New Generation Computing (2021) 39:341–376
https://doi.org/10.1007/s00354-021-00126-2

# Recurring Drift Detection and Model Selection-Based Ensemble Classification for Data Streams with Unlabeled Data

Peipei Li[1,2] · Man Wu[1,2] · Junhong He[3] · Xuegang Hu[1,2]

Received: 9 July 2020 / Accepted: 9 March 2021 / Published online: 20 April 2021

## Abstract
Data stream classification is widely popular in the field of network monitoring, sensor network and electronic commerce, etc. However, in the real-world applications, recurring concept drifting and label missing in data streams seriously aggravate the difficulty on the classification solutions. And this challenge has received little attention from the research community. Motivated by this, we propose a new ensemble classification approach based on the recurring concept drifting detection and model selection for data streams with unlabeled data. First, we build an ensemble model based on the classifiers and clusters. To improve the classification accuracy, we use the ensemble model to predict each data chunk and partition clusters according to the distribution of predicted class labels. Second, we adopt a new concept drifting detection method based on the divergence of concept distributions between adjoining data chunks to distinguish recurring concept drifts. All historical new concepts will be maintained. Meanwhile, we introduce the time-stamp-based weights for base models in the ensemble model. In the selection of the base model, we consider the time-stamp-based weight and the divergence between concept distributions simultaneously. Finally, extensive experiments conducted on four benchmark data sets show that our approach can quickly adapt to data streams with recurring concept drifts, and improve the classification accuracy compared to several state-of-the-art classification algorithms for data streams with concept drifts and unlabeled data.

**Keywords** Data stream classification · Ensemble learning · Recurring concept drift · Unlabeled data

✉ Peipei Li
peipeili@hfut.edu.cn

Extended author information available on the last page of the article

## Introduction

With the rapid development and broad applications of information technologies, streaming data have become universal, such as Internet search requests, sensor data, supermarket transactions, telephone call records, data from satellites and astronomy and alike. However, in contrast to the traditional data sources of data mining, these data present new characteristics as being continuous, high-volume, open-ended, especially concept drifting and label missing. Meanwhile, in the real world, recurring events will cause recurring concepts in data streams, such as weather changes and buyer habits. It is hence a large challenge for most existing work on classification of data streams, because existing approaches probably ignore the facts of concept drifts let alone recurring concept drifts, or they always assume that all streaming data are labeled and the class labels are immediately available.

Considering the issue of concept drifts in data streams, namely the potential and fundamental changes for the target concepts interested by people [1], researchers have proposed many ensemble models for concept drifting data streams [2, 3]. Contrary to the traditional online learning approaches based on a single model [4–7], ensemble learning employs a divide-and-conquer approach to first split the continuous data streams into small data chunks, and then builds light-weight base classifiers from the small chunks. Finally, all base classifiers are combined together for prediction. Thus, there are several advantages for the ensemble model, such as achieving lower variances, easily to be parallelized and easily adapting to the concept drifting data streams. However, these approaches only store the current concept and have to re-learn every time when a new concept occurs. Re-learning significantly affects the performance of these classification models. Therefore, an ideal classification model for streaming data mining should be capable of learning in one pass, be able to do any-time classification, track the drift in the data over time, and remember historically learned concepts.

Considering the label missing issue in data streams, namely a large number of unlabeled instances with a few labeled instances, it is challenging for ensemble approaches mentioned above in the classification accuracy, because they always assume the incoming data streams completely have the class labels. In fact, researchers also have proposed some classification approaches for data streams with unlabeled data, such as an ensemble model combining both classifiers and clusters together [8], a single incremental decision tree-based classification algorithm [9], an efficient semi-supervised classification algorithm based on cluster-impurity measure [10] and an online data stream classification using selective self-training [11]. These approaches mentioned above have explored the challenges in the classification on data streams with concept drifts and unlabeled data, but they mainly recover from concept drifts by updating the ensemble model without the concept drifting detection or they utilize the classification error-based mechanism to detect the concept drifts. There is hence still plenty of room to improve the classification accuracy and the performance in the concept drifting detection.

Therefore, we propose a new ensemble classification approach based on recurring concept drifting detection and model selection for data stream with concept drifts and unlabeled data. Our contributions are as follows.

