Knowledge-Based Systems 75 (2015) 152–160

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

TESC: An approach to TExt classification using Semi-supervised Clustering

Wen Zhang [a,*], Xijin Tang [b], Taketoshi Yoshida [c]

[a] School of Economics and Management, Beijing University of Chemical Technology, Beijing 100029, PR China
[b] Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, PR China
[c] School of Knowledge Science, Japan Advanced Institute of Science and Technology, 1-1 Ashahidai, Nomi, Ishikawa 923-1292, Japan

ARTICLE INFO

Article history:
Received 17 February 2014
Received in revised form 24 November 2014
Accepted 25 November 2014
Available online 8 December 2014

Keywords:
Text classification
Semi-supervised clustering
Unlabeled data
Support vector machines
Expectation maximization

ABSTRACT

This paper proposes an approach called TESC (TExt classification using Semi-supervised Clustering) to improve text classification. The basic idea is to regard one category of texts from one or more than one components. Thus, we use clustering to identify the components in text collection. In clustering process, TESC makes use of labeled texts to capture silhouettes of text clusters and unlabeled texts to adapt its centroids. The category of each text cluster is labeled by the label of texts in it. When a new unlabeled text is incoming, we measure its similarity with the text clusters and give its label with that of the nearest text clusters. Experiments on Reuters-21578 and TanCorp V1.0 text collection demonstrate that, in text classification, TESC outperforms Support Vector Machines (SVMs) and back propagation neural network (BPNN), and produces comparable performance to naïve Bayes with EM (Expectation Maximization) however with lower computation complexity.

1. Introduction

Text classification, namely text categorization, is defined as assigning predefined categories to text documents, where documents can be news stories, technical reports, web pages, etc., and categories are most often topics (genres), pertinence, etc. [1] Whatever the specific method employed, a text classification task starts with a training set $D = (d_1, \ldots, d_n)$ of documents that are already labeled with a category $L \in C$ (e.g. sports, politics). The task is to construct a classification model f as Eq. (1), which is able to assign the correct class label to a new document d of the domain.

$$f : D \to C \quad f(d) = L \tag{1}$$

In order to obtain the classification model f, one typically assumes that one set of labeled data is used for training (this set is called as training set) and another set of labeled data (this set is called as test set, which is typically sampled from the same underlying distribution as the training set) is then used to measure the performance of the trained classification model f. Usually, performance of classification model f can be measured by precision, recall, accuracy, AUC (Area Under Curve [2]), etc. In this paper, we use accuracy as the performance indicator.

In the point of view of machine learning, clustering is a fundamental technique of unsupervised learning, where its task is to find the inherent structure from unlabeled data [3]. Usually, clustering is performed when no information is available concerning the membership of data items to predefined class labels [11]. Text clustering aims to partition a text collection into text clusters, such that texts within same cluster are more semantically similar to each other than they are to texts in other clusters. The notion of similarity can be expressed in various ways according to purpose of text clustering.

Recently, using unlabeled data to improve the performance of classification is becoming an interesting problem in data mining. The basic idea is that unlabeled data can be used to better estimate parameters in building classifier. Moreover, unlabeled data are abundant, easier to obtain than labeled data [4]. Although the combination of both labeled and unlabeled data can reduce variance of the classification model, some researchers argue that unlabeled data is not necessarily to improve the performance of classification. They hold that, in the opposite way, using unlabeled data to train classifier may lead to degradation in the performance due to an increase in bias [5]. However, it is widely admitted that when modeling assumptions are correct, or the decrease of variance is larger than the increase of bias, using unlabeled data will surely improve the classification performance [6]. Here, variance refers to the variability of predicted labels for a given data point and bias refers to

* Corresponding author.
E-mail addresses: zhangwen@mail.buct.edu.cn (W. Zhang), xjtang@amss.ac.cn (X. Tang), yoshida@jaist.ac.jp (T. Yoshida).

http://dx.doi.org/10.1016/j.knosys.2014.11.028
0950-7051/

the difference between the expected prediction of the model and the correct label.

Following this way, many semi-supervised clustering methods have been proposed for the task of classification, using both labeled and unlabeled data [17,18]. The basic idea behind these methods is that the population of data is generated by a mixture model, and there is a correspondence between each component and a predefined class [4]. For instance, SOM (Self-organizing mapping) clustering is used in [6] to label the unlabeled data in non-ambiguous nodes with the label of their nodes. Then, both the originally labeled data and the labeled data by SOM clustering are used to train a multi-layer perception (MLP) as the classifier. They reported that this method (we call it as DKS method later in Section 3.5) significantly improves performances of classification on all the datasets they used. However, it is unclear that in how much proportion we need to relabel the unlabeled data by MLP that were already labeled by SOM, and on how much confidence we can trust the relabeling given by SOM clustering. Demirez et al [7] proposed a semi-supervised clustering algorithm which combines benefits of supervised and unsupervised learning methods. Their basic idea is to find a set of clusters and minimize a linear combination of cluster dispersion, which is measured by mean square error (MSE) and cluster impurity measure. It is very difficult to trade off the two functions in the linear combination.

Generally, both labeled and unlabeled data are used in clustering either by adapting the similarity measure (similarity-based approach) or by modifying the search for appropriate clusters (search-based approach) [8]. Similarity-based approach uses labeled data to train the similarity metric to satisfy the constraints in supervised data. Search-based approach modifies the clustering algorithm so that user-provided labels or constraints (such as must-link constraints and cannot-link constraints) are used to bias the search for an appropriate partition. Basu et al [9] propose a method called MCP-KMeans which combines the above two approaches by adapting the objective function used in the K-means. EM algorithm is employed in their method to update the similarity metric of data points. They reported that similarity-based approach and search-based approach are comparable in performance of semi-supervised clustering. Moreover, the combination of these two approaches can produce a significant improvement in performance. However, the weights of constraints in the objective function are difficult to determine, and the objective function can merely converge to a local minima.

This paper proposes TESC, which is a search-based method, for text classification. Our goal of using semi-supervised clustering is not to improve cluster quality as done in Basu et al [9], but to improve text classification. Although these two tasks are closely related with each other, they are actually different in that the former is unsupervised learning to group texts with similar contents together, while the latter is essentially supervised learning to categorize texts into different categories according to their topics, genres, language types, etc. Similar to the work done in Nigam et al [4], we hold that a document category comprises one or several topics those are expressed by text components. A text component is constructed based on words of texts within the component. We regard that there are correspondences between text categories and text components. We use semi-supervised clustering to identify text components and further to use text components to predict labels of unlabeled documents.

