Information Sciences 357 (2016) 125–143

Contents lists available at ScienceDirect

Information Sciences

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

Dynamic Clustering Forest: An ensemble framework to efficiently classify textual data stream with concept drift☆

Ge Song[a,b], Yunming Ye[b,*], Haijun Zhang[b], Xiaofei Xu[b], Raymond Y.K. Lau[c], Feng Liu[b]

[a] College of Mathematics And Informatics, South China Agricultural University, Guangzhou, China
[b] Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China
[c] Department of Information Systems, City University of Hong Kong, Hong Kong Special Administrative Region

ARTICLE INFO

Article history:
Received 11 October 2014
Revised 10 January 2016
Accepted 28 March 2016
Available online 12 April 2016

Keywords:
Clustering tree
Ensemble learning
Concept drift
Textual stream

ABSTRACT

Textual stream mining with the presence of concept drift is a very challenging research problem. Under a realistic textual stream environment, it often involves a large number of instances characterized by a high-dimensional feature space. Accordingly, it is computationally complex to detect concept drift. In this paper, we present a novel ensemble model named, Dynamic Clustering Forest (DCF), for textual stream classification with the presence of concept drift. The proposed DCF ensemble model is constructed based on a number of Clustering Trees (CTs). In particular, the DCF model is underpinned by two novel strategies: (1) an adaptive ensemble strategy to dynamically choose the discriminative CTs according to the inherent property of a data stream, (2) a dual voting strategy that takes into account both credibility and accuracy of a classifier. Based on the standard measure of Mean Square Error (MSE), our theoretical analysis demonstrates the merits of the proposed DCF model. Moreover, based on five synthetic textual streams and three real-world textual streams, the results of our empirical tests confirm that the proposed DCF model outperforms other state-of-the-art classification methods in most of the high-dimensional textual streams.

1. Introduction

The recent trend in delivering emails, publishing blogs, establishing chatting rooms and forums on the Internet have led to the generation of a huge number of dynamic textual streams. The underlying characteristics of these textual streams pose some serious challenges of effectively classifying the dynamic textual streams. First, the concepts embedded in a data stream will change over time. This characteristic is referred to as concept drift, which requires the adaptation of a classifier with respect to the up-to-dated concepts. For instance, the topical interest of a reader may change over time after s/he has read a large number of online news with diversified topics on the Internet. This phenomenon motivates us to develop adaptive learning models to capture readers' evolving topical interests. Second, a textual stream usually consists of a large number of objects (instances), and these objects are characterized by a high-dimensional feature space (e.g., the news topics referred

☆ Fully documented templates are available in the elsarticle package on CTAN.

* Corresponding author. Tel.: +86 75526033008.
E-mail address: yeyunming@hit.edu.cn, yym@hitsz.edu.cn (Y. Ye).

http://dx.doi.org/10.1016/j.ins.2016.03.043
0020-0255/

to in a textual stream are described by a large vocabulary). Unfortunately, most of the existing data stream classification methods fail to tackle textual streams due to their high-dimensional feature spaces [26].

Many models have been proposed to deal with textual streams. Among them, the ensemble classification method that aims at combining the predictions of individual classifiers to form a final classification decision is a promising approach [5]. Under the ensemble framework, each chunk of a textual stream is treated as a sub-model to train a classifier, and then these well-trained sub-models are combined to predict the labels of incoming instances. However, existing ensemble methods cannot provide a satisfactory solution to tackle the task of textual stream classification because of the following open questions [31]:

1. How to adaptively select a suitable number of sub-models such that outdated concepts can be removed?
2. How to combine sub-models to generate an optimal global prediction?

In this paper, we propose a new ensemble approach named, Dynamic Clustering Forest (DCF), for the classification of textual streams. DCF aims at effectively combining a number of Clustering Trees (CTs) [28] by using a principled way. It is worth noting that CT is a clustering-based classification algorithm [18]. CT is suitable for classifying large, high dimensional, and sparse textual data with many classes [18]. For the proposed DCF model, we employ CT as a sub-classifier for classifying textual streams. First, we train a set of CTs by using some sequential chunks of a textual stream. Then, we dynamically select a suitable number of CTs and combine these CTs to accomplish the optimal classification performance. During the combination process, we exploit two ensemble strategies: an adaptive ensemble strategy and a dual voting strategy. For the adaptive ensemble strategy, a threshold, which is defined according to the accuracy weight of a CT, is applied to determine whether the CT is too "old" to classify a new concept. The threshold is estimated by using the average (or minimum) prediction accuracy of each CT in the model, rather than using the random prediction accuracy employed by most of the existing models [12,29]. For the dual voting strategy, a credibility weight is introduced to each testing sample. This credibility weight enables us to determine whether a CT is "credible" enough to classify the testing sample. The estimation of the credibility weight relies on the similarity between the testing sample and the centroid of the cluster that this testing sample belongs to.

To verify if the proposed strategies are sound or not, we conducted a theoretical analysis of the DCF model in terms of the standard measure of Mean Square Error (MSE). Moreover, we performed empirical tests against the DCF model by comparing the performance of the DCF model with that of seven state-of-the-art ensemble models available on the Massive Online Analysis (MOA) platform [6] in classifying several synthetic and real-world textual streams. Our experimental results confirm that the DCF model demonstrates promising performance in terms of average accuracy and plotting accuracy for high-dimensional textual stream classification tasks.

The rest of the paper is organized as follows. In Section 2, we discuss the existing research work which is related to our study. We then introduce the preliminaries and the background information about textual stream classification with concept drift in Section 3. In Section 4, we present an overview of the proposed DCF framework, followed by the illustration of the proposed adaptive ensemble strategy and the dual voting strategy of the DCF model in Section 5. In Section 6, we report a theoretical analysis of the inherent properties of the proposed DCF model, followed by a description of the empirical tests against the DCF model and some state-of-the-art ensemble models in Section 7. We then discuss the characteristics of the proposed algorithm by referring to our experimental results in Section 8. Finally, we offer concluding remarks and suggest future directions of our research work.

2. Related work

Our approach is related to two main research areas, namely data stream classification with concept drift and text mining. We briefly describe related research work in the following sub-sections.

2.1. Data streams with concept drift

Large amount of data and drifting concepts are two main features of a data stream. These inherent challenges of data streams lead to the development of two common methods to classify data streams: an incremental mining (IM) method, and an ensemble learning (EL) method.

The IM method revises and refines a single model continuously when new data arrive [29]. However, most existing IM algorithms are not efficient because the corresponding models must be updated frequently. In addition, these algorithms are not effective to handle drifting concepts, especially for recurring drifting concepts.

EL approach [10,31] is another promising approach for data stream mining. We summarize three popular ensemble strategies according to existing ensemble methods for data stream classification with concept drift reported in literature[15].

(1) *Ensemble approach based on resampling and adaptive sliding window (Adwin)*: Resampling is a technique that reuses (or selects) data, adaptively reweights and combines sub-models to improve classification performance. Some popular resampling methods include bagging and boosting [3]. But traditional resampling methods cannot deal with dynamic data stream classification. To solve the drifting concept problem, Bifet et al. [3] have proposed a method named, Adwin, which constructs a sliding window with varying size to choose a suitable amount of training data for learning

new concepts. Many resampling models, for example, bagging and boosting, rely on Adwin [5]. OzaBagAdwin is an online bagging method, which applies the Adwin approach to detect concept drift. When a new concept is detected, the sub-classifier that performs the worst will be replaced by a new one of the ensemble structure.

(2) *Building ensemble frameworks by updating sub-models' weights*: The ensemble approach for data stream classification that only updates the weights of sub-models is called racing approach [16]. Popular racing approaches include weighted majority algorithm [14] and mixture of experts [11], to name just a few. These approaches frequently verify each sub-model. For each verification race, the weights of sub-models are updated according to their prediction accuracies. However, this method may fail to handle the data stream with concept drift because the weight of a sub-model that represents a new concept may not correctly estimated.

(3) *Building ensemble frameworks by updating the whole structure*: This approach aims at changing the ensemble structure to adapt a new concept by discarding old sub-models. Selecting the sub-models that need to be dropped is essential. A number of strategies has been proposed [1,22,23,29] to solve this problem. Wang et al. [29] have proposed a popular method called the Accuracy Weight Ensemble (AWE) framework. AWE evaluates all sub-models and replaces the stale classifiers with the new ones. This algorithm maintains a fixed number of sub-models (the top k weighted sub-models) to participate in prediction. However, for practical applications, k should be revised with respect to concept drift. Another ensemble method, Accuracy Update Ensemble (AUE) [8] relies on a structure, which changes not only the weights of sub-models but also the current feature distributions. Nevertheless, it is still a fixed-window ensemble method.

2.2. Textual stream with concept drift

When compared to data stream classification, textual stream classification is more difficult because a textual stream is characterized by a high-dimensional and sparse feature space.

There are only a few EL approaches for textual stream classification with concept drift [12,13,17,30,32]. Katakis et al. have proposed a Conceptual Clustering and Prediction (CCP) [13] framework to classify textual stream with recurring concept. They construct incremental classifiers and update them when new training samples arrive. A new classifier will be constructed when a new concept is detected. But each concept has only one incremental classifier. In this sense, it is not an ensemble approach. In addition, PU Learning by Extracting Likely positive and negative micro-Clusters (LELC) [17], which can be seen as a semi-supervised learning, aims to solve positive and unlabeled textual stream problems. Voted LELC [21] based on LELC is also proposed. This method assigns a voting score to each representative document. It focuses on active learning to label unknown textual data, while it overcomes the drawbacks of AWE to some extent. A similar approach for one-class classification of textual streams with concept drift is proposed by Zhang et al. [32].

2.3. Characteristics of our approach

Our new dynamic ensemble approach, DCF, which includes an adaptive ensemble strategy and a dual voting strategy, is proposed to handle textual streams with concept drift. Apart from adaptively selecting sub-classifiers and revising the number of selected sub-models in accordance with the evolving concepts, DCF is different from other traditional ensemble approaches in the following aspects:

(1) When concept drift occurs, the accuracy weight is not sufficient to judge which sub-classifier is suitable to classify the new concept. Under such a circumstance, acquiring rich information about new concepts is important. Accordingly, DCF utilizes more information from a testing textual stream by introducing a credibility weight. The proposed method aims to determine whether a CT is credible for classifying the current testing sample or not.

(2) In comparison with most traditional ensemble strategies (e.g., AWE and AUE) that are developed based on the accuracy (or MSE) weight alone, our voting strategy relies on both the accuracy weight and the credibility weight. Both our theoretical analysis and our empirical tests show that the performance of the resulting ensemble model is improved by taking into account both factors.

3. Background

In this section, we formally define the notions of concept drift and clustering tree (CT).

3.1. Concept drift

Formally, let the distribution of textual data be $P_t(\mathbf{x}, y)$ at the tth time stamp in a textual stream, and let the distribution of textual data be $P_{t+1}(\mathbf{x}, y)$ at the $(t+1)$th time stamp, where $\mathbf{x}$ is the feature vector of an instance and y is the true class label of this instance. Under the same loss function [9], if $P_t(\mathbf{x}, y) \neq P_{t+1}(\mathbf{x}, y)$, a concept drift has taken place. For example, the concept of fashion news is changed with a user's varying tastes over time. If the rate of changes is slow over time, we define these changes as gradual (incremental) drift, e.g., a user's taste may change gradually. In contrast, if the rate of changes is abrupt and irreversible, we define these changes as sudden drift, e.g., seasonal changes with respect to the calendar.

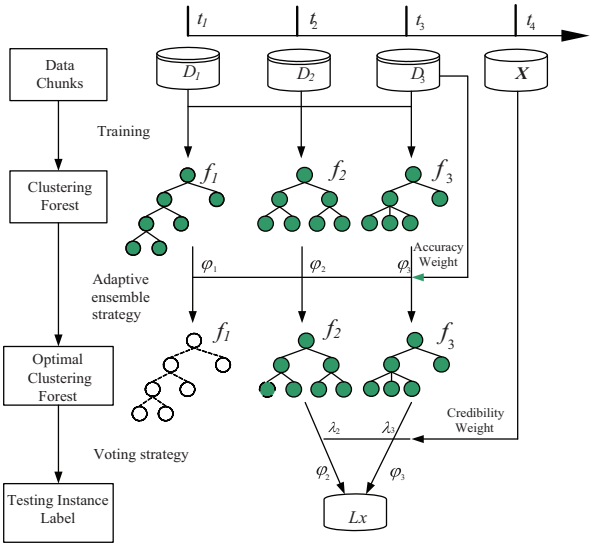


Fig. 1. Framework of Dynamic Clustering Forest (DCF).

