionality reduction is the foundation of LDABoost. LDABoost uses LDA to modeling documents. Gibbs sampling method is used for estimating LDA's parameters and LDA uses the estimated parameters to generating topics. Most representative topics are extracted by evaluating them with Mahalanobis distance to form the feature set. Improved multi-level NB method works on the feature set as weak learns. Weak learns vote on the category which document belonged to. Document sets are input twice in different order and the weights of

Algorithm: Cute Integration (generate strong classifier)

Input: document set $D = \{D_1, D_2, \ldots, D_m\}$ which be represented by topic features
Classifier set $C = \{C_1, C_2, \ldots, C_n\}$
Output: Strong classifier **H**

```
1  begin
2    for ( i=1, i<=m, i++ )
3      for ( j=1, j<=n, J++ )
4        training Cj by Di and E(i-1)^1
5      calculate mutual information I ( j )
6    calculate weight wj
7    for ( i=m-1, i<=1, i-- )
8      for ( j=1, j<=n, j++ )
9        training Cj by Di and E(i-1)^2
10     calculate mutual information I ( j' )
11   calculate weight wj'
12   int j=1
13   while ( j <n )
14     j++
15     if ( wj>α && wj'>α)
16       Wj = wMAX
17     else if ( wj<β && wj'<β)
18       Wj = wMIN
19     else if ( |wj - wj'|> T )
20       Wj = min(wj , wj')
21     else
22       Wj =( wj + wj')/2
23   outliers value of weithts?   goto step 2: continue
24 end
```

Figure 8 Pseudocode of CI.

base classifiers are calculated by introducing MI for performance judgment in each procedure. An adaptive strategy is used to calculating the final weight of a classifier according to the weights generated in the two-weighting procedure. Finally, the strong classifier is constructed similar with AdaBoost according to base classifiers' weight.

Each step of LDABoost uses the former step as its basis. Moreover, all strategies, methods and algorithms used in LDABoost had been verificated effective by previous researchers or are proofed feasible in theoretically in this article. The framework of LDABoost is shown in Figure 9.

The detail workflow of TC using LDABoost is:

1. Input document set.
2. Document set modeling.
3. Model simplification and LDA parameters estimation.

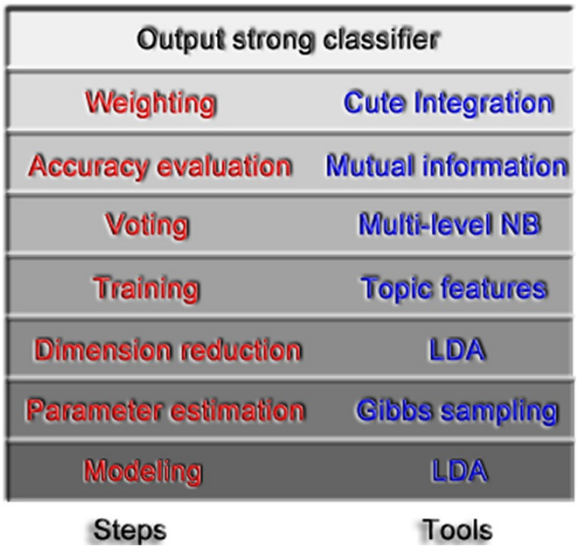

Figure 9 Framework of LDABoost.

4. Topics features extraction.
5. Train multi-level NB by training set.
6. Weak classifiers formed a committee.
7. Weak classifiers voting.
8. Additional voting by input training samples in reverse order.
9. Base classifiers' classification performance evaluation according to MI.
10. Weight allocation based on Steps 7–9
11. According to the weights of weak classifiers to generate a strong classifier.
12. Input test set.
13. Text classification using LDABoost.
14. Output category.

Follow the steps above, the object set of text will be classified in a high accuracy and high efficient way.

6. System application and analysis

The novel text classification tool which called LDABoost in this article is fully proposed in the former sections. To evaluating its performance in real world we made large number of tests to measure LDABoost's precision and time consumption. In addition, we also deployed several experimental control groups and referenced a lot of related literatures to make our conclusion about the performance of LDABoost. We use same training sets and same test set downloaded from same corpora. What's more, all experiments were done on the same platform. Therefore, the only variable is the classification tools.

Hardware and software environments used in the experiment section are shown in Table 1.