First, with the influence from the work in [8], we also use an ensemble model combining classifiers with clusters. However, the difference is that the clusters are built using the distribution of class labels predicted in the current ensemble model instead of the k-means algorithm mentioned in [8]. This is because it is beneficial to avoid the disadvantages of the selection in the inherent *k* value and the impact from noisy data in k-means.

Second, our approach can handle two major challenges in data streams with recurring concept drifts and unlabeled data. More specifically, considering the first challenge of how to detect the recurring concept drifts in data streams with unlabeled data, we adopt a new concept drifting detection method based on the divergence of concept distributions between two adjoining data chunks. In this paper, we use the clusters in each data chunk approximately to represent the concept distribution, and use the distance between clusters to evaluate the divergence between two concept distributions. For adapting to recurring concept drifts, in our approach, we maintain all historical concepts. Considering the second issue of how to choose the base model to predict the incoming data chunk, especially when the concept distribution in the latest data chunk is similar to that of multiple base models in the ensemble model, we first use the time stamp weighting mechanism to weight each base model. In the selection of the base model, we consider both the weights and the divergence between concept distributions.

Finally, extensive studies on benchmark data sets show that our approach can quickly adapt to the recurring concept drifting data streams. Meanwhile, it can improve the classification accuracy compared to the state-of-the-art classification approaches for data streams with concept drifts and unlabeled data.

The rest of this paper is organized as follows. The next section surveys related work. The subsequent section describes our ensemble model based on concept drift detection and model selection in detail. The penultimate section analyzes the experimental results, and we conclude this paper in the final section.

## Related Work

In this section, we will introduce the state-of-the-art non-recurring and recurring concept drifting classification approaches for data streams without and with unlabeled data respectively.

### Classification on Concept Drifting Data Streams Without Unlabeled Data

Regarding the state-of-the-art classification algorithms for non-recurring and recurring concept drifting data streams without unlabeled data, we can divide these algorithms into two categories such as the single model-based and the ensemble-based approaches, where the single model-based approaches can be divided into the following.

*Decision tree-based algorithms* Hulten et al. [12] proposed an efficient algorithm for mining decision trees from continuously changing data streams, called CVFDT (Concept-adapting Very Fast Decision Tree learner). It is based on a single Hoeffding tree, and adopts the "single window of time" and "alternative sub-trees" to tackle concept drifts. Rutkowski et al. [13] proposed a new Hoeffding tree algorithm for data stream classification. It has a solid mathematical basis in the best attribute selection. In the following year, Gama et al. [14] proposed a concept drift detection method using Page Hinkley test from prequential error estimates. It is built on the refined classifiers of Hoeffding tree implemented in MOA [15]. Pears et al. [16] proposed a sequential change detection model with reservoir sampling to detect concept drifts, and it is integrated with the Hoeffding Adaptive Decision tree to validate the better classification performance.

*Decision rule-based algorithms* Almeida et al. [17] presented Adaptive Model Rules (AMRules), the first streaming rule learning algorithm for regression problems. Kpotufe and Orabona [18] proposed a regression-tree tuning-based algorithm for data streams. It proved a new regression lower-bound which is independent of a given data size, and hence it is more appropriate for the streaming setting. Shao et al. [19] proposed an incremental learning algorithm based on the prototype and hypercubic decision rules for mining numeric data streams. It concerns with the simplicity of the model and the time complexity as primary goals. Kosina and Gama [20] presented the Very Fast Decision Rules (VFDR) algorithm for data streams and they further presented the adaptive extension (AVFDR) to detect changes in data streams and adapt to the decision model. That is, each individual rule in AVFDR monitors the evolution of performance metrics to detect concept drifts.