3.2. Clustering Tree (CT)

In order to deal with a high-dimensional textual stream, we choose CTs [18,28] as the sub-classifiers of DCF. The CT algorithm is an approach that combines the decision tree and clustering algorithms. Not only does CT perform well for high-dimensional data and non-liner classification, but it also alleviates the problem of over-fitting.

In order to decide whether the node in CT needs to be split or not, the purity of a cluster Ω is defined by the number of samples of the most frequent class in the cluster [18]:

$$purity_{\Omega} = \frac{1}{|\Omega|} \max \left| x_{c_k,\Omega} \right|, \tag{1}$$

where $|\Omega|$ is the number of samples in the cluster Ω, the sample $x_{c_k,\Omega}$ belonging to class c_k is divided into the cluster Ω, that is $x_{c_k,\Omega} = \{x | x \in c_k \wedge x \in \Omega\}$. A CT can be easily generated according to the following steps: (1) generation of nodes in a CT: training samples are partitioned into several clusters (nodes) by calling a clustering algorithm recursively (e.g., k-means); (2) the ending condition of recursion: continue to split nodes recursively if the class-purity of a cluster is not higher than a predefined threshold; otherwise, the partitioning process ends [18].

For the proposed framework, the main computational apparatus of the clustering algorithm in CT is the similarity relation between a sample and the centroid of a cluster that this sample belongs to, and it is given by [28]:

$$sim(\mathbf{x}, C_k) = \frac{\sum_{r=1}^{R} x_r c_{k,r}}{\sqrt{\sum_{r=1}^{R} x_r x_r \cdot \sum_{r=1}^{R} c_{k,r} c_{k,r}}}, \tag{2}$$

where $\mathbf{x}$ is the feature vector of an instance, x_r is the rth component of feature vector $\mathbf{x}$. $C_k = (c_{k,1}, c_{k,2}, \ldots, c_{k,R})$ is the feature vector of the kth cluster's centroid. The component of the kth cluster's centroid (called $c_{k,r}$) is defined by [28]:

$$c_{k,r} = \mu_{k,r} \left(\sum_{k=1}^{K} \mu_{k,r} \log \frac{\sum_{k=1}^{K} \mu_{k,r}}{K} - \sum_{k=1}^{K} \mu_{k,r} \log \mu_{k,r} \right), \tag{3}$$

where $\mu_{k,r}$ is the rth component of μ_k, and $\mu_k = (\mu_{k,1}, \mu_{k,2}, \ldots \mu_{k,R})$ is the mean of the kth cluster. The value of similarity defined in Eq. (2) ranges from 0 indicating orthogonality of two feature vectors, to 1 meaning exactly the same.

4. Framework of Dynamic Clustering Forest (DCF)

Fig. 1 gives an overview of our proposed DCF framework for classifying textual streams with concept drift. Basically, the proposed framework includes two main steps:

(1) A training step which is applied to build the optimal DCF. The framework first builds CTs to construct the original DCF, then adaptively select some discriminative CTs to develop the optimal DCF.
(2) A testing step which is applied to identify the labels of the incoming testing instances. It is worth pointing out that the testing instances are classified by using a dual voting strategy.

For clarity, we summarize the proposed DCF framework as follows:

(1) Build CTs for the original DCF. The assumption is that only CTs within recent time stamps are used to classify new testing instances, and "old enough" CTs should be discarded. At the tth time stamp, a training chunk D_t is available to create a new CT, f_t. Then, the most recent $M_{\max}$ number of CTs are used to construct an initial clustering forest. That is, if $t \leq M_{\max}$, all of the existing CTs are selected to build the initial clustering forest. However, if $t > M_{\max}$, only CTs $(f_{t-M_{\max}+1}, f_{t-M_{\max}+2}, \ldots, f_t)$ are applied to build the initial clustering forest.
(2) Adaptively select the CTs to construct the optimal DCF. To accomplish the optimal DCF, an accuracy weight of each CT is computed. Then, our framework selects the CTs based on an adaptive ensemble strategy to construct the optimal DCF.
(3) Classify testing instances by using the optimal DCF. For each testing instance $\mathbf{x}$, the framework first computes a credibility weight for each selected CT. Then, a dual voting strategy is applied to select CTs for generating an ensemble prediction result.

Here, it is worth pointing out that the proposed DCF contains the following key strategies:

(1) Adaptive ensemble strategy: this strategy aims to dynamically select "good enough" experts (i.e., CTs) to build the optimal DCF. Unlike most existing models (e.g., AWE and AUE) where the number of sub-models is fixed, the number of CTs in our model varies according to the changes of concepts. To accomplish this goal, our framework employs the accuracy weight to select the discriminative CTs (see Section 5.2).
(2) Dual voting strategy: this strategy not only considers the global prediction capabilities of sub-models, which are estimated based on the accuracy weight, but also takes into account the credibility weight of each CT. This strategy implies that the proposed DCF model utilizes both the historical information of the training data, and the recent information of the testing data to improve classification performance (see Sections 5.1 and 5.3 for more details).

The algorithmic complexity of building DCF is linear with respect to the number of CTs (K). Given a training data chunk D_i at the ith time stamp, the computational complexity of training a CT is $O(mN\log_c N)$, where c is the number of classes in D_i, m is the number of dimensions, and N is the scale of a training chunk. As a result, the total algorithmic complexity of building DCF is $O(mKN\log_c N)$, where K is the number of CTs involved. In practical experiments, owing to termination conditions in growing a CT, the depth of a CT is usually less than $\log N$ in practice. It has been verified that the algorithmic complexity of ADCC [19], which is similar to CT, is linear with the number of samples N. Therefore, the algorithmic complexity of building DCF is also linear with respect to the size of training chunk N in real applications.

The computational cost of testing steps in DCF depends on the testing steps of a CT and computing the two weights, since each CT are executed in parallel. The complexity of testing steps in a CT is $O(mN_{test}\log_c N)$, where N_{test} denotes the size of a testing chunk. In addition, computing the credibility weight and the accuracy weight takes $O(N_{test})$ and $O(N)$ respectively. As a result, the algorithmic complexity of testing steps in DCF is approximately $O(mN_{test}\log_c N)$. Therefore, DCF is scalable for large high dimensional data streams, which will be confirmed by the experimental results in Section 7.

5. Ensemble strategies

In this section, we illustrate two main strategies of the proposed DCF model namely, an adaptive ensemble strategy and a dual voting strategy. We also define two crucial weights: the credibility weight and the accuracy weight of a CT.

5.1. Credibility weight

It is essential to utilize the information of testing data to improve the classification performance in a textual stream with concept drift. If a concept is changed, historical information (e.g., the accuracy and MSE computed based on historical classification performance) may become unreliable. Unfortunately, most of the existing weighted voting strategies such as DMW [14], AUE [8], and LELC [17] rely on historical information only. To consider the latest information extracted from a textual stream (i.e., testing data), we advocate a new concept, credibility weight, which measures the credibility level of a CT for each testing instance. The credibility weight is proportional to the similarity between a testing instance and the centroid of a leaf node that this instance belongs to.

Fig. 2 gives an example to describe the importance of the credibility weight. There are two leaf nodes associated with two classes (Class 1 and Class 2), respectively. In Fig. 2, a pentastar represents the centroid of a leaf node; Label 1 is marked by a triangle, and Label 2 is marked by a circle. The solid line is the decision boundary of the true data distribution between Class 1 and Class 2 at each time stamp; and the dotted line is the decision boundary between these two classes according to each classifier. In Fig. 2(a), we construct three classifiers, Classifier 1, Classifier 2, and Classifier 3, by the training instances (marked by triangles and circles) at the $(i-3)$th, $(i-2)$th, and $(i-1)$th time stamps, respectively; at the ith time stamp, where we suppose that the concept is not changed, and the testing instances can be classified into different classes by these three classifiers. For example, a testing instance (marked by a red circle), which belongs to Label 2, is classified by Classifiers 1, 2, and 3 (see Fig. 2(a)). The similarity between the testing instance and the centroid of the cluster that it belongs to with respect to Classifiers 1, 2, and 3, are denoted as d_1, d_2, and d_3, respectively. Classifier 1 decides that the testing instance belongs to Label 2, whilst Classifier 2 and Classifier 3 decide that it belongs to Label 1. Most of the CTs classify the testing

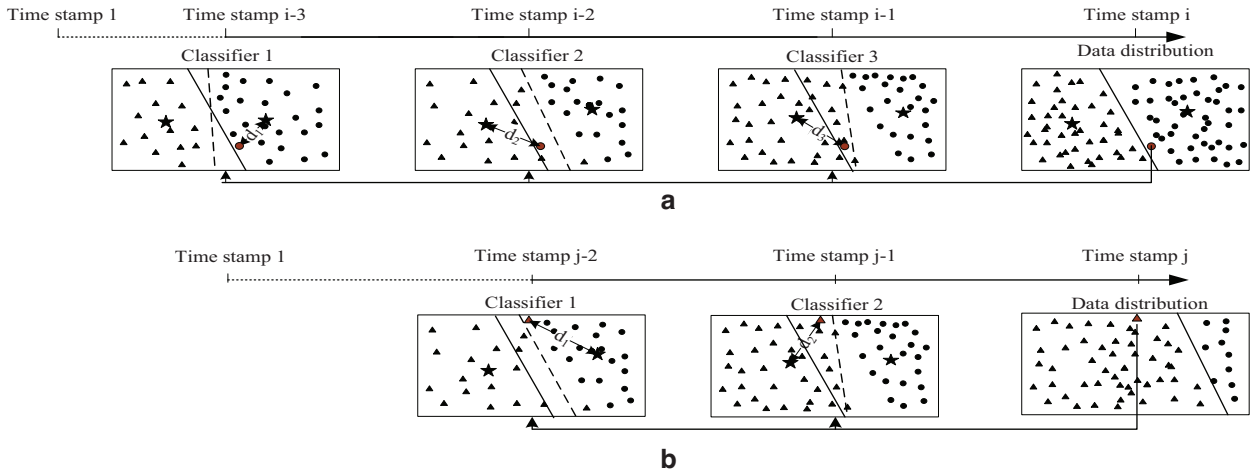


Fig. 2. An example for Dynamic Clustering Forest (DCF). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

instance belongs to Label 1. If we use the general majority voting strategy [1,22,23], the ensemble prediction result of this testing instance is Label 1. However, from Fig. 2, we can observe that $sim_1 > sim_2$, $sim_1 > sim_3$, which indicates that Classifier 1 is more credible than Classifier 2 and Classifier 3 for this testing instance in our DCF framework. According to this, we can predict that the label of this instance is Label 2, which is the true label of the testing instance.

Fig. 2 (b) describes the importance of the credibility weight for textual stream classification with concept drift. At the $(j-2)$th and $(j-1)$th time stamps, two classifiers (Classifier 1 and Classifier 2) are constructed by the training data. Suppose that these two classifiers build the optimal DCF based on the adaptive ensemble strategy (see Section 5.2) to classify testing instances at the jth time stamp. When a concept drift occurs at the jth time stamp, the data distribution is changed. However, the centroid and the decision boundary between two classes according to Classifier 1 and Classifier 2 are not changed, because these two classifiers have been constructed by the "old" textual instances at the $(j-2)$th, $(j-1)$th time stamps, respectively. When a testing instance (marked by a red triangle) arrives at this time, whose true label is Label 1, Classifier 1 assigns this testing instance with Label 2, while Classifier 2 infers that the testing instance should be assigned Label 1. We can observe that the Classifier 1 obtains the higher accuracy than the Classifier 2 based on the "old" decision boundary. So the ensemble prediction result of this testing instance may be Label 2 according to the existing weighted voting strategies [29].

In fact, it is not clear which classifier is more accurate for certain testing instances under the concept drift situation because we may not know if the concept has been changed. So the accuracy weight, which is used by most of the existing weighted voting strategies [29], may fail to classify the testing instance. However, for our DCF model, the credibility weight is computed based on the similarity information which is possible to improve the prediction result of the ensemble. From Fig. 2(b), we can see that the similarity $sim_1 < sim_2$ is established. This indicates that Classifier 2 is more credible than Classifier 1 for this testing instance. Hence, according to credibility weight, the ensemble tends to follow the prediction produced by Classifier 2, i.e. Label 1. As a result, the DCF model can assign the true label to the testing instance by considering a CT's credibility weight as well.