The state-of-art semi-supervised learning techniques such as naive Bayes and EM algorithm [15] and DKS method [6], usually adopt an iterative manner to make use of unlabeled data to refine the classifier. Firstly, each unlabeled data sample will be given a label by the trained classifier of the time. Secondly, those unlabeled data samples with its given labels are used to retrain the classifier. Thirdly, this labeling and retraining procedures loop until the convergence of the classifier. However, TESC is innovative and different from the existing semi-supervised learning techniques in that it does not make use of those unlabeled data explicitly. TESC does not make use of the labeled and unlabeled data one after another in learning process as done by [6,15,17,18] but to cluster both labeled and unlabeled data together at the same time. On the one hand, unlike the state-of-art semi-supervised learning techniques to classify data samples into the given categories, TESC assumes that the data samples come from multiple components and uses clustering process to capture those components. On the other hand, only a small number of labeled data samples are used to characterize the silhouettes of the clusters and, the unlabeled data samples are used to decide the centroids of the clusters together with the labeled data samples.

The remainder of this paper is organized as follows. Section 2 describes problem formulation and TESC for text classification. An example is provided to explain the mechanism of TESC. We also discussed TESC in contrast to other clustering and classification methods. Section 3 evaluates TESC using experiments. SVM, BPNN and naive Bayes with EM (NBEM) were introduced briefly for performance comparison. Section 4 concludes the paper.

2. Semi-supervised clustering for text classification

This section describes the problem formulation of semi-supervised clustering for classification. We propose TESC and present an example to explain its mechanism as well as its advantages compared with other methods.

2.1. Problem formulation

Assuming that we have a document collection as $D = \{D^L, D^U\}$, where D^L is the collection of labeled documents and D^U is the collection of unlabeled documents. $\overline{D}$ is a subset of D and it is used to train the classification model f. Our goal is to find a partition C using the data $\overline{D} = \{\overline{D}^L, \overline{D}^U\}$, where $\overline{D} \subset D$, $\overline{D}^U \subset D^U$, $\overline{D}^L \subset D^L$, $C = \{C_1, \ldots, C_m\}$ and each $C_i = \left\{d_1^{(i)}, \ldots, d_{|C_i|}^{(i)}\right\}$ $(1 \leqslant i \leqslant m)$. Here, $\bigcup_{1 \leqslant i \leqslant m} C_i = \overline{D}$ and $C_i \cap C_j = \varnothing (1 \leqslant i \neq j \leqslant m)$. For all labeled documents in C_i, they are given same labels as l_{C_i}. After we obtain that partition C, we use it to train the classification model f with nearest-neighbor search to predict the labels of unlabeled texts D^U using Eq. (2).

$$l(d_i^{(u)}) = l_{C_j}, \quad C_j = \arg\min_{C_p} \|d_i^{(u)} - c_p\| \tag{2}$$

Here, c_p is the centroid of text cluster C_p. That is, an unlabeled text $d_i^{(u)}$ will be labeled by f as having the same label as C_p, which have the smallest distance with $d_i^{(u)}$. Thus, our goal is to find the partition C and construct f, which could predict class labels of unlabeled texts D^U. It should be noted that the distance between texts (which has the opposite meaning as similarity) should be predefined according to specific situations and based on underlying distribution of the data.

2.2. TESC: The proposed approach

The TESC approach comprises two processes: one is clustering process to identify components from both labeled and unlabeled texts and another is predicting process to use identified text components to label unlabeled texts D^U.

In clustering process, we use labeled texts to supervise learning silhouettes of text components and, unlabeled texts were used to adapt the centroids of text components. We refer to the silhouette of a text cluster as a hyper ellipsoid that has minimized hyper

Input:
$\overline{D}$: a collection of labeled and unlabeled texts;
Output:
C : a collection of labeled text clusters;
Procedure:
(1) Initialization
1. For each $d_i \in \overline{D}$
2. Construct a cluster candidate C_i using each d_i and label C_i with l_{d_i};
3. Set C_i as unidentified and the centroid of C_i as d_i;
4. Add C_i to cluster candidate set S_C;
5. End for
(2) Clustering
1. Loop while the number of unidentified and labeled cluster candidates in S_C is larger than 1;
2. Find a cluster candidate pair (C_i, C_j) in S_C, which has the smallest distance between centroids among all the possible identified cluster candidate pairs in S_C;
3. If C_i and C_j are all labeled cluster candidates and have different cluster labels
4. Then set C_i and C_j as identified;
5. Otherwise
6. Merge C_i and C_j into a new cluster candidate C_k;
7. Remove C_i and C_j from S_C;
8. Add C_k to S_C;
9. End if
(3) Output
1. Remove the cluster candidates whose size is smaller than 3 from S_C;
2. Output each cluster candidate C_i in S_C to C

Fig. 1. The process of TESC in clustering both labeled and unlabeled texts.

volume to accommodate all the data points in the cluster. Fig. 1 shows the detailed process of TESC that includes three steps: initialization, clustering and output.

In initialization, each text is regarded as a cluster candidate with label of the text. If a text is unlabeled, the label of the cluster candidate will be given as "unlabeled". In clustering, two cluster candidates with the smallest distance among all candidate pairs will be either merged into a new cluster candidate or identified as two unique clusters.

Four particular situations are encountered in handling the selected two candidates: (1) if the two candidates are all labeled and have different labels, they will be identified as two unique clusters; (2) if the two candidates are all unlabeled, they will be merged into a new cluster candidate with label as "unlabeled"; (3) if only one of the two candidates is unlabeled, they will be merged into a new cluster candidate with the label of the labeled candidate; (4) if both candidates are labeled and have same label, they will be merged into a new cluster candidate with the same label. In the last three situations, the newly merged cluster is added into the cluster candidate set and the original two cluster candidates are removed from the cluster candidate set.

We loop the clustering step until the number of unidentified labeled candidates is less than 2 because, if there are two unidentified candidates in the candidate set, they should be merged into a new candidate or identified as two unique clusters. In output, in order to avoid dead units [10,19], the identified clusters with more than 3 document members are retained for predicting because by our observation, dead nodes are mostly outliers and noises in the document collection and they will degenerate the performance of text classification.

After the semi-supervised clustering process as described in Fig. 1, the output text clusters are regarded as components corresponding to document categories and used to predict labels of unlabeled texts shown in Fig. 2. The basic heuristics we adopt here is to label an unlabeled text with the label of the text cluster that is nearest to the unlabeled text in Euclidian distance.

2.3. An example of TESC for classification

Assuming we have 15 data points sampled from 2 classes to classify shown in Table 1. The third dimension of the data point denotes its label and "?" represents the label of the data point is unknown. We use 12 data points in model training and the remaining 3 data points in classification. The training set includes 7 labeled and 5 unlabeled documents, to identify text components to train the classifier.