*Other model-based algorithms* Mena-Torres and Aguilar-Ruiz [21] introduced a similarity-based data stream classifier. It adopts a novel insertion/removal policy that quickly adapts to the data tendency and maintains a representative, small set of examples and estimators. And it removes useless classes that do not add any value to the classification process to detect novel classes. Rosa et al. [22] introduced a new algorithmic approach for nonparametric learning in data streams. It learns a model that covers the input space using simple local classifiers. The distribution of these classifiers dynamically adapts to the local (unknown) complexity of the classification problem. Frías-Blanco et al. proposed new concept drifting methods based on Hoeffding's Bounds [23] to monitor the performance metrics measured during the learning process, to trigger drift signals when a significant variation has been detected. It adopts a Naïve Bayes classifier and a Perceptron to evaluate the performance of the methods over synthetic and real data. Chen et al. proposed a new selective prototype-based learning (SPL) method on evolving data streams. It dynamically maintains a set of important instances to capture the time-changing concepts [24], and then uses these instances to predict the labels of new incoming instances with nearest neighbor classifier. If there are many wrongly classified examples with the same label in a period, it speculates that a concept drift may emerge.

However, all aforementioned approaches only depend on a single classification model in the tackling of the data streaming classification, thus there is still a lot of room to improve the classification accuracy compared to the ensemble model-based approaches.

In fact, the study actually found that constructing a series of simple classifiers is simpler and more feasible than building a complex single classifier [25, 26]. Therefore, many ensemble-based approaches for data streams have been proposed as follows. More precisely, Kolter and Maloof proposed an additive expert ensemble algorithm [25] and Sun et al. proposed an ensemble multi-classifier-based method to detect concept changes from the data streams in an incremental way [27]. Fan et al. proposed a concept drifting method based on random decision trees [26]. It uses the new and old data corresponding to the accuracy of the model to detect whether it has the concept drifts, and uses the cross validation mechanism to improve the classification accuracy. Ramamurthy and Bhatnagar proposed an ensemble learning-based approach to handling data streams with multiple underlying modes [28]. The proposed approach builds a global set of decision trees from sequential data chunks and creates new classifiers to represent the recurrent concept in the stream. Li et al. proposed an ensemble algorithm based on random decision trees [29]. It adopts the Hoeffding boundary inequality to specify the thresholds in the concept drifting detection. Zhu et al. proposed a random decision tree-based ensemble classification algorithm for data streams [30]. It uses the changes of the original data distribution monitored in double windows to distinguish concept drifts. Bardda et al. presented the scale-free network classifier-based ensemble method for data streams [31]. It represents the ensemble as a network to extract centrality metrics, which are used to perform a weighted majority vote, where the weight of a classifier is proportional to its centrality value. Islam proposed a class-based ensemble of classification model addressing the issues of recurring and novel class in the presence of concept drift and noise [32]. Zahra et al. proposed a GraphPool framework that keeps the concepts and the transition among concepts by first-order Markov chain [33], and it refines the pool of concepts by applying a merging mechanism when necessary. Ikonomovska et al. proposed an online bagging algorithm with Hoeffding-based model trees and an online Random Forest method for regression in data stream [3]. Anderson et al. proposed an algorithm that always trains both a new classifier and a reused classifier [34], and retains the more accurate classifier when a concept drift occurs. Chiu et al. proposed a framework that makes use of diversity to decide which classifier to keep in the pool once the classifier pool reaches the maximum size [35]. Gomes et al. proposed a data stream classification system to address the challenges of learning recurring concepts in a dynamic feature space [36]. To handle recurring concepts, all stored models are combined in a dynamically weighted ensemble. Sakthithasan et al. applied ensembles of Fourier-encoded spectra to capture and mine recurring concepts in a data stream environment [37].

In sum, aforementioned algorithms mainly concerned the concept drifting detection in an ensemble classification model and few ones concerned recurring concept drifts. Meanwhile, they all are supervised and cannot handle the data streams with unlabeled data.

## Classification on Concept Drifting Data Streams with Unlabeled Data

Researchers have proposed some classification algorithms for data streams with unlabeled data. For instances, Li et al. proposed a Semi-supervised classification algorithm for data streams with concept drifts and UNlabeled data called SUN [9]. The SUN algorithm is based on an incremental decision tree model. It uses the k-Modes algorithm to generate clusters at leaves, and uses the information of the labeled instances to label the unlabeled ones. Meanwhile, it adopts the difference between clusters to detect the concept drifts. Loo and Marsono proposed an online data stream classification approach that learns with limited labels using selective self-training [11]. Patil et al. presented an adaptive model for electricity demand supply and prices by detecting and adapting changes in trends and values [38]. Meanwhile, it uses a correlation-based similarity method to produce concept clusters to handle unlabeled data and trend analysis. Sethi et al. proposed a new incremental grid density-based learning framework [39] to perform classification of streaming data with concept drift and limited labeling. Silva and Krohling proposed an online elastic ELM based framework with a semi-supervised forgetting parameter for data streams with concept drifts [40]. Ferreira proposed an online semi-supervised approach based on density-based adaptive model [41]. However, these algorithms mentioned above are built on a single model, there is hence a lot of room to improve the classification accuracy.