Based on the aforementioned example, we can see that the credibility weight of a CT is based on the similarity (i.e., $sim(\mathbf{x}_j, C_{k'})$, where $C_{k'}$ is the centroid of the cluster nearest to $\mathbf{x}_j$, $k' = (1, 2, \ldots, K)$, and $j = (1, 2, \ldots, N)$, and N is the number of the testing instances) between a testing instance and the centroid of a cluster that this instance belongs to. So, the credibility weight of each CT f_i for the testing instance $\mathbf{x}$ is defined by

$$\lambda_{i,j} = sim_i(\mathbf{x}_j, C_{k'}), \tag{4}$$

where $\lambda_{i,j}$ represents the credibility weight of a CT f_i in the optimal clustering forest f^E, for the testing instance $\mathbf{x}_j$, $sim(\mathbf{x}_j, C_{k'})$ is the similarity between the testing instance $\mathbf{x}$ and the centroid of the cluster nearest to $\mathbf{x}$ in CT f_i, and M is the number of CTs in the optimal clustering forest f^E. By definition, this credibility weight is able to evaluate the reliability of a CT for classifying a testing instance.

5.2. Adaptive ensemble strategy

As textual instances are continuously fed to our DCF system, We may need to construct an infinite number of CTs over time. However, it is dispensable to organize all these CTs for prediction. The fundamental questions are how many CTs should be selected to predict the testing instance and how to select these CTs. Adaptive ensemble strategy in our DCF framework is used to dynamically divide the original clustering forest into two categories: the "good enough" CTs and the

rest. At each time stamp, to obtain the accuracy weight, we estimate the accuracy of each CT using the latest training chunk. If we recognize that the concept is not changed by detecting the accuracy of the CTs, these CTs are regarded as the useful sub-classifiers to predict the testing instances. The number of CTs increases during the latest time stamps where concept drift does not occur. However, if the concept drift is detected, historical CTs are not useful to classify the new testing instances. Only a few of the "useful" trees should participate in the classification process. We then select these useful CTs according to the accuracy weight.

Algorithm 1 illustrates the adaptive ensemble strategy of the proposed DCF model. When a new textual chunk arrives, we

Algorithm 1 Adaptive ensemble strategy.

Output: The optimal DCF
Input: D_t: The data chunk at each time stamp
M: The number of CTs in DCF f^E
Step 1: At the tth time stamp:
Step 2: Create a new CT f_t using D_t;
Step 3: Obtain current original clustering forest f^E;
Step 4: For each f_i in f^E:
Step 5: Estimate the accuracy of f_i using D_t;
Step 6: Compute the accuracy weight ϕ_i of f_i;
Step 7: **If** $\phi_i < 0$:
Step 8: Discard the CT f_i update the model f^E;
Step 10: Update the model f^E;
Step 11: **If** $M = 0$:
Step 12: Select the tree with the maximum accuracy;

build a new CT by this chunk and add this CT to the original clustering forest. It is worth noting that the original clustering forest contains the latest M ($M \le M_{\max}$) CTs. We then estimate the accuracy of all the CTs by this new textual chunk. After computing the accuracy weight of each CT, we use all the selected CTs to classify the testing instance.

We use the following equation to compute the accuracy weight ϕ_i of each CT:

$$\phi_i = \frac{Acc_i - Acc_\theta}{\sum_{i=1}^{M}(Acc_i - Acc_\theta)}, \tag{5}$$

where Acc_i is the accuracy of each CT f_i, and the threshold Acc_θ is used to decide whether each CT is discarded or not. The threshold Acc_θ is derived by:

$$Acc_\theta = \begin{cases} \max(\min Acc_i - \tau, \frac{1}{2}) & \overline{Acc} - \min Acc_i \le \varepsilon \\ \max(\overline{Acc}, \frac{1}{2}) & \overline{Acc} - \min Acc_i > \varepsilon \end{cases}, \tag{6}$$

where ε is a predefined parameter to detect concept drift. Here, we detect concept drift by the difference between the minimum accuracy $\min Acc_i$ and $\overline{Acc}$ in the original clustering forest. If $\overline{Acc} - \min Acc_i \le \varepsilon$, it indicates that concept drift does not occur. So each CT is useful to classify the new testing instances. The CTs whose accuracies are more than the random prediction (1/2) in the original clustering forest can be selected to conduct the classification task [29]. τ is the minimum value to ensure $\forall \phi_i > 0$ in Eq. (6). Therefore, all the selected sub-models should be used in DCF. Otherwise, if $\overline{Acc} - \min Acc_i > \varepsilon$, it indicates that concept drift occurs, and some CTs fail to classify the new testing instances. Thus, only a few of the "useful" trees, whose accuracies greater than $\max(\overline{Acc}, \frac{1}{2})$ should be applied to the classification process.

The existing weighted voting methods, e.g. AWE and LELC, use random prediction (1/2) as the accuracy weight threshold with a fixed number of classifiers. However, the number of CTs in our strategy changes adaptively according to different conditions (with concept drift or without concept drift). Moreover, we use the threshold Acc_θ to discard the CTs with lower accuracies and to select the CTs that are more efficient for classification.

5.3. Dual voting strategy

Each CT in the optimal DCF delivers the local prediction result of the testing instance. We may design a dual voting strategy to integrate these local results into the final prediction of the ensemble. There are two important factors namely, a credibility weight and an accuracy weight for developing the voting strategy. The credibility weight reflects the credibility level of a CT for classifying the current testing instance. The accuracy weight makes full use of the new and historical information of a textual stream to select the suitable CTs for building the optimal DCF. For the ith CT, we first calculate a voting weight for the jth testing instance $\mathbf{x}_j$ by combining the credibility weight and the accuracy weight as follows:

$$v_i(\mathbf{x}_j) = \phi_i \lambda_{i,j}, \tag{7}$$

where ϕ_i is the accuracy weight of each CT f_i, and $\lambda_{i,j}$ is the credibility weight of the testing instance $\mathbf{x}_j$ in the CT f_i.

The output function $f_i(\mathbf{x}_j, l)$ of the ith CT is given by:

$$f_i(\mathbf{x}_j, l) = \begin{cases} 1 & L_{i,j} = l \\ 0 & L_{i,j} \neq l \end{cases}, \tag{8}$$

where $L_{i,j}$ is the output label of the testing instance $\mathbf{x}_j$ by the ith CT, and l represents all the possible labels of $\mathbf{x}_j$.
The ensemble output function of DCF $f^E(\mathbf{x}_j, l)$ is calculated by:

$$f^E(\mathbf{x}_j, l) = \sum_{i=1}^{M} v_i(\mathbf{x}_j) f_i(\mathbf{x}_j, l), \tag{9}$$

where $v_i(\mathbf{x}_j)$ is the voting weight, and $f_i(\mathbf{x}_j, l)$ is the output of the ith CT.
The ensemble prediction label L_j is set to the maximum value of $f^E(\mathbf{x}_j, l)$ according to the following equation:

$$L_j = \arg\max_l f^E(\mathbf{x}_j, l) = \arg\max_l \sum_{i=1}^{M} v_i(\mathbf{x}_j) f_i(\mathbf{x}_j, l). \tag{10}$$

Thus, the ensemble prediction function of DCF is given by:

$$f^E(\mathbf{x}_j, L_j) = f^E(\mathbf{x}_j, \arg\max_l f^E(\mathbf{x}_j, l)). \tag{11}$$

For clarity, we give a numerical example to illustrate the aforementioned voting strategy. In Fig. 2(a), at the ith time stamp, suppose that Classifiers 1, 2, and 3 are the three latest classifiers, whose accuracies are 0.8, 0.8, and 0.7, respectively. Let $\varepsilon = 0.15$ and $\tau = 0.1$. According to Eq. (6), $\overline{Acc} - \min Acc_i = 0.06 < 0.15 = \varepsilon$ and the accuracy threshold $Acc_\theta = \max(\text{minAcc}_i - \tau, \frac{1}{2}) = 0.6$. The Classifier 1's accuracy weight is $\phi_1 = 0.4$ by Eq. (5). Similarly, the accuracy weights of Classifiers 2 and 3 are ϕ_2=0.4, ϕ_3=0.2. When a testing instance $\mathbf{x}$ arrives (marked as red color), our model computes the credibility weights of these three classifiers by Eq. (4), that is $\lambda_1 = 0.5$, $\lambda_2 = 0.3$, $\lambda_3 = 0.3$. As can be seen from Fig. 2(a), the values of three classifiers' output functions are $f_1(\mathbf{x}, 1) = 0$, $f_1(\mathbf{x}, 2) = 1$, $f_2(\mathbf{x}, 1) = 1$, $f_2(\mathbf{x}, 2) = 0$, $f_3(\mathbf{x}, 1) = 1$, $f_3(\mathbf{x}, 2) = 0$. We can compute the ensemble output values, which are $f^E(\mathbf{x}, 1) = v_1(\mathbf{x}) f_1(\mathbf{x}, 1) + v_2(\mathbf{x}) f_2(\mathbf{x}, 1) + v_3(\mathbf{x}) f_3(\mathbf{x}, 1) = 0.2 \times 0 + 0.12 \times 1 + 0.06 \times 1 = 0.18$, $f^E(\mathbf{x}, 2) = 0.2 \times 1 = 0.2$. So, the final prediction label of the ensemble $L = \arg\max_l f^E(\mathbf{x}, l) = \arg\max_l (f^E(\mathbf{x}, 1), f^E(\mathbf{x}, 1)) = 2$ is determined according to the ensemble output value $f^E(\mathbf{x}, L) = f^E(\mathbf{x}, 2) = 0.2$.

Fig. 2 (b) illustrates the situation under concept drift at the jth time stamp. Because the textual data distribution is unknown at the jth time stamp, we estimate the two classifiers' accuracies as 0.8 and 0.7, respectively, by the historical textual data chunk at the $(j-1)$th time stamp. Thus, the accuracy weights are ϕ_1=0.67, ϕ_2=0.33. If we use the traditional weighted voting strategy [29] that is established according to these accuracy weights ($\phi_1 > \phi_2$), it will make a wrong decision. The ensemble will assign Label 2 to the testing instance because the data distribution is unknown at the jth time stamp and the accuracy weight adopted by the traditional weighted voting strategy at the $(j-1)$th time stamp is unreliable.

However, for the proposed dual voting strategy, suppose the credibility weights are computed as follows: $\lambda_1 = 0.3$, $\lambda_2 = 0.75$. Then, the voting weights are $v_1(\mathbf{x}) = 0.2$, $v_2(\mathbf{x}) = 0.25$. From Fig. 2(b), the values of the output function are $f_1(\mathbf{x}, 1) = 0$, $f_1(\mathbf{x}, 2) = 1$, $f_2(\mathbf{x}, 1) = 1$, $f_2(\mathbf{x}, 2) = 0$, and the ensemble output values are $f^E(\mathbf{x}, 1) = 0.25$, $f^E(\mathbf{x}, 2) = 0.2$. Accordingly, we can have $L = \arg\max_l f^E(\mathbf{x}, l)$=1 and $f^E(\mathbf{x}, L) = f^E(\mathbf{x}, \arg\max_l f^E(\mathbf{x}, l))$=0.25. When compared to the existing weighted voting strategies [14,29], this example shows that the proposed DCF model is more effective under concept drift.

6. Theoretical analysis

In this section, we conduct a theoretical analysis against the DCF model with respect to the Mean Square Error (MSE) measure. We prove that the MSE achieved by the voting strategy of DCF (see Section 5.3) is smaller than those delivered by two other traditional ensemble strategies, i.e. the majority voting strategy [1,22,23], and the weighted voting strategy [14,29], respectively.