We use texts download from standard corpora. For evaluating its performance in different language, Reuters

Table 1 Hardware and software environments of the experiment

Item	Product	Edition/Indicator
CPU	Intel Core 2 Duo	2.93GHz
Memory	Kingston DDR3 1333	2G
Hard disk	Seagate ST500DM002	500G
OS	Windows XP	Professional SP3
IDE	Eclipse	3.4
Simulation tool	Matlab	7.0

21578 Classic text categorization corpus and CCL (Peking University modern Chinese corpus) are used.

6.1. Efficiency test and analysis

Huge time consumption is the major reason of why some theoretically high-accuracy classification algorithms could not get out of the laboratory. Therefore, when appraising a TC tool, the runtime complexity must be taken into account.

In TC field, most classic algorithms include neural networks, NB, SVM, decision tree, Rocchio, AdaBoost, *k*-nearest neighbors, etc. We have chosen most four representative algorithms of them: neural networks, NB, SVM, and AdaBoost for the comparative experiment with LDABoost. Two hundred thousands of English texts were downloaded from Reuters 21587 and two hundred thousands of Chinese texts were downloaded from CCL. In each language, we use a hundred thousands of texts as the training set and the others as the test set. For controlling the number of variables, each text is 2 KB. txt document. For ease of display, logarithmic axe is used to indicating that the amount of documents. The results of test are shown in Figure 10.

In the above figure, the unit of *X*-axis is second (s) and the unit of *Y*-axis is $\log_{10}$ (number of documents). We choose different modes to evaluate the efficiency contribution of different strategies in LDABoost. LDABoost is the original LDABoost algorithm proposed in this article. LDABoost.1 uses VSM model and words for feature representation. LDABoost.2 uses LDA to reducing dimensionality but give up CI which is the smart mechanism of strong classifier generation.

Open source toolkits: JGibbLDA [22], svmcls 2.0 [23], ParzenPNN, and CLIF_NB [24] are used for the test. In order to meet the requirements of this article, we made some modifications to the source code.

Figure 10 reveals that NB has the highest efficiency and SVM needs the longest classification time. The time consumption of LDABoost is much lower than neural networks and SVM. In addition, it is more effective than original AdaBoost.

In all editions of LDABoost, LDABoost.2 has the best efficiency. That probably because CI leads to additional time overhead. However, the difference of time consumption between LDABoost and LDABoost.2 is small. The reason probably is CI has low runtime complexity. Using LDA for feature extraction can significantly improve efficiency according to the experiment result. Approximately 10% of time costs are saving by using topic features.

It is interesting to note that various tools have roughly the same efficiency in English and Chinese documents processing. Original AdaBoost is exception which has a bit higher time cost than neural works when classifying Chinese texts.

6.2. Experiment for precision analysis

Precision is the most important criterion for evaluating the performance of TC system. Since the most data in internet are textual information, the precision of TC will largely determine the extent of our information utilization, even affect our life quality.

Therefore, we measured the novel algorithm's precision carefully and Referenced to a lot of previous literatures to comparing its accuracy with other classic classification algorithms. We selected 60,000 English documents and 60,000 Chinese documents randomly from 6 categories: Society, Economics, Science, Politics, Military, and Culture. For each language, every category has 5,000 training texts and 5,000 test texts. The accuracy of LDABoost, LDABoost.1, and LDABoost.2 are compared with NB [25], neural networks [26], SVM [27], and AdaBoost [28]. The comparative results are shown in Tables 2 and 3.

As shown in above tables, standard LDABoost has higher accuracy than other algorithm. The performance of LDABoost is far beyond NB and neural networks. In addition, the novel algorithm has better performance than SVM and original AdaBoost. That because boosting itself is a powerful ideology, LDA and CI further improve its performance. Comparative data of LDABoost.1 and LDABoost.2 proved the contribution of LDA and CI. Without both of them, LDABoost will be similar with original AdaBoost. Therefore, the performances of LDABoost.1 and LDABoost.2 are better than AdaBoost and worse than LDABoost.

Some text classification tools have a problem of training set scale dependence. It means those algorithms need a very large-scale labeled training set to ensure the accuracy of classification. However, large-scale labeled training set has an extremely high labor cost and not readily available. We test the precision of LDABoost when using different size of training sets. The result is shown in Figure 11.