Thus, some ensemble classification algorithms for data streams with unlabeled data have been proposed below. For instances, Zhang et al. proposed an ensemble model which combines both classifiers and clusters together for mining data streams with unlabeled data [8]. It uses a small number of labeled instances to train a few classifiers, and uses a large number of unlabeled instances to build clusters from data streams. Masud et al. proposed an efficient semi-supervised classification algorithm based on cluster-impurity measure [10]. It builds a classification model as a collection of micro-clusters using semi-supervised clustering, and uses an ensemble of these models to classify unlabeled data. Haque et al. proposed a semi-supervised framework which uses change detection on classifier confidence to detect concept drifts [42] and to determine chunk boundaries dynamically.

Din et al. proposed an online semi-supervised learning algorithm based on micro-clusters and error-based representative learning [43]. However, the above approaches mainly recover from concept drifts by updating the ensemble model without the concept drifting detection or they use the classification error-based mechanism to detect concept drifts. It is hence incapable of detecting recurring concepts.

In fact, there exit some semi-supervised algorithms with recurring concept detection. For example, Li et al. build a decision tree for the task of detecting recurring concepts in semi-supervised datasets, called REDLLA [44]. It uses the k-means clustering algorithm to produce concept clusters and label unlabeled data in the method of majority-class at leaves of the decision tree. To detect recurring concepts, it measures the deviation between two concept clusters based on their radius and distance. However, there is still some room to improve the prediction accuracy because the REDLLA algorithm is based on a single classifier. In the following years, many ensemble algorithms for recurring concept detection have been proposed. More precisely, Gonçalves et al. proposed a framework for detecting recurring concept drifts [45]. It works by storing classifiers and samples of data used to build the classifiers. When a new concept drift occurs by the concept drift detector (e.g., DDM), the algorithm compares the new context to previous ones using a non-parametric multivariate statistical test to verify if both contexts come from the same distribution. Hosseini et al. proposed an ensemble algorithm to classify instances of non-stationary data streams in a semi-supervised environment [46]. It maintains a pool of classifiers with each classifier being representative of one single concept.

In the processing, it uses the similarity between a data chunk of instances and a classifier of the pool (namely the accuracy of that classifier on the labeled instances of that data chunk) to detect concept drifts.

Ren et al. proposed an ensemble framework that supervised information comes from the knowledge of the past chunks [47], it reuses the information of previous hypotheses and the labelled data of recurrent concepts.

Unlike the above algorithm, the proposed approach in this paper considers both recurring concept drifting and label missing issues. Considering the classification accuracy, we build the cluster models using the distribution of class labels predicted in the current ensemble model instead of the k-means algorithm used in [8]. Meanwhile, our approach simultaneously adopts the recurring concept drifting detection mechanism and the base model selection mechanism for improving the classification performance.

## Our Approach

In this section, we give the details of our approach based on the recurring concept drifting detection and model selection. With the influence from the model mentioned in [8], our work is also built on an ensemble model combined with classifiers and clusters, but our work further considers the improvement of the classification accuracy in the ensemble model and the recurring concept drifting detection. We can formalize our problem as follows.