First, let us define two parameters α_i and $\beta_{i,j}$ for the testing instance $\mathbf{x}_j$ in the ith CT. Given the ensemble output function $f^E(\mathbf{x}_j, l)$ and the output function of the ith CT $f_i(\mathbf{x}_j, l)$, we have:

$$f^E(\mathbf{x}_j, l) = \sum_{i=1}^{M} \alpha_i \beta_{i,j} f_i(\mathbf{x}_j, l), \tag{12}$$

where $\sum_{i=1}^{M} \alpha_i = 1$, $\sum_{i=1}^{M} \beta_{i,j} = 1$.
Based on Eq. (12), the prediction label L_j of the testing instance $\mathbf{x}_j$ depends on the maximum of $f^E(\mathbf{x}_j, l)$ as shown by:

$$L_j = \arg\max_l f^E(\mathbf{x}_j, l) = \arg\max_l \sum_{i=1}^{M} \alpha_i \beta_{ij} f_i(\mathbf{x}_j, l). \tag{13}$$

Thus, the output function of DCF is given by:

$$f^E(\mathbf{x}_j, L_j) = f^E(\mathbf{x}_j, \arg\max_l f^E(\mathbf{x}_j, l)). \tag{14}$$

We then define the error function of the ith CT:

$$e_i(\mathbf{x}_j, L_j) = f_i(\mathbf{x}_j, L_j) - f_i(\mathbf{x}_j, y_j), \tag{15}$$

where y_j is the true label of $\mathbf{x}_j$. We can rewrite $f^E(\mathbf{x}_j, L_j)$ as follows:

$$\begin{aligned} f^E(\mathbf{x}_j, L_j) &= \sum_{i=1}^{M} \alpha_i \beta_{i,j} f_i(\mathbf{x}_j, L_j) \\ &= f^E(\mathbf{x}_j, y_j) + f^E(\mathbf{x}_j, L_j) - f^E(\mathbf{x}_j, y_j) \\ &= f^E(\mathbf{x}_j, y_j) + \sum_{i=1}^{M} \alpha_i \beta_{i,j} e_i(\mathbf{x}_j, L_j). \end{aligned} \tag{16}$$

On the other hand, we define the symmetric correlation matrix Cor_{mn} as $Cor_{mn} = E(e_m(\mathbf{x}_j, L_j)e_n(\mathbf{x}_j, L_j))$, where $e_m(\mathbf{x}_j, L_j)$ and $e_n(\mathbf{x}_j, L_j)$ are the error functions of the mth CT and the nth CT, respectively. The MSE σ^2 of $f^E(\mathbf{x}_j, L_j)$ is given by:

$$\sigma^2 = \sum_{m,n}^{M} \alpha_m \beta_{n,j} \alpha_m \beta_{n,j} Cor_{mn}. \tag{17}$$

Likewise, for the traditional weighted voting strategy [29], we define its output function $f^E(\mathbf{x}_j, L_j)$ as follows:

$$\begin{aligned} f^E(\mathbf{x}_j, l) &= \sum_{i=1}^{M} w_i f_i(\mathbf{x}_j, l) \\ s.t. \sum_{i=1}^{M} w_i &= 1, \end{aligned} \tag{18}$$

where w_i is the weight of the ith CT.

Then, the ensemble output function $f^E(\mathbf{x}_j, L_j)$ of the traditional weighted voting strategy can be rewritten as follows:

$$\begin{aligned} f^E(\mathbf{x}_j, L_j) &= \sum_{i=1}^{M} w_i f_i(\mathbf{x}_j, L_j) \\ &= f^E(\mathbf{x}_j, y_j) + f^E(\mathbf{x}_j, L_j) - F^E(\mathbf{x}_j, y_j) \\ &= f^E(\mathbf{x}_j, y_j) + \sum_{i=1}^{M} w_i e_i(\mathbf{x}_j, L_j). \end{aligned} \tag{19}$$

Obviously, the MSE σ'^2 of the traditional weighted voting strategy is given by:

$$\sigma'^2 = \sum_{m,n}^{M} w_m w_n Cor_{mn}. \tag{20}$$

Theorem 1. $\exists \alpha_i \ \exists \beta_{i,j}$, $\sigma^2 \leq \min \sigma'^2$.

Assuming that the variances of different classifiers are independent in DCF, and the traditional weighted voting strategy is defined by the symmetric correlation matrix $Cor_{mn} = 0$ $(m \neq n)$ [2].

For the traditional weighted voting strategy, if we set the accuracy weight to $w_i = \frac{1}{\Omega_i^2} / \sum_{i=1}^{M} \frac{1}{\Omega_i^2}$, we can achieve the minimum value of σ'^2, where Ω_i^2 is the variance. The minimum of σ'^2 is given by [2,25]:

$$\min \sigma'^2 = \sum_{m,n}^{M} w_m w_n Cor_{mn} = \sum_{i}^{M} \Omega_i^{-2} / \left(\sum_{i}^{M} \Omega_i^{-2} \right)^2 = \left(\sum_{i,j}^{M} \Omega_i^{-2} \right)^{-1}. \tag{21}$$

For comparing σ^2 and σ'^2, suppose that α_i is equal to w_i as defined by:

$$\alpha_i = w_i = \frac{1}{\Omega_i^2} / \sum_{i=1}^{M} \frac{1}{\Omega_i^2}. \tag{22}$$

From Eq. (22), α_i is reversely proportional to the variance Ω_i^2, which has an inverse relationship with the accuracy of the ith CT, i.e. Acc_i. In addition, Acc_i is proportional to the accuracy weight ϕ_i from Eq. (5). Hence, α_i is proportional to the accuracy weight ϕ_i. Moreover, we suppose that $\beta_{i,j}$ is referred to as the credibility weight and it is defined as follows:

$$\beta_{i,j} = \lambda_{i,j}, (0 \leq \lambda_{i,j} \leq 1). \tag{23}$$

Substituting α_i *and* $\beta_{i,j}$ *into Eq. (17), we have:*

$$\sigma^2 = \sum_{m,n}^{M} \alpha_m \beta_{m,j} \alpha_n \beta_{n,j} Cor_{mn} = \sum_{i}^{M} \beta_{i,j}^2 \Omega_i^{-2} / \left(\sum_{i}^{M} \Omega_i^{-2} \right)^2. \tag{24}$$

As $0 \le \lambda_{i,j} \le 1$, *we have* $\sigma^2 \le \min \sigma'^2$. *We define the ensemble output function* $f'^E(\mathbf{x_j}, l)$ *in the majority voting strategy. Likewise, we have* $f'^E(\mathbf{x}_j, L_j)$ *derived from:*

$$\begin{aligned} f''^E(\mathbf{x}_j, L_j) &= \frac{1}{M} \sum_{i=1}^{M} f_i(\mathbf{x}_j, L_j) \\ &= f''^E(\mathbf{x}_j, y_j) + f''^E(\mathbf{x}_j, L_j) - f''^E(\mathbf{x}_j, y_j) \\ &= f''^E(\mathbf{x}_j, y_j) + \frac{1}{M} \sum_{i=1}^{M} e_i(\mathbf{x}_j, L_j). \end{aligned} \tag{25}$$

Theorem 2. $\exists \alpha_i, \exists \beta_{i,j}, \sigma^2 \le \min \sigma''^2$. *As each CT is independent, the minimum MSE* σ''^2 *is given by [25]:*

$$\min \sigma''^2 = \frac{1}{M^2} \sum_{m,n}^{M} Cor_{mn} = \frac{1}{M^2} \sum_{i}^{M} \Omega_i^{-2}. \tag{26}$$

We can easily prove the following:

$$M^2 \le \sum_{i}^{M} \Omega_i^{-2} \sum_{i}^{M} \frac{1}{\Omega_i^{-2}} \le \left(\sum_{i}^{M} \Omega_i^{-2} \right)^3 / \sum_{i}^{M} \beta_{i,j} \Omega_i^{-2}. \tag{27}$$

According to Theorem 1, α_i *is defined by Eq. (22), and* $\beta_{i,j}$ *is defined by Eq. (23), the value of MSE* σ^2 *in the DCF framework (see Eq. (24)) is smaller than the minimum value of MSE* $\min \sigma'^2$ *in the traditional weighted voting strategy (see Eq. (21)). Likewise, according to Theorem 2, the MSE produced by the voting strategy of DCF is smaller than those delivered by the majority voting strategy. These two Theorems guarantee that the DCF method is able to produce smaller MSE by taking into account the accuracy weight and credibility weight, respectively.*

7. Experiments

7.1. Datasets

The main problem of evaluating a classifier's performance for classifying textual streams with concept drift is the lack of benchmark datasets. Existing studies in this area usually use two types of datasets [20]: real-world datasets independently collected by researchers, and synthetic datasets.

7.1.1. Real-world dataset

At present, there are no public benchmark datasets for textual stream classification with concept drift. In fact, it is difficult to collect large-scale real-world textual streams because it costs huge manpower and time to label such a large number of instances. Thus, in order to evaluate the performance of DCF, we have compiled three real-world datasets which involve concept drift. The first one is the Spam Assassin stream, which is collected via naturally occurring processes. Another one with 500,000 instances is created by two real-world streams according to the sigmoid function [6]. The third one is an imbalanced real-world stream generated from the Spam Assassin stream.

We use the Spam Assassin Collection (Spam for short) [13] as the real-world textual stream. This textual stream consists of 9324 instances, and each instance contains 500 attributes. Essentially, the instances represent either spam and ham E-mails, and they are arranged according to a chronological order. In previous studies [13], the attributes of the ham and spam emails gradually change over time.

We construct a new large-scale textual stream (called Spam-Enron stream, S-E for short) based on the Spam Assassin Collection and another real-world textual stream namely, the Enron Email Dataset. The Enron dataset consists of 33,702 instances defined by 1545 attributes. This dataset is constructed according to the arrival order with 17,157 instances belonging to the spam class, and the remaining 16,545 instances belonging to the ham class. The sigmoid function is applied to formulate a weighted combination of two real-world datasets in order to characterize the target concepts under an evolving environment [6]. Based on this process, we construct a new textual stream with 500,000 instances, each of which contains 2044 attributes.

The Spam Assassin stream and the Spam-Enron stream are two imbalanced streams where the proportion of two classes is approximately 3:1. We add another imbalanced stream with 1187 instances in one class and 6937 instances in the other one to evaluate whether DCF is effective for classifying imbalanced textual stream. This stream (called imbalanced Spam Assassin stream, Imba. for short) is generated from the Spam Assassin stream by sampling a certain number of instances from the spam class. The ratio of the number of instances in two classes is approximately 6:1.

Table 1
Description of synthetic textual streams.

	20ng Grad.	20ng Sudd.	Reuters Grad.	Reuters Sudd.	Recurring
Size of training chunk	10,000	10,000	3000	3000	3000
Size of testing chunk	3500	3500	2450	2450	2450
Size of attributes	26,215	26,215	19,834	19,834	19,834
Number of concepts	4	2	3	2	2
Number of time stamps	40	20	30	20	40
Size of stream	540,000	270,000	163,500	10,900	327,000

7.1.2. Synthetic dataset

The 20 Newsgroups dataset[1] (20ng for short) and the Reuters dataset[2] are considered as textual sources, which are used to generate the textual streams with concept drift. The details of the generation process can be found in [20].

In responding to the changing rate of concepts, three typical kinds of concept drift are created in our study, i.e. gradual concept drift where concepts change slowly over time [16], sudden concept drift with abrupt changes respect to all the defined classes, and recurring concept drift. Here, our objective is to distinguish the relevant class from the irrelevant class under the conditions of gradual, sudden and recurring concept drift [16].

(1) *Synthetic stream with gradual drift*: for the 20ng gradual stream, 6 of the 20 topics are changed to the other class over every 10 time stamps. Likewise, a synthetic Reuters stream with gradual drift includes 30 time stamps. In every 10 time stamps, some topics are re-labeled for other classes. At the 11th time stamp, 37 topics with 2190 instances in the Reuters stream are changed to other concepts. This is a gradual concept drift as the rate of change is small (26.41%). However, at the 21st time stamp, concept drift occurs on 27 topics with 6037 instances; it is considered as a sudden drift with 72.8% concept change rate.
(2) *Synthetic stream with sudden drift*: the synthetic 20ng stream with sudden drift includes 20 time stamps. The topics in the first 10 time stamps are the same as those in the 20ng stream with gradual drift. At the 10th time stamp where the sudden drift occurs, we re-group all the topics to the other class. The synthetic Reuters stream with sudden drift is generated using the same way.
(3) *Synthetic stream with recurring drift*: two concepts of the synthetic recurring concept stream are generated from the Reuters dataset. The first concept occurs during the first 10 time stamps, and it is recurring during the third partition of the block of 10time stamps, while the second concept appears in the second partition of the block of 10 time stamps, and appears again in the forth partition of the block of 10 time stamps.

7.2. Evaluation procedure

The evaluation procedure determines which instances in a textual stream are used for training and which instances are used for testing. Assume that the testing chunk is basically similar to the most recent chunk in the period that a concept becomes stable. However, the distribution of the testing chunk is changed when concept drift occurs. Two main evaluation methods are considered in our study. The first method is the kind of batch evaluation, called *Holdout*. The arriving chunk is independently divided into the training chunk and the testing chunk at each time stamp. The second method, *Interleaved Chunk*, works by first testing the model with each incoming chunk and subsequently use that chunk for training [7]. Holdout procedure is used in all synthetic streams and S-E stream. While interleaved chunk is applied to other real-world streams. More details about the synthetic textual streams are shown in Table 1.