Fig. 3 shows the clustering process of TESC in clustering the 12 labeled and unlabeled texts in Table 1 to training the classifier. The blue point denotes category 1 and, the red plus denotes category 2 and, the green asterisk denotes the unlabeled data points. We see from Fig. 3(a) that first, two data points with smallest distance are merged into a new cluster (cluster a). Second, in Fig. 3(b), the new cluster (cluster a) has smallest distance with another unlabeled data point so the three data points are merged into a new cluster (cluster b). Third, in Fig. 3(c), another new cluster (c) is coming into being. After many iteration as depicted in Fig. 3(a)–(c), finally, the 12 data points are partitioned into 3 clusters as shown in Fig. 3(d).

It can be seen there are 3 clusters produced from the clustering process of TESC. Cluster 1 is labeled with category 2, cluster 2 is labeled with category 1 and cluster 3 is labeled with category 2. The role of labeled data points is to capture the silhouettes of each cluster and label them while the role of unlabeled data points is to adjust the centroids of clusters. After the clustering process

Input:
C: a collection of labeled text clusters;
$D^{(U)}$: a collection of unlabeled texts;
Output:
$Q^{(U)}$: a collection of labels corresponding to each text in $D^{(U)}$;
Procedure:
1. For each $d_i^{(U)} \in D^{(U)}$
2. $C_{temp} = C_0$;
3. $l_{d_i^{(U)}} = l_{C_{temp}}$;
4. For each $C_j \in S_C$
5. Dis($d_i^{(U)}, C_j$)=$\left\| d_i^{(U)} - C_j \right\|$;
6. If(Dis($d_i^{(U)}, C_{temp}$)>Dis($d_i^{(U)}, C_j$))
7. $C_{temp} = C_j$;
8. End if
9. End for
10. $l_{d_i^{(U)}} = l_{C_{temp}}$;
11. And $l_{C_{temp}}$ to $Q^{(U)}$;
12. End for

Fig. 2. The predicting process of TESC.

Table 1
The 15 data points used in the example to show TESC for classification.

No.	Data point	Category	Phase
1	(4.5,3.2)	1	Training
2	(5,2)	1	Training
3	(3,3)	?	Training
4	(5.7,4)	1	Training
5	(7.6,3.6)	2	Training
6	(7.4,3.2)	?	Training
7	(8.1,3.2)	?	Training
8	(9.6,4.2)	2	Training
9	(4,8)	2	Training
10	(4.5,9)	?	Training
11	(6,9)	?	Training
12	(7,8)	2	Training
13	(4,3.5)	?	Testing
14	(8.5,2.7)	?	Testing
15	(6,5)	?	Testing

(4 iterations), TESC labels data point 13 with category 1 because it is nearest to the centroid of cluster 2. By analogy, data point 14 is labeled as category 2 and data point 15 as category 1, respectively.

2.4. Analysis of TESC

The solution of classification by clustering lies in the assumption that there are correspondences between the underlying components behind the data and the categories of the data. We argue that this assumption is reasonable for the texts in a document collection. Usually, clustering is used to discover the topics of texts [3]. In most cases, there is a great need to categorize text according to their contents.

In theory, the data points in Fig. 4 cannot be easily classified if linear separable methods were employed. We sampled these data points from 4 bivariate normal distributions. Thus, SVM, Decision Tree, neural networks, etc., are developed to deal with this problem. However, intuitively, if we can identify that there are in fact four components or clusters in the dataset and each class contains two components or clusters, it would be more efficient to classify the data points. Moreover, due to the limited number of labeled data, we cannot capture the silhouettes of clusters precisely. Thus, unlabeled data are used in clustering process in TESC, as a way to use more data to train the classifier. Consequently, the distribution of data population is more accurately estimated and the components of data points are more accurately characterized by using more data points than not.

We claim the advantages of TESC in the following three aspects. First, the unlabeled data can be regarded as a kind of prior knowledge of the data we are dealing with. For documents in a collection, they can usually be classified by their content topics. Even if all documents in the collection belong to the same topic, they can be divided into different sub-topics. Thus, the idea of components in TESC could be regarded as composed by these sub-topics.

Second, in hierarchical clustering, we have the problem of deciding a critical point to divide clusters. As illustrated in Fig. 1, the clustering process TESC is very similar to agglomerative hierarchical clustering [11] in merging document pairs with the smallest distance. However, for TESC, labeled data is used to supervise partitioning clusters in clustering process. Here, the basic idea is that a cluster is included in a hyper ellipsoid whose silhouettes are characterized by different labels of different clusters. For instance, in Fig. 2, an ellipse is used to encircle the data points in a cluster if Euclidean distance is used for similarity measure. The kernel methods can also be used [13] to measure the similarities of data points if a complex hyper ellipsoid is used to capture the cluster.

Third, in k-nearest-neighbor (KNN) method for classification [12], it is required to predefine the number of neighbors for an unlabeled data to learn its label, that is, the number as k. However, in TESC, we regard k is dynamically changing based on the distribution of data on different categories and, the label of an unlabeled data point is given by its distance to the trained text clusters. That is, the unlabeled data point has the smallest distance to the centroid of the cluster that labels the data point. In this sense, for different data distribution on the whole space, we have different k neighborhoods for an unlabeled data point. The computation complexity of each loop of the proposed method in clustering step is $O(n^2)$, where n is the number of data points.

3. Experiments

In this section, a series experiments are conducted to evaluate the performance of the proposed method. Support Vector Machine (SVM), back-propagation neural network (BPNN), the method proposed by [6] DKS method and naïve Bayes with EM algorithm (NBEM) are used as the baseline methods.

3.1. SVM

Support Vector Machine (SVM) is employed to classify Reuters-21578 and Tancorp V1.0 documents as baseline performance.[1] We use linear kernel as $(u * v)$[1] for SVM training because it is superior to the non-linear kernel in text categorization indicated by our prior research [9]. Considering there are 4 categories we sample from Reuters-21578 and Tancorp V1.0 text collection, the One-Against-the Rest approach [9] for multi-class categorization is adopted for SVM classifier. We repeat each classification task for 10 times, by resampling training and test data, in order to reduce the variance of performance. The performance of SVM classifier is computed by averaging the accuracies of the 10 repetitions. The computational complexity of standard SVM is $O(n^3)$, where n is the number of data points [16].

3.2. BPNN

The back-propagation neural network (BPNN) [14] defines two propagations of the network: first a forward propagation from the input layer to the output layer and second a backward propagation

[1] Here, Libsvm is used to conduct the work of text categorization which is online and can be downloaded freely from: http://www.csie.edu.tw/cjlin/libsvm/.