We use 2,000, 4,000, 6,000, 8,000, 10,000 and 20,000 texts as the training sets. Figure 11 reveals that the precision of LDABoost increases very slowly while the size of training set increases largely. Although the algorithm proposed

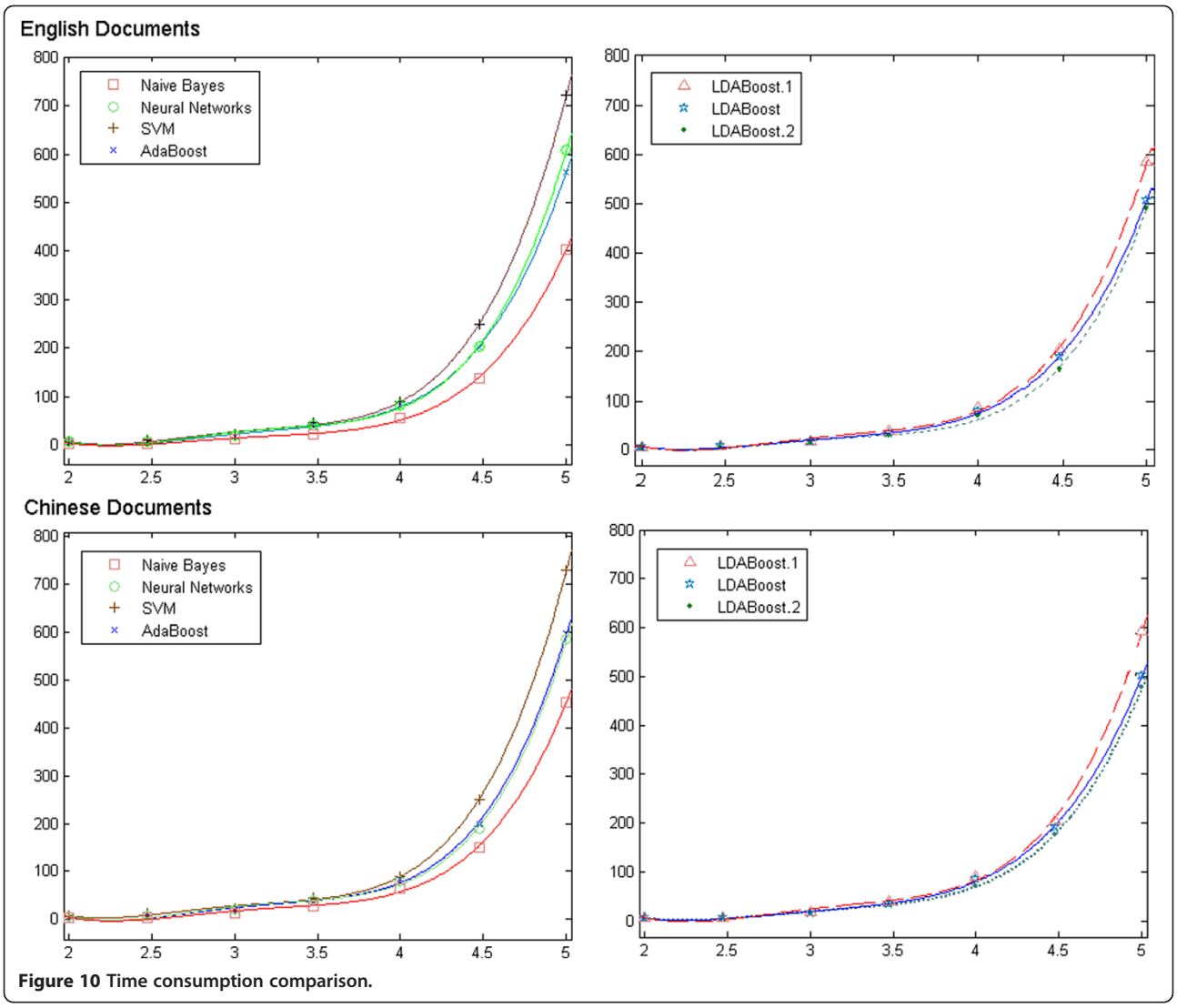

Figure 10 Time consumption comparison.

in this article is not absolutely size-independent, the correlation between algorithm's accuracy and size of training set is low enough for building a high performance classification with very little manual cost.

Moreover, experimental results shown that there is no significant different between the English and the Chinese texts classification precisions. System can be considered as language-insensitive.