7.3. Evaluation measures

The average accuracy is usually used for evaluating the overall performance of a learning algorithm. But in an evolving environment, the plotting accuracy is of great importance. It represents the percentage of instances of the entire population at each time stamp. The calculations of the average accuracy and the plotting accuracy can be found in Ref. [20].

7.4. Comparative models

For a comparative evaluation, our DCF model has been compared with seven state-of-the-art ensemble methods which can perform similar textual stream classification. These methods are shown as follows:

(1) Accuracy Weighted Ensemble (AWE) [29]: a popular method that uses the weighted ensemble classifier.
(2) Accuracy Updated Ensemble (AUE) [8]: a method extends the AWE by using incremental sub-models and updating them according to the prediction accuracy.

[1] http://people.csail.mit.edu/jrennie/20Newsgroups .
[2] http://www.daviddlewis.com/resources/testcollections/reuters21578 .

Table 2
Results of different algorithms on all scenarios.

Algorithm	Textual stream							
	20ng-Grad.	Reuters-Grad.	20ng-Sudd.	Reuters-Sudd.	Spam	Recur-ring	S-E	Imbal-anced
DCF	**90.82**$_{(6.8)}$	**91.33**$_{(12.7)}$	**89.91**$_{(10.6)}$	**93.85**$_{(10.9)}$	**91.80**$_{(11.2)}$	**91.01**$_{(15.0)}$	**91.06**$_{(10.8)}$	**93.51**$_{(4.5)}$
AUE-RT	76.79$_{(4.0)}$	80.45$_{(9.8)}$	75.99$_{(14.1)}$	84.45$_{(18.0)}$	81.73$_{(5.5)}$	79.86$_{(11.7)}$	83.39$_{(14.1)}$	82.24$_{(2.1)}$
AUE-RF	**84.82**$_{(6.0)}$	**86.28**$_{(11.5)}$	79.3$_{(18.11)}$	**89.33**$_{(19.7)}$	**86.59**$_{(5.4)}$	**88.43**$_{(14.1)}$	**88.43**$_{(13.6)}$	**89.75**$_{(2.7)}$
AUE-SVM	67.46$_{(10.4)}$	71.59$_{(6.5)}$	80.13$_{(14.7)}$	82.05$_{(16.9)}$	**85.33**$_{(5.9)}$	70.08$_{(9.8)}$	83.02$_{(17.2)}$	82.27$_{(6.3)}$
AUE-HT	53.81$_{(30.6)}$	72.54$_{(18.6)}$	35.26$_{(27.8)}$	82.35$_{(17.8)}$	83.08$_{(6.8)}$	65.33$_{(17.6)}$	73.43$_{(22.4)}$	78.10$_{(11.3)}$
AUE-SGD	53.48$_{(37.3)}$	81.90$_{(10.2)}$	58.66$_{(29.4)}$	86.58$_{(14.9)}$	77.94$_{(6.9)}$	79.88$_{(11.7)}$	76.30$_{(21.0)}$	79.86$_{(3.5)}$
AUE-NBMT	**87.82**$_{(9.6)}$	**89.67**$_{(12.2)}$	**85.86**$_{(21.1)}$	86.63$_{(20.4)}$	**85.98**$_{(6.1)}$	**89.31**$_{(14.5)}$	88.17$_{(13.5)}$	**89.99**$_{(2.0)}$
AWE-RT	80.33$_{(6.4)}$	83.73$_{(11.8)}$	78.39$_{(16.2)}$	**86.67**$_{(18.9)}$	77.24$_{(5.5)}$	84.91$_{(13.7)}$	72.18$_{(30.8)}$	70.69$_{(6.1)}$
AWE-RF	**84.93**$_{(7.2)}$	**85.40**$_{(14.0)}$	**85.49**$_{(19.3)}$	**88.22**$_{(20.1)}$	82.45$_{(5.6)}$	**88.63**$_{(14.5)}$	78.49$_{(30.8)}$	75.54$_{(6.0)}$
AWE-SVM	67.43$_{(10.4)}$	71.59$_{(6.5)}$	**80.19**$_{(14.6)}$	82.05$_{(16.9)}$	81.11$_{(5.7)}$	70.08$_{(9.8)}$	73.20$_{(28.6)}$	71.38$_{(6.1)}$
AWE-HT	24.20$_{(30.6)}$	54.58$_{(7.1)}$	32.80$_{(31.9)}$	71.87$_{(21.4)}$	75.57$_{(5.9)}$	54.80$_{(8.4)}$	76.13$_{(27.5)}$	67.55$_{(9.4)}$
AWE-SGD	53.48$_{(37.3)}$	81.89$_{(12.5)}$	58.66$_{(29.4)}$	86.50$_{(15.0)}$	76.18$_{(5.5)}$	79.89$_{(11.7)}$	67.99$_{(28.4)}$	68.39$_{(5.3)}$
AWE-NBMT	**91.91**$_{(8.5)}$	**90.12**$_{(12.2)}$	**90.53**$_{(19.6)}$	**86.77**$_{(20.4)}$	84.50$_{(16.2)}$	**89.42**$_{(14.4)}$	80.56$_{(27.3)}$	78.27$_{(3.7)}$
LB-RT	45.50$_{(1.7)}$	60.39$_{(9.6)}$	50.33$_{(3.3)}$	73.72$_{(22.6)}$	64.68$_{(11.4)}$	64.16$_{(10.4)}$	80.84$_{(14.9)}$	73.42$_{(4.3)}$
LB-RF	45.50$_{(1.7)}$	54.93$_{(6.8)}$	50.33$_{(3.3)}$	71.94$_{(22.3)}$	63.47$_{(12.7)}$	55.16$_{(21.0)}$	86.59$_{(14.7)}$	84.27$_{(1.8)}$
LB-SVM	45.50$_{(1.7)}$	64.87$_{(7.1)}$	50.33$_{(3.3)}$	71.87$_{(22.2)}$	76.09$_{(11.1)}$	68.02$_{(10.4)}$	79.82$_{(23.9)}$	50.69$_{(7.4)}$
LB-HT	70.57$_{(34.2)}$	57.82$_{(7.3)}$	53.16$_{(3.3)}$	73.76$_{(20.6)}$	86.17$_{(7.2)}$	57.61$_{(6.2)}$	**91.12**$_{(11.0)}$	80.66$_{(12.1)}$
LB-SGD	64.98$_{(37.3)}$	75.47$_{(12.9)}$	67.33$_{(31.7)}$	84.31$_{(12.9)}$	78.70$_{(7.6)}$	76.97$_{(15.4)}$	80.51$_{(17.6)}$	77.76$_{(3.7)}$
LB-NBMT	61.59$_{(32.1)}$	82.23$_{(16.0)}$	54.70$_{(26.6)}$	85.70$_{(19.01)}$	**86.61**$_{(6.0)}$	75.78$_{(19.1)}$	**88.25**$_{(11.9)}$	**86.32**$_{(3.7)}$
OZA-RT	45.50$_{(1.7)}$	71.53$_{(11.8)}$	50.33$_{(3.3)}$	82.08$_{(19.1)}$	54.34$_{(11.9)}$	71.71$_{(15.4)}$	85.58$_{(11.4)}$	75.19$_{(13.7)}$
OZA-RF	45.50$_{(1.7)}$	68.71$_{(12.0)}$	50.33$_{(3.3)}$	82.41$_{(18.6)}$	67.80$_{(12.1)}$	72.25$_{(14.8)}$	84.56$_{(16.1)}$	79.39$_{(14.2)}$
OZA-SVM	45.50$_{(1.7)}$	55.31$_{(8.3)}$	50.33$_{(3.3)}$	71.87$_{(22.2)}$	70.88$_{(11.0)}$	54.12$_{(12.5)}$	84.70$_{(17.9)}$	78.28$_{(14.7)}$
OZA-HT	69.62$_{(34.7)}$	55.84$_{(7.3)}$	54.24$_{(1.8)}$	69.11$_{(8.4)}$	81.39$_{(8.3)}$	56.40$_{(9.6)}$	81.76$_{(15.3)}$	78.86$_{(11.8)}$
OZA-SGD	61.52$_{(37.3)}$	72.93$_{(14.0)}$	52.98$_{(35.9)}$	71.87$_{(21.4)}$	76.43$_{(7.6)}$	70.22$_{(14.6)}$	78.58$_{(18.6)}$	77.87$_{(3.8)}$
OZA-NBMT	66.81$_{(33.9)}$	65.38$_{(14.3)}$	53.72$_{(1.8)}$	77.70$_{(18.9)}$	86.38$_{(6.2)}$	65.13$_{(14.9)}$	87.88$_{(12.5)}$	**86.60**$_{(4.5)}$
CFIM	**89.91**$_{(7.7)}$	**89.82**$_{(12.6)}$	**89.29**$_{(19.8)}$	**90.31**$_{(9.7)}$	**91.42**$_{(11.1)}$	**89.63**$_{(13.8)}$	**89.32**$_{(10.8)}$	**91.60**$_{(5.9)}$
M3	44.97$_{(4.2)}$	50.82$_{(14.2)}$	47.81$_{(7.8)}$	72.59$_{(21.0)}$	76.68$_{(6.1)}$	48.39$_{(16.7)}$	79.92$_{(19.8)}$	78.31$_{(3.0)}$
ASHT	56.45$_{(7.8)}$	76.56$_{(13.2)}$	56.6$_{(7.0)}$	83.09$_{(17.7)}$	77.95$_{(8.8)}$	69.02$_{(11.3)}$	**95.39**$_{(9.2)}$	77.85$_{(13.0)}$

(3) Leveraging Bag (LB) [4]: a method combines the Adwin and the Leveraging Bagging by using Random Output Codes (ROC) .
(4) OzaBagAdwin (OZA) [5]: which uses the Adwin algorithm to detect and estimate the drift for providing the ensemble weights of the boosting method.
(5) Clustering Forest for IMbalanced textual stream (CFIM) [27]: an ensemble method based on accuracy weight to deal with the imbalanced textual stream.
(6) Modal Mixture Model (M3) [24]: a weighted majority ensemble (using Naive Bayes and Hoeffding Tree as basic learner) where model weights are updated on-line using Reinforcement Learning techniques.
(7) Adaptive-Size Hoeffding Tree (ASHT) [5]: a new bagging method derived from the Hoeffding Tree using trees of different sizes.

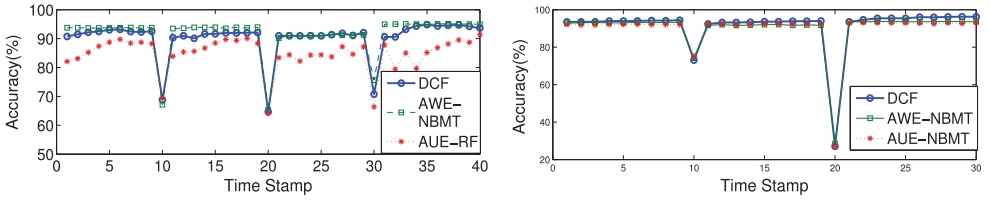
In our experiment, we choose Random Tree (RT), Random Forest (RF), libSVM (SVM for short), Multinomial Naive Bayes (NBMT), Hoeffding Tree (HT), SGD (Stochastic Gradient Descent, a gradient descent optimization method to minimize an objective function) as basic learners. All six sub-classifiers are used in AUE, AWE, LB, and OZA, respectively. Thus, 27 combined methods take part in our evaluation. The proposed DCF model employs CTs as the basic learners because it is relatively easy to estimate the credibility weight for this type of learner.

It is worth noting that it is not straightforward to estimate the credibility weight for basic learners such as RT, RF, SVM, NBMT, HT, and SGD because it is difficult to compute the centroid of each class in these learners. All these baseline methods have been implemented in the Massive Online Analysis (MOA) framework [6]. We adopted the cost function parameter $c = 1$ for SVM. The remaining parameters of basic learners were set by using the defaults in MOA. The maximum number of sub-classifiers in all ensemble models was set to $M_{\max} = 15$ in the S-E stream, and it was set to $M_{\max} = 5$ for the other streams. We set the parameters $\varepsilon = 0.4,\ \tau = 0.05,\ p = 1$ for the DCF model in our experiments.