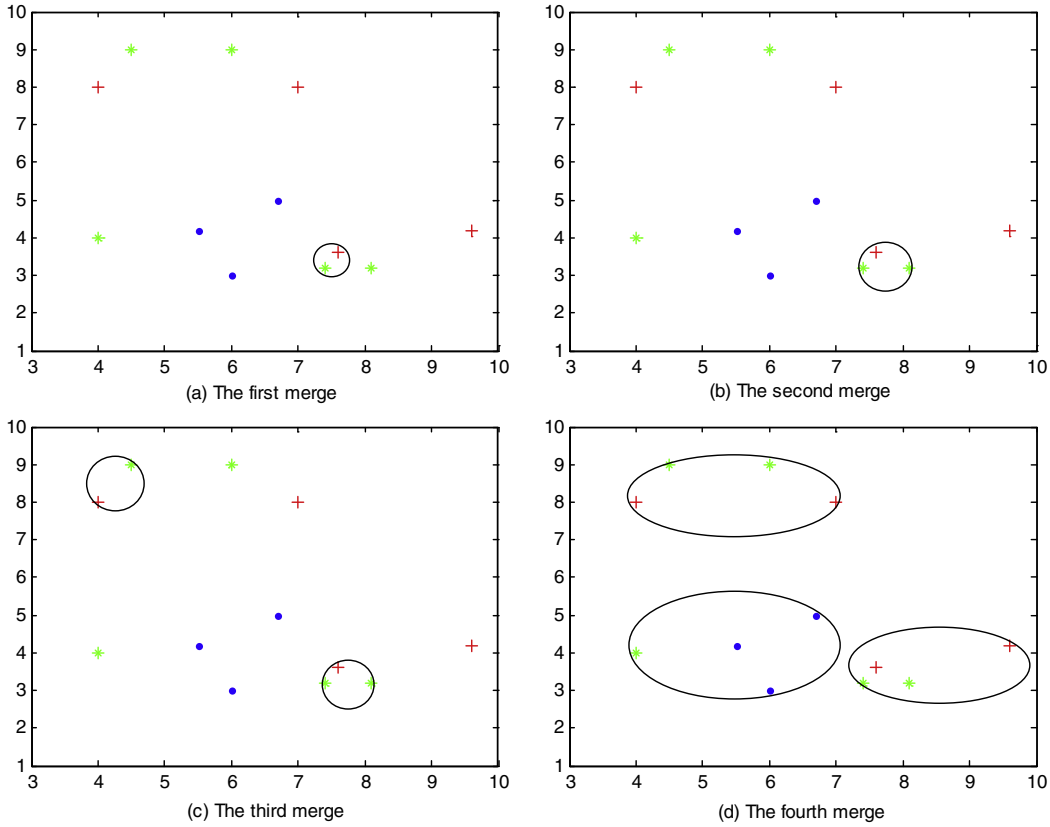


Fig. 3. The clustering process of 12 data points.

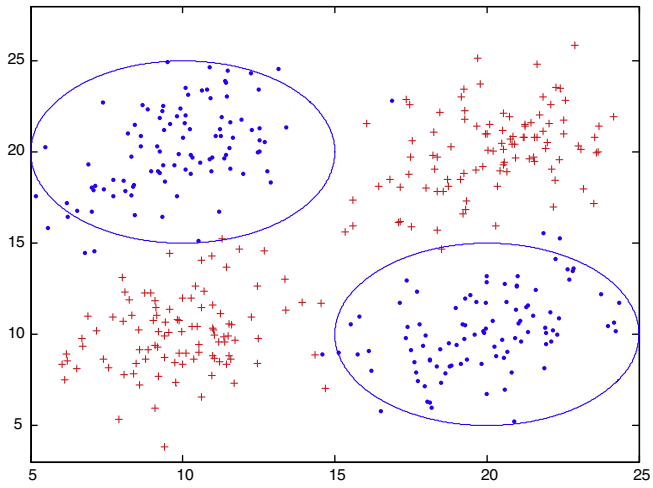


Fig. 4. The situation of data points that cannot be classified using linear separable classifier.

from the output layer to the input layer. The backward propagation is similar to the forward propagation except that error values are propagated back through the network to determine how the weights are to be changed during training. During training each input sample will have an associated target vector. The objective of training is to find a set of network weights that provide a solution to the particular problem at hand. We use tansigmod activation function $\left(y = \frac{1}{1+e^{-x}}\right)$ in hidden layer with 100 nodes and purelinear activation function $(y = x)$ in output layer for text classification. We learned that the time complexity of BPNN is $O(nmkt)$ from its implementation [14], where n is the number of data points, m is the number of hidden nodes, k is the number of output nodes and t is the number of number of iterations for convergence.

3.3. DKS

The DKS method is quite straightforward to combine SOMs and multi-layer perception for classification. First, SOMs is constructed using only labeled texts to group them into different clusters. Second, unlabeled texts are then mapped to the trained SOMs' nodes based on the associated weights. If an unlabeled text was mapped to a cluster in which all the training texts have same label, then the unlabeled is given the label unambiguously. Otherwise, the label of the unlabeled text is unknown. Third, the newly derived unambiguously labeled texts are used to train the SOMs again and label the remaining unlabeled texts iteratively until there is no change. Fourth, a multi-layer perception is constructed using the labeled (both in the originally labeled in the training set and those newly labeled by SOMs) texts for classification.

3.4. NBEM

NBEM is adopted from Nigam et al [4]. Naïve Bayes is based on the Bayes' theorem of posteriori probability and assumes that the effect of attribute value on a given class is independent of the value of the other attributes. This class conditional independence assumption simplifies computation involved in building the classifier so we call the produced classifier "naive". In text classification, naïve Bayes assumes word independence to produce a probabilistic generative model for text.

In the framework of naïve Bayes, a document, d_i, is considered to be an ordered list word, $\{w_{d_i,1}, \ldots, w_{d_i,|d_i|}\}$, where $w_{d_i,k}$ is the word in position k of document d_i. Thus, the probability of a document d_i with respect to a class C_j, $p(d_i/C_j)$, can be described using the product of the probabilities of its constituent words with respect to the class C_j, that is, $\prod_{k=1}^{|d_i|} p(w_{d_i,k}/C_j)$. The prior probability $p(C_j)$ is estimated by $\frac{1}{n}\sum_{i=1}^{n} p(C_j/d_i)$ using labeled documents,

where n is the number of labeled documents. The probability of class C_j given document d_i, i.e. $p(C_j/d_i)$, is proportional to $p(d_i, C_j)$, which can be figured out by $p(d_i/C_j)p(C_j)$.