Table 2 Precision of different algorithms in English TC

Category algorithm	Society	Economics	Science	Politics	Military	Culture
NB	0.781	0.769	0.774	0.799	0.772	0.773
Neural networks	0.818	0.830	0.829	0.815	0.815	0.806
SVM	0.859	0.868	0.863	0.868	0.864	0.871
AdaBoost	0.864	0.860	0.853	0.854	0.867	0.866
LDABoost	0.904	0.897	0.901	0.911	0.892	0.912
LDABoost.1	0.854	0.866	0.863	0.870	0.867	0.871
LDABoost.2	0.863	0.871	0.858	0.864	0.870	0.855

Table 3 Precision of different algorithms in Chinese TC

Category algorithm	Society	Economics	Science	Politics	Military	Culture
NB	0.785	0.769	0.771	0.794	0.769	0.772
Neural networks	0.817	0.832	0.825	0.807	0.808	0.803
SVM	0.847	0.867	0.852	0.862	0.860	0.871
AdaBoost	0.856	0.848	0.851	0.855	0.861	0.859
LDABoost	0.899	0.896	0.904	0.907	0.879	0.910
LDABoost.1	0.851	0.868	0.855	0.858	0.867	0.869
LDABoost.2	0.863	0.859	0.877	0.866	0.866	0.872

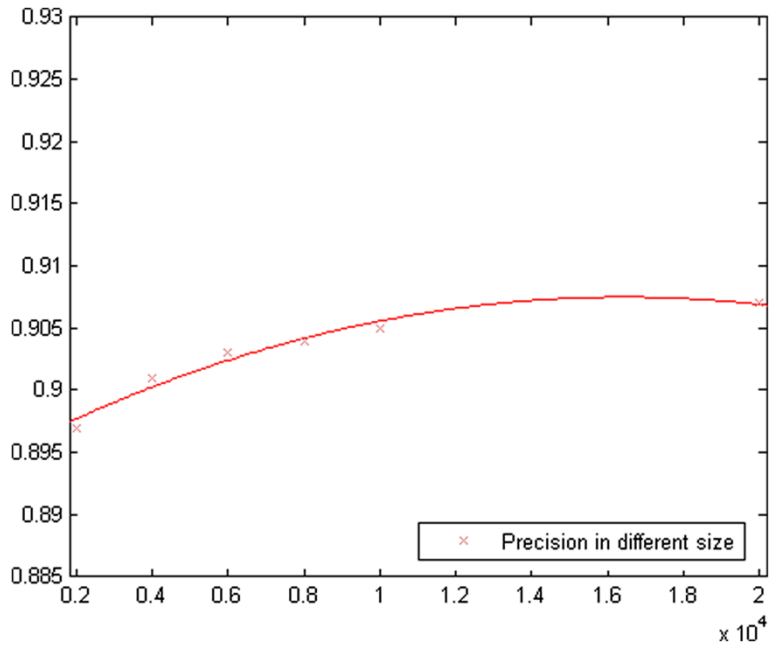

Figure 11 Prision of LDABoost with different size of train set.

In a word, LDABoost is an excellent tool for TC, it achieves really high accuracy while control the runtime complexity in a very low degree. That because the feature extraction based on LDA improves the efficiency and accuracy, the two-procedure MI based strong classifier generation mechanism further enhances the precision.

7. Conclusion and future work

An improved boosting algorithm is proposed in this article. It uses LDA as the dimension reduction tool to extracting topic features. This method largely decreased the feature dimensionality. To the best of the authors' knowledge, it is the first time LDA be introduced into boosting algorithm, this innovation enhance accuracy and efficiency at the same time. A multi-level NB algorithm is designed as weak classifiers. It keeps the advantage of high efficiency in original NB and has higher accuracy. Furthermore, different with AdaBoost, a two-procedure weighting algorithm which uses MI as the judgment of base classifiers' performance is used to construct the final strong classifier. Experimental result shown that the novel algorithm has lower time consumption and higher efficiency than many other categorization tools. In addition, LDABoost is proved language-insensitive and not large training set dependent.

However, probably the parameters of LDA could be estimated in a more efficient and accurate way. Furthermore, LDABoost based on other weak classifiers such as C4.5, kNN, or SVM may achieve higher precision or lower runtime complexity. The utility of LDABoost in other classification tasks such as image processing and speaker identification should be tested. This will be undertaken as future works on this topic.

Competing interests
The authors declare that they have no competing interests.

Acknowledgement
The authors have partially been supported by the China Association for Science and Technology.