7.5. Results

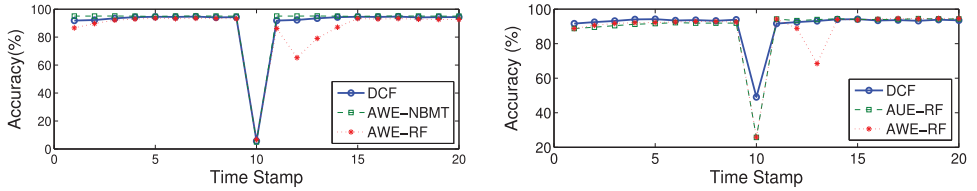
We evaluated the performance of all ensemble methods under different concept drift situations: synthetic gradual drift stream, synthetic sudden drift stream, synthetic recurring drift stream, and real-world stream under an imbalanced environment. The applied the measures averaged accuracy and plotting accuracy to our comparative evaluation.

We summarize the results of different approaches in terms of the average accuracy for all the streams in Table 2 (the standard deviation is shown in parenthesis). As observed from Table 2, DCF achieves the highest average accuracy in comparison to all the baseline methods for most of the textual streams, especially for the Spam stream. Compared with AWE-NBMT, DCF obtains a significant accuracy improvement in three real-world textual streams, but it achieves a slightly lower average



(a) 20ng gradual stream (b) Reuters gradual stream

Fig. 3. Plotting accuracies in the gradual streams.



(a) 20ng sudden stream (b) Reuters sudden stream

Fig. 4. Plotting accuracies in the sudden streams.

accuracy for the 20ng textual streams. In addition, although LB-HT and ASHT perform better than DCF for the S-E stream classification according to average accuracy, DCF outperforms LB-HT and ASHT for the 20ng and Reuters streams. CFIM is another baseline method with the highest average accuracy for nearly all the textual streams, but the average accuracy of CFIM is lower than that of DCF. The probable reason is that CFIM only utilizes the accuracy weight to tune the ensemble, while the proposed DCF model can take into account both credibility weight and accuracy weight.

7.5.1. Synthetic stream with gradual drift

Table 2 presents the average accuracies of all 25 methods for the gradual drift stream generated from 20ng. We observe that the average accuracy of DCF (90.82%) is slightly lower than that of AWE-NBMT (91.91%), but it is higher than that of the other 26 methods. Moreover, AUE and AWE perform better than the LB and the OZA methods. It seems that the LB and the OZA methods fail to cope with this gradual change. Compared with sub-learners, NBMT is considered as the best sub-classifier in the AUE and the AWE models.

Fig. 3(a) with the time stamps shown on the *x*-axis reveals the plotting accuracies of DCF, AWE-NBMT, and AUE-RF, which are considered as the best three methods for classifying the gradual drift stream generated from 20ng. When the gradual drift occurs at the 10th, 20th, and 30th time stamps, all of the three methods react to the changes with a considerable drop of plotting accuracy. For example, around 24% accuracy decrement is shown from the 9th time stamp to the 10th time stamp. After adjusting to the new concept, all these three methods achieve improvements of plotting accuracy gradually. However, the AWE-NBMT and the DCF methods still outperform the AUE-RF method. In comparison with the AUE-RF method, the plotting accuracy improvements of DCF reach 6.42%, 7.57%, and 2.75% at the 11th, 21st, and 31st time stamps, respectively. In addition, the increments of the plotting accuracies of DCF are 21.51% (26.28%, 19.83%) from the 10th (20th, 30th) time stamp to the 11th (21st, 31st) time stamp. During the periods with stable concepts, the AUE-RF method produces large performance fluctuations in terms of plotting accuracies.

More complex gradual drift is generated from the Reuters corpus. As shown in Table 2, DCF achieves the highest average accuracy (91.33%) over all the baseline methods; AWE-NBMT, CFIM and AUE-NBMT are the second best methods by achieving 90.12% accuracy and 89.82% accuracy, 89.67% accuracy, respectively. Fig. 3(b) describes the classification results of the gradual drift stream generated from the Reuters corpus, where the drifting concepts appear at the 10th, and the 20th time stamps. At the 10th time stamp, as the concept change is small (only around 2200 instances are changed to the other classes), DCF, AWE-NBMT, and AUE-NBMT take time to adapt to this concept drift, which causes a small drop in terms of plotting accuracy. At the 20th time stamp, we observe a large decrement of plotting accuracy since the ensembles need to cope with the concept drift. However, the proposed DCF model is effective to handle concept drift by rebuilding the model quickly, which leads to a high plotting accuracy in this period.

7.5.2. Synthetic stream with sudden drift

According to the experimental results depicted in Table 2, DCF achieves the second highest average accuracy for the 20ng sudden stream, only around 0.6% lower than AWE-NBMT. Meanwhile, the average accuracy achieved by DCF is 89.91% which is higher than the forth highest accuracy given by AUE-NBMT. The plotting accuracies of the tested methods in the 20ng sudden stream are shown in Fig. 4(a). All the methods have a large decrement of the plotting accuracy due to the concept drift occurring at the 10th time stamp. During the period of rebuilding classification models, the plotting accuracies obtained by the evaluating methods rise gradually. We observe that the accuracy curve of AWE-RF fluctuates considerably, especially

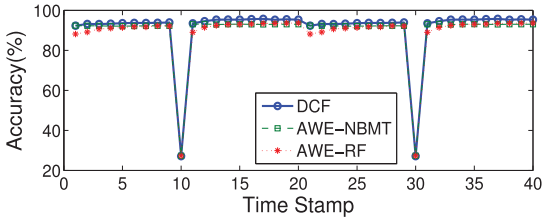


Fig. 5. Plotting accuracies in the recurring stream.

at the 12th time stamp. This result reveal that the effect of concept drift may influence the performance of the AWE-RF method even after the concept drift period. In contrast, DCF achieves relatively stable plotting accuracy after concept drift occurs. Thus, we can clearly distinguish the stable period from the drift period. Similar result is observed for the AWE-NBMT method in terms of plotting accuracy.

Likewise, DCF achieves the best performance for the Reuters sudden drift stream as illustrated in Table 2 and Fig. 4(b) because DCF is effective to deal with sudden drifts. After the drop of accuracies of all models at the 10th time stamp, their performance starts to improves due to the rebuilding of the corresponding classification models. However, the AWE-RF method fails to achieve consistent accuracy at the 13th time stamp as shown in Fig. 4(b).

7.5.3. Synthetic stream with recurring drift

In this subsection, we analyze the performance of DCF under the concept recurrence scenario. From the experiment results depicted in Table 2, we observe that DCF is still the best method to deal with the recurring concept drift as measured according to average accuracy. Fig. 5 shows the plotting accuracies of DCF, AWE-NBMT, and AWE-RF methods for the recurring concept stream where the first concept begins to reappear at the 21st time stamp, and the second concept reappears 10 time stamps later. At the 10th time stamp, the change between the first concept and the second one causes performance degradation of all three methods in terms of plotting accuracy. At the 21st time stamp, the performance of DCF and AWE-RF degrades with respect to plotting accuracy when the first concept is recurring. However, DCF achieves the higher plotting accuracy when compared with that of AWE-RF. The plotting accuracies of these two methods improve later on due to the successful learning of the new concept. However, AWE-NBMT fails to react to concept recurrence as evidenced by not having any performance improvement when concept recurrence occurs.

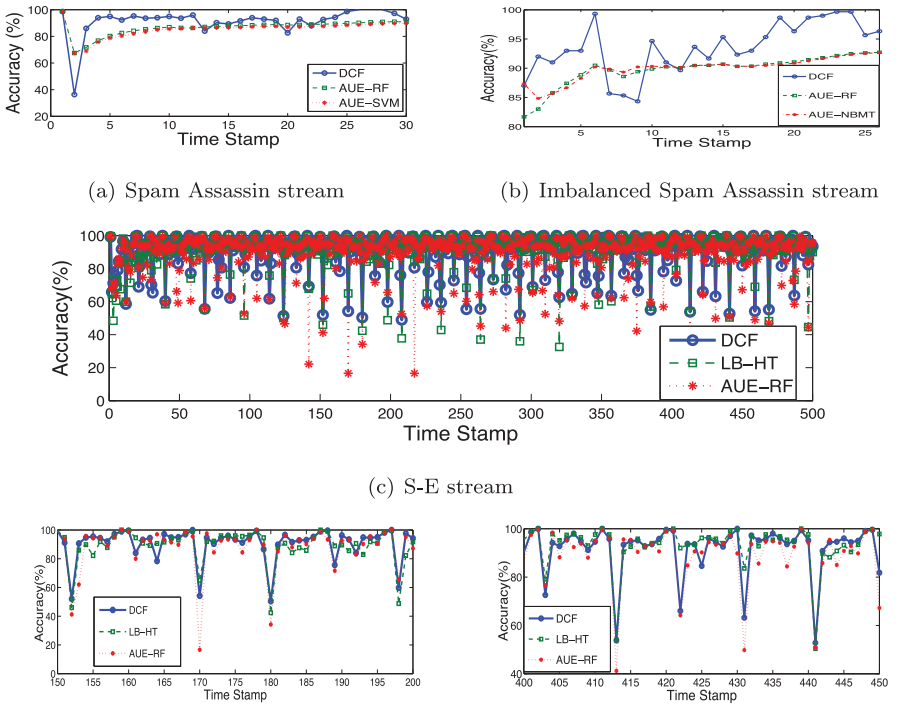
7.5.4. Real-world stream

In this subsection, we further investigate the performance of the various methods based on three real-world drift streams under n imbalanced classification environment.

The average accuracy achieved by each method based on three real-world datasets is summarized in Table 2. Compared with LB-NBMT, DCF achieves 5.19% accuracy improvement. As seen in Fig. 6(a), a large concept drift may have occurred at the 3rd time stamp. The plotting accuracy of DCF is lower than that of AUE-RF and AUE-SVM at this time stamp. The probably reason of such a performance degradation of DCF is explained as follows. The first two chunks contain only 4 ham instances, while the 3rd chunk contains nearly 200 ham instances. So, for each CT, the leaf node belonging to the ham class is constructed by only 4 instances at the first two time stamps. It cannot effectively represent the feature distribution of the ham class. The poor leaf node may cause the misclassification of the testing instances. Moreover, if the number of instances is too small to affect the correct calculation of purity, which decides whether the node needs to be split or not, a leaf node may not be generated to represent the ham class. Accordingly, the instances belonging to the ham class in the test chunk cannot be labeled correctly. This problem may cause the degradation of DCF's performance at this concept drift time stamp. At the 12th time stamp and the 20th time stamp, a small drop (around 10%) of plotting accuracy is conveyed by the DCF method. However, the same problem does not occur for AUE-RF and AUE-SVM. The possible reason is that DCF with 5 maximum CTs is slightly more sensitive to noises for the Spam Assassin drift stream.

The average accuracy of DCF in Table 2 is less than that of LB-HT (0.06%) and ASHT (4.33%) for the S-E stream. The three methods' (DCF, LB-HT, and AUE-RF) plotting accuracies for the S-E stream are presented in Fig. 6(c). Fig. 6(d) and (e) describe the variances of the local plotting accuracy of three methods for the S-E stream. It is noticed that DCF accomplishes the better performance with respect to plotting accuracy as the fluctuation of accuracy produced by DCF is smaller than those of the other methods. For AUE-RF, the plotting accuracy fluctuates in a range of [16.6%, 100%) while it is limited in the range of [49%, 100%) for DCF. The LB-HT method performs slightly better if a relatively small change of a concept occurs. For example, we can observe this in Fig. 6(d) during the period between the 171st stamp and the 181st stamp. However, LB-HT even falls down below the level of DCF when a drastic concept drift occurs. In the period of rebuilding the classification model (such as between the 153rd time stamp and 160th time stamp), the plotting accuracy of LB-HT grows slowly. This indicates that LB-HT is not so effective to learn new concepts, especially after a drastic concept drift. In contrast, DCF, with a smooth updating of classification components, is able to adapt to concept drift effectively.

The average accuracy for the imbalanced Spam Assassin stream is shown in Table 2. DCF achieves the best result (93.51%). The average accuracy of DCF is 3.76% higher than that of AUE-RF. The plotting accuracies of 3 methods for the imbalanced

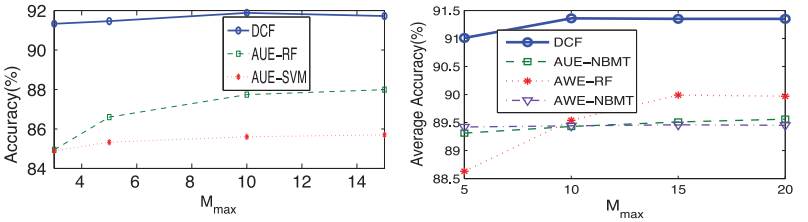


(a) Spam Assassin stream (b) Imbalanced Spam Assassin stream

(c) S-E stream

(d) Time stamps between 150th and 200th in S-E stream (e) Time stamps between 400th and 450th in S-E stream

Fig. 6. Plotting accuracies in the real-world streams.