The EM algorithm is used to update $p(w_{d_i,k}/C_j)$ by using $P(w_{d_i,k}|C_j) = \frac{\sum_{i=1}^{n} exist(w_{d_i,k})P(C_j|d_i)}{\sum_{j=1}^{m}\sum_{i=1}^{n} P(C_j|d_i)}$, where m is the number of categories in the text collection and n is the number of texts in the text collection (including both labeled and unlabeled documents). $exist(w_{d_i,k}) = 1$ if $w_{d_i,k}$ exists in the text d_i, otherwise, $exist(w_{d_i,k}) = 0$. Due to the varying posterior probability of the category C_j given the unlabeled document $d_i^{(U)}$ at each iteration, the probability of word $w_{d_i,k}$ with respect to C_j, $P(w_{d_i,k}|C_j)$, is updated. The loop will be terminated if $\prod_{i=1}^{n}\prod_{j=1}^{m} p(d_i|C_j)$ is converged. On can refer to [15] for more information about the detailed computation process of NBEM.

It can be deduced that NBEM [4] has computation complexity as $O(mnt)$ for each iteration, where m is the number of words in text collection, n is the number of documents and t is the number of components used to generate one class. Because m is usually much larger than n in a document collection, TESC has much smaller computation complexity ($O(n^2)$) than that of NBEM.

3.5. The datasets

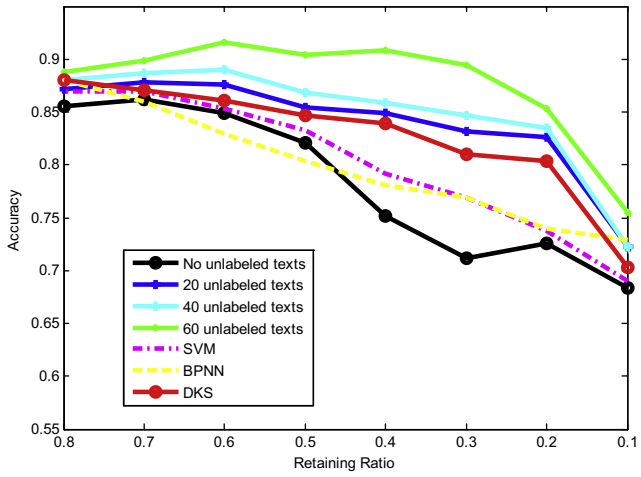
Reuters-21578 distribution 1.0 is used for experiments in this paper which is available online.[2] It collects 21,578 news from Reuters newswire in 1987. Since 1991, it appeared as Reuters-22173 and was assembled and indexed with 135 categories by the personnel from Reuters Ltd. in 1996. In this paper, the documents from 4 categories as "crude" (520 documents), "agriculture" (574 documents), "trade" (514 documents) and "interest" (424 documents) are assigned as the target English document collection. Thus, we select in total 2042 English documents, which have 50,837 sentences and 281,111 individual words after stop-word elimination. We obtain the stop-words from USPTO (United States Patent and Trademark Office) patent full-text and image database.[3] The part of speech of English word is determined by QTAG which is a probabilistic parts-of-speech tagger and can be downloaded freely online.[4]

TanCorp V1.0 is used as the Chinese corpus in the experiments which can be downloaded freely from the internet.[5] On the whole, this corpus has 14,150 documents in 20 categories from Chinese academic journals concerning computers, agriculture, politics, etc. In the experiments, texts from 4 categories, "agriculture", "history", "politics" and "economy" are assigned as target Chinese text collection because they have relatively larger size than other categories. For each of the 4 categories, we randomly sample 300 documents for it. As a result, 1200 texts are used in the experiments. They include 219,115 sentences and 5,468,301 individual words. We conduct Chinese morphological analysis using ICTCLAS.[6]

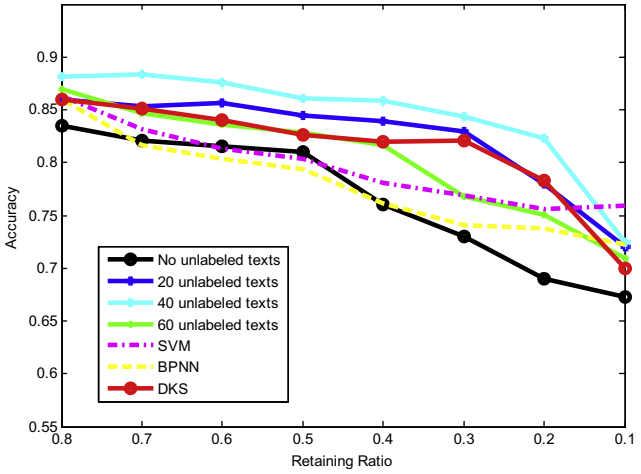
3.6. Experimental results

3.6.1. Varying retaining ratio

Fig. 5 shows the accuracies of text classification using TESC at different retaining ratios. The vertical axis indicates average accuracies in classifying test texts, and the horizontal axis indicates retaining ratio denoting percent of labeled texts used in TESC, SVM and BPNN. For instance, when we set retaining ratio as 0.1, that means we retain 10 percent of labeled documents to train



(a) Retuers-21578 text collection



(b) Tancorp V1.0 text collection

Fig. 5. Accuracies of text classification using TESC at different retaining ratios.

TESC, SVM, BPNN and DKS classifiers and, the remaining 90 percent of texts are used in testing the trained classifier. We do not increase the retaining ratio to 0.9 because all the mentioned methods have similar accuracies with little differences. Moreover, at retaining ratio 0.9, TESC cannot obtain enough unlabeled texts as its input.

It can be seen from Fig. 5 that TESC outperforms SVM, BPNN and DKS classifiers. In the best case of Reuters-21578, when the retaining ratio is 0.3, the accuracy of TESC with 60 unlabeled texts (0.8943) is increased by 16.3%, 16.1% and 10.04% in comparison with those of SVM (0.7690), BPNN (0.7702) and DKS (0.8100). In the worst case, which is at retaining ratio 0.8, the accuracy of TESC with 60 unlabeled texts (0.8881) is slightly larger than those of SVM (0.8783), BPNN (0.8723) and DKS (0.8800).

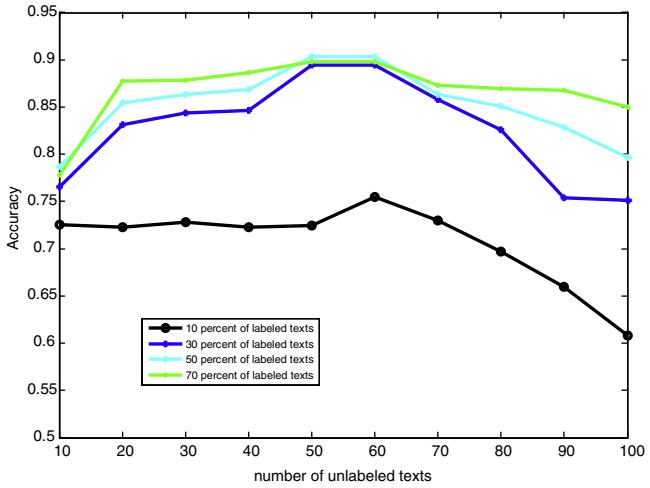
In the best case of TanCorp V1.0, when the retaining ratio is set as 0.2, the accuracy of TESC with 40 unlabeled texts (0.8231) is increased by 8.9%, 11.5% and 5.08% in comparison with that of SVM (0.7558), BPNN (0.7380) and DKS (0.7833). In the worst case, which is at retaining ratio 0.8, the accuracy of TESC with 40 unlabeled texts (0.8812) is slightly larger than those of SVM (0.8634), BPNN (0.8598) and DKS (0.8600). This outcome illustrates that text clusters do naturally exist within document categories. Texts cannot be categorized easily by SVM and BPNN classifiers but can be identified by TESC approach.