Received: 24 April 2012 Accepted: 19 October 2012
Published: 2 November 2012

References

1. D Thorleuchter, D Van den Poel, A Prinzie, Mining ideas from textual information. Expert Syst Appl **37**(10), 7182–7188 (2010)
2. C Dinga, T Lib, W Peng, On the equivalence between Non-negative Matrix Factorization and Probabilistic Latent Semantic Indexing. Comput Stat Data An **52**(8), 3913–3927 (2008)
3. DM Blei, AY Ng, MI Jordan, Latent Dirichlet allocation. J Mach Learn Res **3**(1), 993–1022 (2003)
4. H Kim, P Howland, H Park, Dimension reduction in text classification with support vector machines. J Mach Learn Res **6**(1), 37–53 (2006)
5. S Paris, B Raj, S Madhusudana, Sparse and shift-invariant feature extraction from Non-negative data. Int Conf Acoust Spee, 2069–2072 (2008)
6. E Stark, Indefiniteness and specificity in Old Italian texts. J Semitic Stud **19**(3), 315–332 (2002)
7. P Claudia, P Foster, Aggregation-based feature invention and relational concept classes. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 167–176 (2003)
8. W Cui, S Liu, L Tan, S Conglei, S Yangqiu, G Zekai, Q Huamin, T Xin, IEEE transactions on visualization and computer graphics. **17**(12), 2412–2421 (2011)
9. R Feldman, J Sanger, *The Text Mining Handbook- Advanced Approaches in Analyzing Unstructured Data [M]* (Post &Telecom Press, Beijing, 2009), pp. 4–7
10. C Qiang, ZH Chengjie, Selecting maximally separable feature subset for multiclass classification with applications to high-dimensional data. Lect Notes Comput Sc **8**(2), 1217–1233 (2001)

11. D Newman, A Asuncion, P Smyth, M Welling, Distributed inference for latent Dirichlet allocation. Adv Neural Inf Syst **16**(4), 1–9 (2007)
12. D Xing, M Girolami, Employing LatentDirichletAllocation for fraud detection in telecommunications. Pattern Recogn Lett **28**(13), 1727–1734 (2007)
13. MA Chappell, AR Groves, B Whitcher, MW Woolrich, Variational Bayesian inference for a nonlinear forward model. IEEE T Signal Proces **57**(1), 223–236 (2009)
14. X Dong, MA Gonzalez Ballester, G Zheng, Automatic extraction of femur contours from calibrated X-Ray images using statistical information. J of Mult **2**(5), 46–54 (2007)
15. G Heinrich, Parameter estimation for text analysis. http://www.arbylon.net/publication/text-est.pdf
16. A García-Serrano, X Benavent, R Granados, JS Goñi-Menoyo, Some results using different approaches to merge visual and text-based features in CLEF'08 photo collection. 9th Workshop of the Cross-Language Evaluation Forum, 568–571 (2009)
17. J Mitéran, J Matas, E Bourennane, M Paindavoine, J Dubois, Automatic hardware implementation tool for a discrete adaboost-based decision algorithm. Eurasip J Appl Sig P **7**, 1035–1046 (2005)
18. C Tiantan, L Hongwei, Z Shuisheng, Large scale classification with local diversity AdaBoost SVM algorithm. J Syst Engin El **20**(6), 1344–1350 (2009)
19. H Schwenk, Y Bengio, Boosting neural networks. Neural Comput **12**(8), 1869–1887 (2000)
20. G Rätsch, T Onoda, K-R Müller, Soft Margins for AdaBoost. Mach Learn **42**(3), 287–320 (2001)
21. H Peng, F Long, C Ding, Feature selection based on mutual information: criteria of Max-dependency, Max-relevance, and Min-redundancy. IEEE Trans. Pattern Anal. Mach. Intell **27**(8), 1226–1238 (2005)
22. http://jgibblda.sourceforge.net
23. W Shaojun, L. Qi, Y Peng, P Xiyuan, CLS-SVM: a local modeling method for time series forecasting. Chinese J Scien Instrum **32**(8), 1824–1829 (2011)
24. L Liu, H Song, Y Lu, Method of CLIB_NB text classification learning based on naive bayes. Mini-Micro Syst **28**(9), 1575–1577 (2005)
25. I Rish, An empirical study of the naive Bayes classifier. IJCAI-01 Workshop on Empiracal Method in Artificial Intelligence, 41–46 (2001)
26. ME Ruiz, P Srinivasan, Hierarchical text categorization using neural networks. Information Retrieval **5**(1), 87–118 (2002)
27. M Arun Kumar, M Gopal, A comparison study on multiple binary-class SVM methods for unilabel text categorization. Pattern Recogn Lett **34**(11), 1437–1444 (2010)
28. E Romero, L Marquez, X Carreras, Margin maximization with feed-forward neural networks: a comparative study with SVM and AdaBoost. Neurocomputing **57**, 313–344 (2004)

doi:10.1186/1687-6180-2012-233
Cite this article as: Lei *et al.*: LDA boost classification: boosting by topics. *EURASIP Journal on Advances in Signal Processing* 2012 **2012**:233.