(a) Spam Assassin stream (b) Recurring stream

Fig. 7. Average accuracies across different values of $M_{\max}$ in two streams.

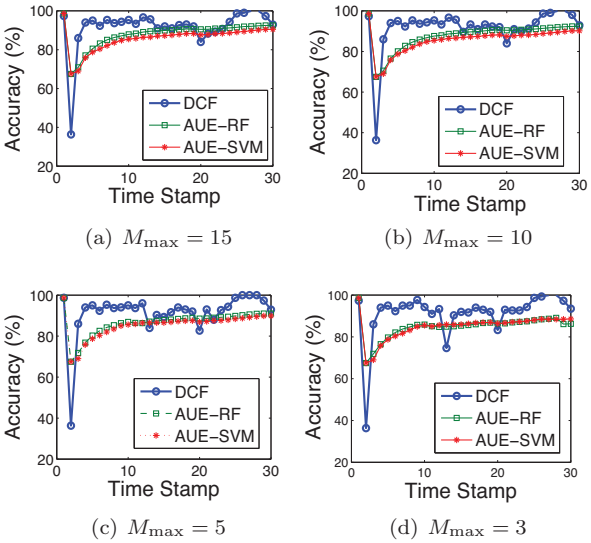
Spam Assassin stream in Fig. 6(b) show that the plotting accuracy curve of DCF is above other curves at most of the time stamps. However, the plotting accuracy of DCF appears to be more fluctuated when compared to that of other methods.

7.6. Parametric study

The parameter estimation of $M_{\max}$ is explained in this subsection. For almost all the ensemble methods, $M_{\max}$ is an important parameter to control the number of sub-classifiers. ε is a predefined parameter estimated according to the difference between the minimum accuracy $minAcc_i$ and $\overline{Acc}$ in the original clustering process to help selecting the suitable sub-classifiers. τ is the minimum value to ensure that the accuracy weight of each CT is greater than 0. The purity p is the measure to evaluate whether the cluster node should be split or not in the clustering tree.

To study the impact of $M_{\max}$ on the performances of all methods under testing, we repeat the experiments related to the Spam Assassin textual stream by setting $M_{\max}$ to different values, i.e. $M_{\max} = 15,\ 10,\ 5$ and 3. We select AUE-RF and AUE-SVM as the baseline methods because of their high average accuracy.

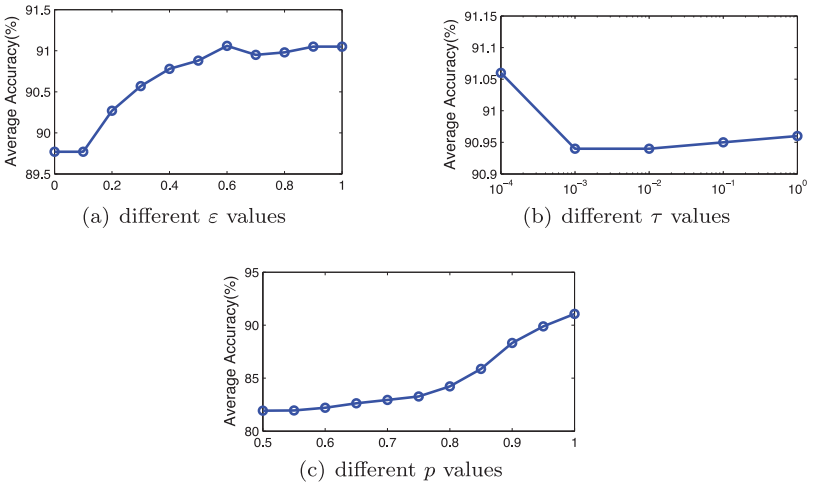
The results of DCF, AUR-RF and AUE-SVM with respect to four $M_{\max}$ values for the Spam Assassin stream are summarized in Fig. 7(a). We observe that DCF consistently performs better than the other two methods. As expected, AUE-RF and AUE-SVM achieve better average accuracy as the value of $M_{\max}$ increases. Especially, the accuracy of AUE-RF increases in a significant rate (around 3%) when compared to that of the other methods. However, the DCF's average accuracy increases by 0.39% only when the value of $M_{\max}$ increases from 3 to 15. This indicates that the classification performance of DCF is



(a) $M_{\max} = 15$ (b) $M_{\max} = 10$

(c) $M_{\max} = 5$ (d) $M_{\max} = 3$

Fig. 8. Plotting accuracies with different $M_{\max}$ values.



(a) different ε values (b) different τ values

(c) different p values

Fig. 9. Plotting accuracies with different parametric values.

more stable across different values of $M_{\max}$. Moreover, as seen in Fig. 7(a), the proposed DCF method achieves the best performance with $M_{\max} = 10$ instead of $M_{\max} = 15$ when AUE-RF and AUE-SVM achieve the best average accuracies. It indicates that the DCF method may achieve the best accuracy with a smaller value of $M_{\max}$ when compared to AUE-RF and AUE-SVM.

Fig. 8 describes the plotting accuracy of DCF in terms of four $M_{\max}$ values in the Spam stream. From results, the curve of plotting accuracy by setting $M_{\max} = 10$ seems more smooth than those by setting the other three $M_{\max}$ values. The bigger curve undulation is delivered by DCF with $M_{\max} = 3$. On the contrary, the curve of plotting accuracy shows a slight change when $M_{\max} = 5$, 10 and 15. Therefore, it implies that the choice of $M_{\max}$ is not crucial to obtain the optimal results as long as the value of $M_{\max}$ is not too small.

Fig. 7(b) shows the experimental results of DCF, AUE-NBMT, AWE-RF, AWE-NBMT with respect to average accuracy for the recurring stream when $M_{\max}$ is different. We observe that DCF consistently achieves the highest accuracy with different values of $M_{\max}$. The AUE model improves its average accuracy as the value of $M_{\max}$ increases. However, AWE-RF and AWE-NBMT achieve the best performance when $M_{\max} = 15$ is set, while DCF achieves the best average accuracy when $M_{\max} = 10$ is set. In addition, the average accuracies of all methods seem to be stable across different values of $M_{\max}$.

Fig. 9(a) shows the sensitivity of DCF for the S-E stream with different values of ε. The x-axis indicates a series of values of ε from 0 to 1, and the y-axis shows the percentage of average accuracy. Note that $\varepsilon = 0$ means only clustering trees with higher average accuracies participate in classification, while $\varepsilon = 1$ means that all the clustering trees participate in classification. We observe that the average accuracy varies slightly with ε ranging from 0 to 1. That is probably because we define the credibility weight for each test instance, which reduces the impact of incorrect classifier selection. Accordingly,

Table 3
Running time of different methods on all scenarios.

Algorithm	DCF	AUE-RT	AUE-RF	AUE-SVM	AUE-HT	AUE-SGD	AUE-NBMT	AWE-RT	AWE-RF	AWE-SVM	AWE-HT	AWE-SGD	AWE-NGMT	LB-RT
20ng Grad.	7.9E4	1.0E4	6.2E4	1.0E5	2.5E4	1.2E3	2.1E2	2.3E4	8.5E4	1.1E5	2.1E4	1.1E3	1.9E2	4.2E1
Reuters Grad.	5.6E3	1.5E3	6.8E3	4.5E3	6.9E3	1.2E2	1.6E1	1.7E3	6.2E3	4.4E3	8.8E3	1.1E2	3.1E1	1.8E2
20ng Sudd.	3.7E4	5.3E3	4.3E4	5.3E4	1.2E4	3.5E2	8.9E1	1.0E4	4.5E4	5.5E4	1.0E4	3.4E2	1.7E2	1.9E1
Reuters Sudd.	3.2E3	8.5E2	4.0E3	2.0E3	6.7E3	1.0E2	1.1E1	9.3E2	4.7E3	2.0E3	6.8E3	9.9E1	1.2E1	4.4E1
Spam	2.3E0	6.0E0	6.1E0	2.1E1	1.4E1	1.7E0	5.3E0	5.6E0	6.1E0	2.1E1	1.3E1	1.7E0	5.0E0	2.3E0
Recurring	8.0E3	3.5E3	1.3E4	6.4E3	8.4E4	1.9E2	3.1E1	3.5E3	1.4E4	6.0E3	1.2E4	1.4E2	3.2E1	3.2E2
S-E	2.7E2	2.3E2	4.3E2	3.8E3	2.3E2	2.1E2	3.0E2	2.2E2	4.3E2	3.4E3	3.8E2	3.4E2	2.9E2	9.0E1
Imbalanced	2.1E0	4.8E0	5.4E0	1.4E1	1.4E1	1.6E0	4.6E0	4.9E0	5.6E0	2.0E1	1.1E1	1.5E0	4.2E0	2.0E0
Algorithm	LB-RF	LB-SVM	LB-HT	LB-SGD	LB-NBMT	OZA-RT	OZA-RF	OZA-SVM	OZA-HT	OZA-SGD	OZA-NBMT	CFIM	M3	ASHT
20ng Grad.	4.2E1	4.9E2	2.9E4	7.3E2	1.4E2	3.5E1	3.5E1	1.5E2	2.2E4	5.3E2	7.9E1	1.2E5	3.0E4	4.9E3
Reuters Grad.	3.8E2	4.7E2	5.6E3	8.9E1	1.5E1	1.9E2	6.4E2	2.7E2	4.1E3	6.6E1	1.5E1	7.7E3	8.0E3	8.6E2
20ng Sudd.	1.9E1	2.2E2	8.4E3	3.1E2	5.7E1	1.8E1	1.9E1	2.1E2	7.3E3	2.5E2	4.6E1	7.0E4	1.4E4	2.3E3
Reuters Sudd.	1.5E2	2.7E2	4.9E3	6.4E1	1.0E1	8.3E1	2.5E2	9.7E1	3.2E3	4.6E1	1.0E1	7.6E3	6.3E3	5.2E2
Spam	2.7E0	2.0E1	2.0E1	1.3E0	2.4E0	2.2E0	2.7E0	2.3E1	6.1E0	1.3E0	3.4E0	3.2E0	1.4E1	0.7E0
Recurring	6.2E2	5.8E2	8.3E3	2.0E2	2.5E1	4.6E2	2.5E3	6.2E2	7.0E3	1.2E2	2.4E1	6.1E2	9.5E3	1.2E3
S-E	2.4E2	2.9E3	7.6E2	1.8E2	1.4E2	8.7E1	1.7E2	2.8E3	5.4E2	3.2E2	1.3E2	1.1E4	1.2E3	2.2E2
Imbalanced	2.4E0	1.4E1	1.9E1	1.2E0	3.1E0	2.1E0	2.4E0	1.6E1	6.1E0	1.1E0	3.0E0	2.5E0	8.9E0	0.8E0

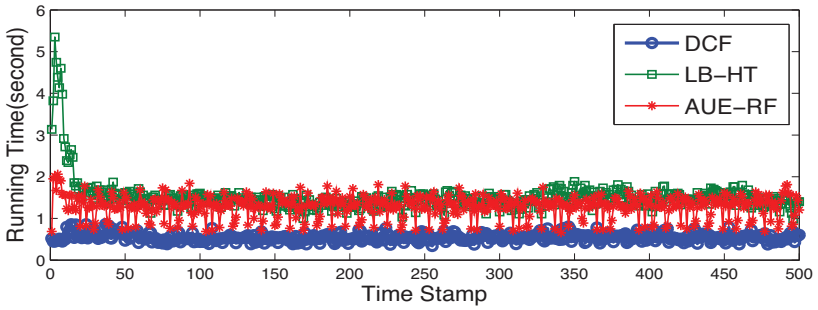


Fig. 10. Running time of the three algorithms in the S-E stream.

the robustness of the ensemble model is enhanced. The highest accuracy is achieved when ε is increased to 0.6. After a slight decline, the accuracy reaches the second highest point as ε keeps increasing. The main reason is that the scope of concept drift in the S-E stream is different. The small value of ε is sensitive to tiny changes, but it leads to the discarding of some useful clustering trees when a drastic concept drift occurs. On the contrary, setting parameter ε with a large value enables the ensemble to adapt to drastic concept drift, but it may bring some useless clustering trees participating in classification.