[2] http://www.research.att.com/~lewis.
[3] http://ftp.uspto.gov/patft/help/stopword.htm.
[4] http://www.english.bham.ac.uk/staff/omason/software/qtag.html.
[5] http://www.searchforum.org.cn/tansongbo/corpus.htm.
[6] http://nlp.org.cn/~zhp/ICTCLAS/codes.html.

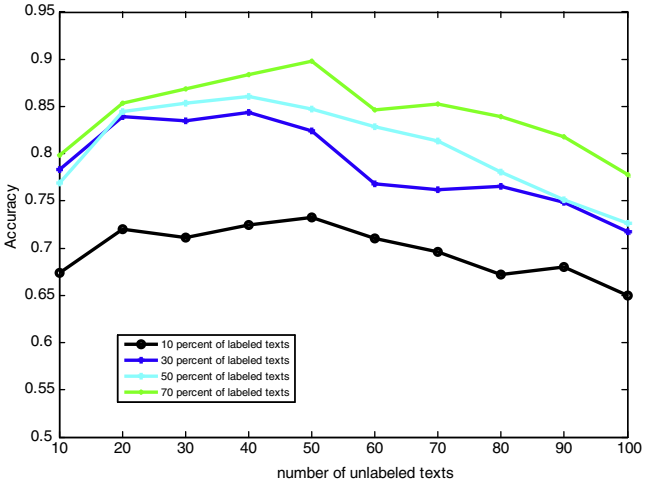
The performances of TESC with unlabeled documents are better than those without unlabeled documents. In the best case of Reuters-21578, which is at the retaining ratio as 0.3, the accuracy with 60 unlabeled texts (0.8943) is increased by 25.6% in comparison with the accuracy with no unlabeled texts (0.7117). In the best case of TanCorp V1.0, which is at the retaining ratio as 0.2, the accuracy with 40 unlabeled texts (0.8230) is increased by 19.3% in comparison with the accuracy with no unlabeled texts (0.6900).

In the worst case of Reuters-21578, which is at the retaining ratio as 0.7, the accuracy with 60 unlabeled texts (0.8984) is of 4.17% increase than that with no unlabeled texts (0.8624). In TanCorp V1.0, which is at the retaining ratio as 0.8, the accuracy with 40 unlabeled texts (0.8815) is of 5.54% increase than that with no unlabeled texts (0.8352). This outcome illustrates that unlabeled texts augment TESC to improve its performance in text classification.

Moreover, for each category, we used several unlabeled text sets having the same size in our experiments (i.e., we used different 20/40/60 unlabeled texts for each category without overlap in the experiments of Fig. 5). We observed that the derived experiment results are very similar to each other despite there are minor differences. We found that when different unlabeled text sets are used in clustering process, although those labeled clusters produced by TESC are slightly different from each other, the silhouettes of labeled clusters are very similar to each other due to the relatively smaller number of unlabeled texts than the number of labeled texts. Nevertheless, these different labeled clusters actually do not bring about much difference for the derived classification performance. If both the labeled and unlabeled texts were selected randomly in the experiment, the performance of TESC would be kept stably on text classification. The labeled texts to characterize the silhouettes of text clusters however, the unlabeled texts can only adjust their centroids in the clustering process. Although the labeled clusters will be different due to different selection of unlabeled texts, the classification for a test text is not changed much because it is always nearer to clusters of a certain label than to clusters of another label. This is the very reason that different selection of unlabeled texts involved in TESC would not change its performance significantly. That is, TESC is robust to produce classification results.



(a) Retuers-21578 text collection



(b) Tancorp V1.0 text collection

Fig. 6. Accuracies of text classification using TESC with different percent of labeled texts.

3.6.2. Varying number of unlabeled texts

Fig. 6 shows the accuracies obtained by varying the number of unlabeled texts when the number of labeled texts is predefined. For instance, on Retuers-21578 dataset, when we set 10 percent of labeled texts, that means that 52 texts from the category "crude", 58 texts from the category "agriculture", 52 texts from the category "trade" and 43 texts from the category "interest" with varying number of unlabeled texts indicated by the horizontal axis are used in training TESC classifier.

We can see from Fig. 6 that labeled texts can improve the performance of text classification readily. However, unlabeled texts can merely improve performances on two conditions: (1) the number of unlabeled texts is not too large; (2) the number of labeled texts is small. We speculate that when the number of unlabeled texts is small, the variance of each cluster is reduced by adding unlabeled texts in training. However, when more and more unlabeled documents are added into the training data, the bias of each cluster, which means the discrepancies between the centroids of the clusters and the centroid of the population, is also increased by the unlabeled texts.

Tables 2 and 3 show the number of text clusters of Reuters-21578 and TanCorp V1.0 produced by TESC when we obtain the accuracies in Figs. 5 and 6, respectively. It can be seen that when the more documents are used in TESC, the more clusters are discovered in the document collection. We draw that when the number of clusters is small (from 7 to 18 for Reuters-21578 dataset and from 5 to 13 for TanCorp V1.0 dataset); the performances of TESC in text classification are not good because, in this case, each cluster mixes with more than one category. When the number of clusters is too large (from 41 to 58 for Reuters-21578 dataset and from 36 to 42 for TanCorp V1.0 dataset), the performances of TESC in text classification are not good either, because, in this case, the number of documents in a cluster is too small and they cannot capture the silhouette of the cluster precisely as the centroids of clusters are prone to be dominated by the random co-occurrences of words in texts.

3.6.3. TESC vs NBEM

Fig. 7 shows the comparative results between TESC and NBEM with different number of mixture components. Here, we use 70 percent of documents as labeled documents, because too many labeled texts will constrain unlabeled documents to make effects on clustering for both methods. We set the retaining ratio as 0.5 because we see from Fig. 6 that at this retaining ratio, TESC produce best performances in both datasets. It can be seen from Fig. 7 that, for both Reuters-21578 and TanCorp V1.0 datasets, when the number of mixture components in NBEM is varied from 3 to 9, the performance of NBEM is improved significantly. The performance of NBEM decreases when we set its number of mixture

Table 2
The number of clusters produced by TESC corresponding to the accuracies of Fig. 5 ("R.R" stands for retaining ratio and "UD" stands for unlabeled documents). The numbers out of parentheses are of Retuers-21578 dataset and the numbers within parentheses are of TanCorp V 1.0.

# of UDs	R.R							
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0	5 (6)	8 (9)	19 (11)	20 (13)	19 (16)	23 (18)	27 (22)	29 (23)
20	11 (9)	14 (12)	18 (14)	24 (15)	24 (18)	29 (21)	31 (25)	34 (29)
40	15 (11)	20 (13)	26 (17)	30 (17)	31 (19)	33 (23)	38 (28)	40 (33)
60	15 (13)	24 (16)	28 (22)	31 (21)	37 (24)	36 (28)	43 (32)	45 (37)

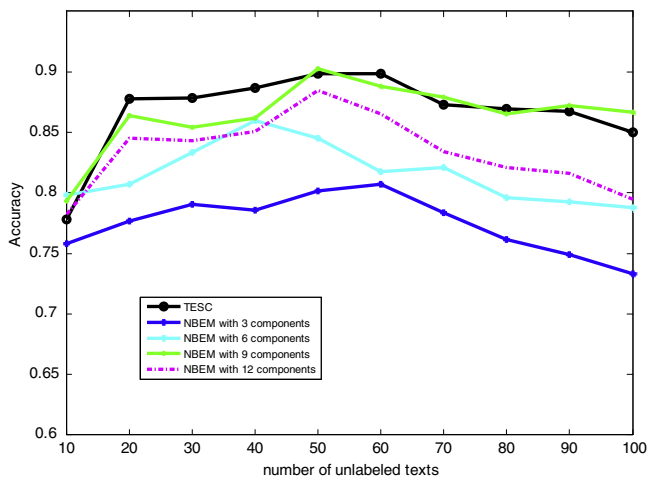
Table 3
The number of text clusters produced by TESC corresponding to the accuracies of Fig. 6. ("R.R" stands for retaining ratio and "UD" stands for unlabeled documents.) The numbers out of parentheses are of Retuers-21578 dataset and the numbers within parentheses are of TanCorp V 1.0.

# of UDs	R.R			
	0.1	0.3	0.5	0.7
10	7 (5)	15 (13)	22 (14)	27 (21)
20	11 (9)	18 (14)	24 (18)	31 (25)
30	12 (11)	22 (16)	29 (21)	35 (26)
40	15 (11)	26 (17)	31 (19)	38 (28)
50	15 (12)	26 (19)	34 (22)	41 (30)
60	15 (13)	28 (22)	37 (24)	43 (32)
70	17 (16)	32 (23)	38 (36)	47 (35)
80	19 (18)	34 (25)	41 (38)	51 (38)
90	23 (21)	38 (29)	44 (39)	53 (39)
100	26 (23)	42 (31)	47 (39)	58 (42)

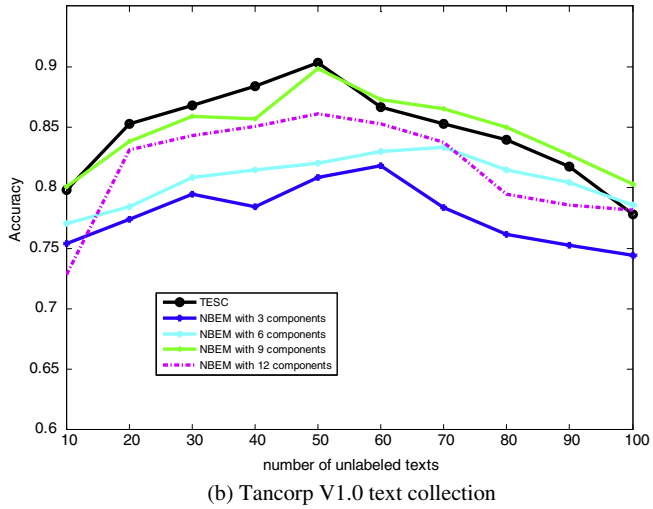
components as 12. Thus, we speculated that for both Reuters-21578 and TanCorp V1.0 datasets, the best number of mixture components of each class is 9.

In Reuters-21578 dataset, the performance of TESC is comparable with that of NBEM with 9 mixture components. The best performance of TESC as 0.8984 is produced with 60 unlabeled texts and, the best performance of NBEM with 9 components as 0.8978 is obtained with 50 unlabeled texts. The worst case of TESC is with 10 unlabeled texts as 0.7781 and, the worst performance of NBEM with 9 components as 0.7994 comes up with 100 unlabeled texts.

In TanCorp V1.0, TESC outperforms NBEM when the number of unlabeled texts is smaller than 50. The best case of TESC turns up

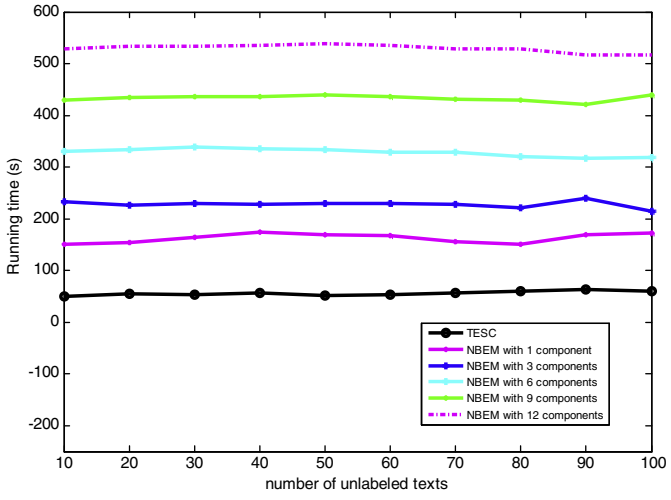


(a) Retuers-21578 text collection

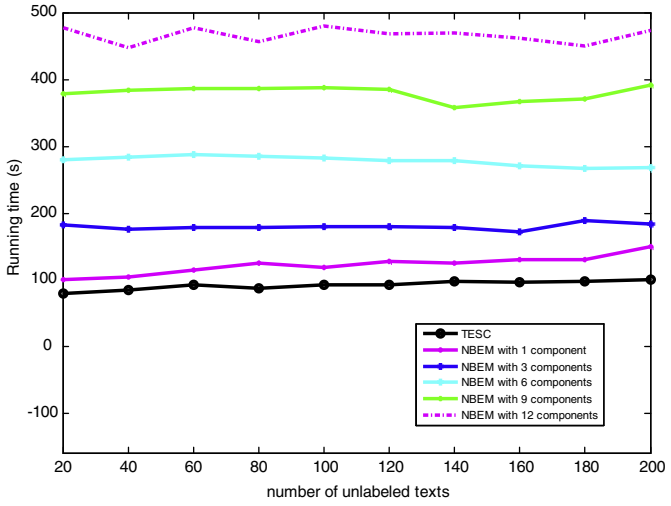


(b) Tancorp V1.0 text collection

Fig. 7. Accuracies of text classification of NBEM compared with TESC, both at retaining ratio 0.5.