τ is the minimum value to ensure that the accuracy weight of each CT is greater than 0. It has subtle impact on classification accuracy. Fig. 9(b) describes the impact of τ on classification performance. We can notice that the average accuracy remains close to a constant for different values of τ.

Purity p is the measure to determine whether to split a cluster node in the clustering tree. The sensitivity of purity with respect to average accuracy for the S-E stream is shown in Fig. 9(c). Note that the average accuracy keeps increasing when p is assigned the values from 0.5 to 1. That is mainly because the higher value of p is, the more possible the node belonging to a single class. Therefore, a test instance is more likely to be classified correctly.

7.7. Running time

We compare DCF with other baseline methods with respect to the running time. Table 3 shows that it is more time-consuming for methods which achieve higher accuracy in general, while methods that achieve lower accuracy consume less time. Compared with the top 5 high-accuracy methods, we can observe that: (1) the ensemble models with NBMT as sub-classifier require the least running time for the synthetic streams. Apart from these methods, DCF requires the least running time for these textual streams. On average, the time consumption of DCF is approximately 75% of that of the method that achieves the second highest accuracy; (2) DCF requires only 2.06 s and 2.29 s for the imbalanced Spam Assassin streams and the Spam Assassin stream, respectively. It costs the minimal time to achieve the highest average accuracy for these two streams; (3) for the S-E stream, the running time overhead of DCF is slightly more than that of the methods (i.e., LB-NBMT and ASHT) that achieve the highest accuracy, but it requires less time than the method (AUE-NBMT) that achieves the second highest accuracy. Furthermore, the running time of DCF is only around 35.5% of the running time of LB-HT; (4) the most time-consuming textual stream is the 20ng gradual stream because of its scale and high dimensionality. The running time overhead of DCF is more than that of AUR-RF, but it is less than that of AWE-RF.

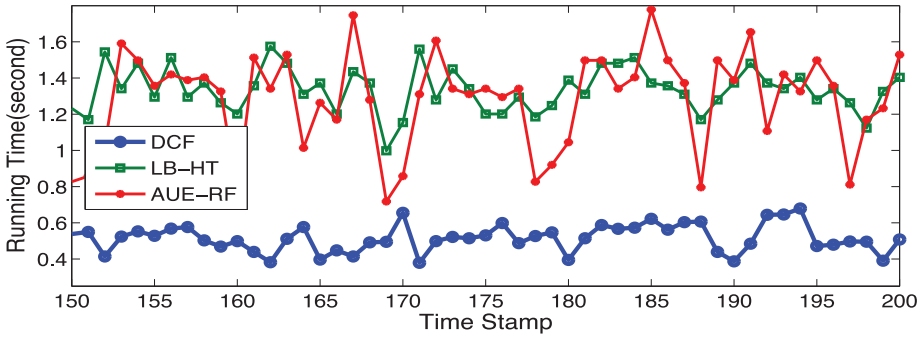


Fig. 11. Running time of the three algorithms at time stamps between 150th and 200th in the S-E stream.

Fig. 10 shows an example of time consumption at each time stamp for the S-E stream. Fig. 11 reveals the experimental results of three methods with respect to the running time during the 150–200 time stamps of the S-E stream. The time consumption of LB-HT is similar to that of AUE-RF in different periods. However, we can notice that the time curve of DCF is relatively stable when compared to that of LB-HT and AUE-RF. It is possible that the running time of the DCF model is heavily influenced by the number of classifiers participating in classification.

8. Discussion

Based on our experimental results, we can observe that the proposed DCF method outperforms other baseline methods for most of the textual streams. DCF consistently achieves the highest average accuracy with around 1–5% improvement for five textual streams (eight in total), especially for the real-world streams. DCF is able to quickly adapt its classification process after concept drift, and it achieves a steady growth of plotting accuracy during the stable concept period for most of the experiments.

In comparison to DCF, AWE-NBMT, ASHT and LB-HT are three competitive methods with respect to average accuracy. AWE-NBMT obtains the highest accuracy for the 20ng stream. Though the average accuracy of AWE-NBMT is around 1% higher than that of DCF, it fails to achieve high accuracy for three real-world streams. Meanwhile, LB-HT and ASHT perform better than DCF for the S-E stream, but the average accuracies of LB-HT and ASHT are much lower than that of DCF for the synthetic streams. On the other hand, the plotting accuracy curve of LB-HT shows large performance fluctuation when sudden concept drift occurs, and its performance slowly improves during the model rebuilding period. In addition, our experimental results reveal that NBMT is the best sub-classifier for all the baseline methods with respect to most of the textual streams.

From the empirical study of the parameters i.e., $M_{\max}$ pertaining to DCF, we can observe that DCF performs better than both AUE-RF and AUE-SVM for the Spam stream when the values of $M_{\max}$ vary. Similar results are shown in the recurring stream with different values of $M_{\max}$. Moreover, DCF is not sensitive to the other parameters such as ε, τ, and p according to the experimental results pertaining to the S-E stream.

Regarding the running time described, our experimental result show that the running time of the ensemble models with NBMT adopted as a sub-classifier seems to be shorter than that of DCF for the synthetic textual streams. However, DCF requires less running time for the three real-world streams. This indicates that DCF is applicable to real-world textual stream classification tasks.

9. Conclusions

In this paper, a new ensemble method named, DCF, is proposed to deal with textual stream classification with concept drift. More specifically, we design an adaptive ensemble strategy for selecting the discriminative CTs and a dual voting strategy that takes into account both credibility and accuracy of classifiers to boost classification performance. Based on five synthetic textual streams and three real-world textual streams, our experimental results demonstrate that the DCF model is more effective than other state-of-the-art streaming classification methods for most of the high-dimensional textual streams.

For future work, we plan to further extend our current model in two aspects. First, we will extend our model by designing a dynamic stream clustering tree that adopts an incremental learning approach to reduce the execution time. Second, it is an interesting research topic to extend our model by using a semi-supervised stream classification approach.

Acknowledgement

This research was supported in part by NSFC under Grant No.61572158 and No.61272538, Shenzhen Science and Technology Program under Grant No.JCYJ20140417172417128, JSGG20141017150830428 and JSGG20150512145714247. Lau's work

was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project: CityU 11502115).

References

[1] H. Abdulsalam, D. Skillicorn, P. Martin, Classification using streaming random forests, IEEE Trans. Knowl. Data Eng. 23 (1) (2011) 22–36.
[2] E. Acar, M. Rais-Rohani, Ensemble of metamodels with optimized weight factors, Struct. Multidiscip. Optim. 37 (3) (2009) 279–294.
[3] A. Bifet, R. Gavalda, Learning from time-changing data with adaptive windowing, in: Proceedings of 2007 SIAM International Conference on Data Mining, Minnesota, USA, 2007, pp. 443–448.
[4] A. Bifet, G. Holmes, B. Pfahringer, Leveraging bagging for evolving data streams, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2010, pp. 135–150.
[5] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, R. Gavaldà, New ensemble methods for evolving data streams, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2009, pp. 139–148.
[6] A. Bifet, G. Holmes, B. Pfahringer, P. Kranen, H. Kremer, T. Jansen, T. Seidl, MOA: Massive online analysis, a framework for stream classification and clustering, JMLR: Workshop Conf. Proc. 22 (2010) 44–50.
[7] A. Bifet, R. Kirkby, Data Stream Mining a Practical Approach, Technical Report, University of Waikato, 2009.
[8] D. Brzeziński, J. Stefanowski, Accuracy updated ensemble for data streams with concept drift, Hybrid Artificial Intelligent Systems, Springer, 2011, pp. 155–163.
[9] W. Fan, Systematic data selection to mine concept-drifting data streams, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2004, pp. 128–137.
[10] M.J. Hosseini, Z. Ahmadi, H. Beigy, Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification, Evol. Syst. 4 (1) (2013) 43–60.
[11] I. Katakis, G. Tsoumakas, E. Banos, N. Bassiliades, I. Vlahavas, An adaptive personalized news dissemination system, J. Intell. Inf. Syst. 32 (2) (2009) 191–212.
[12] I. Katakis, G. Tsoumakas, I. Vlahavas, Dynamic feature space and incremental feature selection for the classification of textual data streams, in: Knowledge Discovery from Data Streams, Springer, 2006, pp. 107–116.
[13] I. Katakis, G. Tsoumakas, I. Vlahavas, Tracking recurring contexts using ensemble classifiers: an application to email filtering, Knowl. Inf. Syst. 22 (3) (2010) 371–391.
[14] J. Kolter, M. Maloof, Dynamic weighted majority: an ensemble method for drifting concepts, J. Mach. Learn. Res. 8 (2007) 2755–2790.
[15] L. Kuncheva, Classifier ensembles for changing environments, Multiple Classifier Systems, Springer, 2004, pp. 1–15.
[16] L.I. Kuncheva, Classifier ensembles for detecting concept change in streaming data: overview and perspectives, in: Proceedings of the 2nd Workshop on Supervised and Unsupervised Ensemble Methods and Their Applications (SUEMA), 2008, 2008, pp. 5–10.
[17] X. Li, P. Yu, B. Liu, S. Ng, Positive unlabeled learning for data stream classification, Proceedings of 2009 SIAM International Conference on Data Mining, 2009, pp. 256–270.
[18] Y. Li, E. Hung, K. Chung, A subspace decision cluster classifier for text classification, Expert Syst. Appl. 38 (10) (2011) 12475–12482.
[19] Y. Li, E. Hung, K. Chung, J. Huang, Building a decision cluster classification model for high dimensional data by a variable weighting k-means method, in: AI 2008: Advances in Artificial Intelligence, Springer, 2008, pp. 337–347.
[20] P. Lindstrom, S. Delany, B. Mac Namee, Handling concept drift in a text data stream constrained by high labelling cost, in: Proceedings of Florida Artificial Intelligence Research Society Conference, FLAIRS, AAAI Press, Menlo Park, 2010, pp. 32–37.
[21] B. Liu, Y. Xiao, L. Cao, P. Yu, Vote-based LELC for positive and unlabeled textual data streams, in: Proceedings of 2010 IEEE International Conference on Data Mining Workshops (ICDMW), IEEE, 2010, pp. 951–958.
[22] M. Masud, J. Gao, L. Khan, J. Han, B. Thuraisingham, Classification and novel class detection in concept-drifting data streams under time constraints, IEEE Trans. Knowl. Data Eng. 23 (6) (2011) 859–874.
[23] M.M. Masud, Q. Chen, L. Khan, C.C. Aggarwal, J. Gao, J. Han, A. Srivastava, N.C. Oza, Classification and adaptive novel class detection of feature-evolving data streams, IEEE Trans. Knowl. Data Eng. 25 (7) (2013) 1484–1497.
[24] B.S. Parker, L. Khan, A. Bifet, Incremental ensemble classifier addressing non-stationary fast data streams, in: Proceedings of 2014 IEEE International Conference on Data Mining Workshop (ICDMW), IEEE, 2014, pp. 716–723.
[25] M. Perrone, L. Cooper, When Networks Disagree: Ensemble Methods for Hybrid Neural Networks, Technical Report: DTIC Document, DTIC, 1992.
[26] M. Scholz, R. Klinkenberg, An ensemble classifier for drifting concepts, in: Proceedings of the Second International Workshop on Knowledge Discovery in Data Streams, Citeseer, 2005, pp. 53–64.
[27] G. Song, Y. Ye, A dynamic ensemble framework for mining textual streams with class imbalance, Sci. World J. 2014 (1) (2014) 1–11.
[28] Z. Sun, Y. Ye, W. Deng, Z. Huang, A cluster tree method for text categorization, Proc. Eng. 15 (2011) 3785–3790.
[29] H. Wang, W. Fan, P. Yu, J. Han, Mining concept-drifting data streams using ensemble classifiers, in: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2003, pp. 226–235.
[30] X. Wang, C. Zhai, X. Hu, R. Sproat, Mining correlated bursty topic patterns from coordinated text streams, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2007, pp. 784–793.
[31] Y. Zhang, X. Jin, An automatic construction and organization strategy for ensemble learning on data streams, ACM SIGMOD Rec. 35 (3) (2006) 28–33.
[32] Y. Zhang, X. Li, M. Orlowska, One-class classification of text streams with concept drift, in: Proceedings of 2008 IEEE International Conference on Data Mining Workshop (ICDMW), 2008, pp. 116–125.