(a) Retuers-21578 text collection



(b) Tancorp V1.0 text collection

Fig. 8. The running time of TESC and NBEM (at the retaining ratio as 0.5).

when the number of unlabeled texts is 50 where it produces accuracy as 0.9032. The best case of NBEM with 9 mixture components produces accuracy as 0.8990 when the number of unlabeled texts is 50. The worst case of TESC turns up when the number of unlabeled texts is set as 100 where TESC comes up with accuracy as 0.7781 and for NBEM, with 9 mixture components, it produces accuracy as 0.8008 with 10 unlabeled texts. Thus, we may draw that in most cases, TESC is comparable to NBEM in text classification.

3.6.4. Running time

Fig. 8 reports the running time consumed by TESC and NBEM algorithm on Reuters-21578 and TanCorp V1.0 datasets, respectively, to complete the task of text classification with different initial settings as in Fig. 7. All the experiments run on a PC with a 2 GHz CPU and a 2 GB RAM. As expected, due to its lower computation complexity as described in Section 3.3, TESC runs much faster than NBEM algorithm, with 5–6 times savings in running time. The running time of NBEM algorithm is increased dramatically with the initialized number of components. The number of unlabeled texts has merely a small effect on the running time of both TESC and EM algorithm. The running time keeps stable when we vary the number of unlabeled texts. The reason is that its number is much smaller than that of labeled texts which consume most computation. Thus, it can be drawn that although TESC can produce merely comparable performance to NBEM, it is at least half order of magnitude scalable than the latter in dealing with huge amount of texts that includes a large number of text components.

4. Concluding remarks

In this paper, we propose TESC, an approach using semi-supervised clustering, to text classification. Our basic assumption is that each category of documents comes from multiple components, which can be identified by clustering. In order to elaborate the clustering process, unlabeled documents are utilized to adapt the centroids of cluster candidates iteratively and labeled documents are utilized to capture the silhouettes of the clusters.

TESC is a search-based approach to semi-supervised clustering. After describing the detailed process of TESC, we present an example to explain its mechanism in text classification and a theoretical analysis of TESC. Then, we conduct a series of experiments to evaluate the performances of TESC in text classification. The experimental results demonstrate that TESC outperforms SVM, BPNN and the DKS method [6], is comparable to NBEM with better scalability.

Although the experimental results have provided us with some promising aspects of TESC in text classification, we admit that our work is currently on the initial step. More investigations and experiments should be undertaken to validate TESC approach, especially its scalability to handle big data. Moreover, we notice that there are unbalanced categories in our datasets and we will adopt TESC to handle unbalanced problem in text classification.

Acknowledgments

This work is supported by the National Science Foundation of China under Grant Nos. 71101138, 61379046, 61473284 and 7171187; the Beijing Natural Science Fund under Grant No. 4122087; the Research Fund from State Key Laboratory of Software Engineering, Wuhan University. The authors of the paper would like to appreciate the anonymous reviewers for their valuable comments.

References

[1] W. Zhang, X. Tang, T. Yoshida, Text classification toward a scientific forum, J. Syst. Sci. Syst. Eng. 16 (3) (2007) 356–369.
[2] A.P. Bradley, Half-AUC for the evaluation of sensitive or specific classifiers, Pattern Recogn. Lett. 38 (1) (2014) 93–98.
[3] W. Zhang, X. Tang, T. Yoshida, Q. Wang, Text clustering using frequent itemsets, Knowl.-Based Syst. 23 (5) (2010) 379–388.
[4] K. Nigam, A.K. Mccallum, S. Thrun, T. Mitchell, Text classification from labeled and unlabeled documents using EM, J. Mach. Learn. 39 (2000) 103–134.
[5] F.G. Cozman, I. Cohen, M.C. Cirelo, Semi-supervised learning of mixture models, in: Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), 2003.
[6] R. Dara, S. Kermer, D. Stacey, Clustering unlabeled data with SOMs improves classification of labeled real-world data, in: Proceedings of the 2002 International Joint Conference on Neural Networks, 2002, pp. 2237–2242.
[7] A. Demirez, K. Bennett, M. Embrechts, Semi-supervised clustering using genetic algorithms, in: Proceedings of Artificial Neural Networks in Engineering (ANNIE-99), 1999, pp. 809–814.
[8] X. Zhu, Semi-Supervised Learning Literature Survey, Technical Report No. 1530, Computer Sciences, University of Wisconsin-Madison, 2006.
[9] S. Basu, M. Bilenko, J.R. Mooney, Comparing and unifying search-based and similarity-based approached to semi-supervised clustering, in: Proceedings of the ICML-2003 Workshop on the Continuum from Labeled to Unlabeled data in Machine Learning and Data Mining Systems, 2003, pp. 42–49.
[10] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF net, and curve detection, IEEE Trans. Neural Netw. 4 (4) (1993) 636–649.
[11] P. D'Urso, L.D. Giovanni, M. Disegna, R. Massari, Bagged clustering and its application to tourism market segmentation, Expert Syst. Appl. 40 (12) (2013) 4944–4956.
[12] S. Tan, An effective refinement strategy for KNN text classifier, Expert Syst. Appl. 30 (2) (2006) 290–298.
[13] W. Zhang, T. Yoshida, X. Tang, Text classification based on multi-word with support vector machine, Knowl.-Based Syst. 21 (8) (2008) 879–886.
[14] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning Internal Representations by Error Propagation. Parallel Distributed Processing, Exploitations in the Microstructure of Cognition, MIT Press, Cambridge, MA, 1986. pp. 318–362.
[15] W. Zhang, Y. Yang, Q. Wang, Using Bayesian regression and EM algorithm with missing handling for software effort prediction, Inf. Softw. Technol. 58 (2015) 58–70.
[16] I.W. Tsang, J.T. Kwok, P. Cheung, Core vector machines: fast SVM training on very large data sets, J. Mach. Learn. Res. 6 (2005) 363–392.
[17] X. Luo, F. Liu, S. Yang, X. Wang, Z. Zhou, Joint sparse regularization based sparse semi-supervised extreme learning machine (S3ELM) for classification, Knowl.-Based Syst. 73 (2015) 149–160.
[18] D. Tian, Semi-supervised learning for refining image annotation based on random walk model, Knowl.-Based Syst. 72 (2014) 72–80.
[19] N. Lu, G. Zhang, J. Lu, Concept drift detection via competence models, Artif. Intell. 209 (2014) 11–28.