Suppose a streaming data consists of $N$ instances $(x_i, y_i^a)$ $(N \to \infty)$, where $x_i \in R^d$ indicates an instance with $d$ dimensions, $y_i^a \in Y = \{c_1, c_2, \dots, c_v\}$ indicates the class label, and $a \in \{u, l\}$, $u$ and $l$ indicates the class label is unknown and known, respectively, $v$ indicates the number of class labels. To build the ensemble model, we divide the streaming data $S$ into $m$ data chunks with $p$ labeled data chunks (denoted as $D$) and $m-p$ unlabeled data chunks (denoted as $U$), namely $S = \{D, U\}$, $D = \{D_1^l, D_2^l, \dots, D_p^l\}$, and $U = \{D_1^u, D_2^u, \dots, D_{m-p}^u\}$. As shown in Fig. 1, with the arrival of data chunks, we first build a classifier or clusters using the latest data chunk, denoted as $f$. In terms of the recurring concept drifting detection, we use the $K$ data chunks in the same concept distribution to build a base classifier model $E_j$. Suppose the streaming data $S$ contains $n$ difference concepts, our ensemble model can be represented by $E = \{E_1, E_2, \dots, E_n\}$, where $E_j = \{f_{j1}^a, f_{j2}^a, \dots, f_{jK}^a\}$ $(1 \le j \le n, a \in \{u, l\})$, $f_{jk}^a (1 \le k \le K)$ indicates a sub-model built on the $k$ $th$ data chunk with the same concept in $E_j$, namely the classifier or the cluster model. In

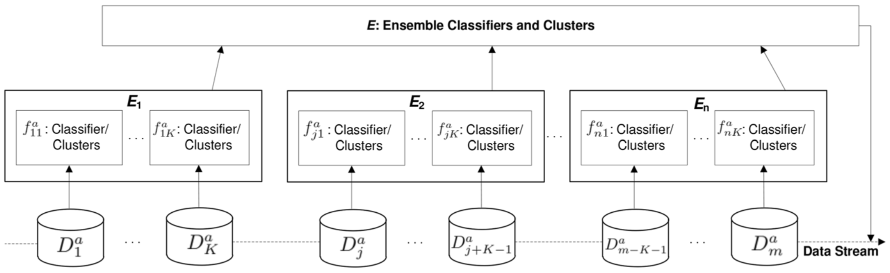
*E*: Ensemble Classifiers and Clusters
$E_1$
$f_{11}^a$: Classifier/ Clusters
$f_{1K}^a$: Classifier/ Clusters
$E_2$
$f_{j1}^a$: Classifier/ Clusters
$f_{jK}^a$: Classifier/ Clusters
$E_n$
$f_{n1}^a$: Classifier/ Clusters
$f_{nK}^a$: Classifier/ Clusters
$D_1^a$
$D_K^a$
$D_j^a$
$D_{j+K-1}^a$
$D_{m-K-1}^a$
$D_m^a$
Data Stream

**Fig. 1** Our framework of ensemble classifiers and clusters

sum, our work aims to train an ensemble model $F : E_{\sum_{i=1}^{m} D_i^a} \rightarrow Y$ that maps a feature vector to a set of labels and can adaptively adjust varying with the seen streaming data with recurring concept drifts and unlabeled data.

According to the framework of our ensemble model, we require fixing the following three issues.

*Building, prediction and updating for the ensemble model:* In the analysis of the ensemble model in [8], on one hand, it requires extra time consumption using k-means to build clusters. On the other hand, the accuracy of clusters impacts the prediction accuracy and it relies in the selection on the initial *k* centers. However, the k-means algorithm is sensitive to the noise especially in the concept drifting data streams. It is hence hard to maintain the accuracy of clusters. Therefore, to improve the effectiveness, we partition the clusters in terms of the class labels predicted in the ensemble model instead of the k-means algorithm mentioned in [8]. That is, we use the ensemble model to predict the unlabeled data chunk and partition the clusters according to the distribution of the predicted class labels. In this paper, we use the prequential evaluation [14, 48] to predict the testing data for drawing the classification accuracy curves.

*Recurring concept drifting detection* It is inevitable for recurring concept drifts and noise hidden in the real data streams. To build a robust ensemble model, it is necessary to distinguish the recurring concept drifts from noisy data for updating the ensemble model in time. However, we cannot directly apply existing concept drifting detection approaches based on the classification error, because there are large amounts of unlabeled data in data streams. Thus, we give the definition on the concept distribution and propose the divergence between concept distribution-based concept drifting detection mechanism. In this processing, we will maintain all historically learned new concepts.

*Selection on the base model* For better adapting to infinite concept drifting data streams, we update the model selectively, especially when the concept distribution of the latest data chunk is similar to that of multiple base models. Thus, we propose a model selection method based on the time stamp weighting mechanism and the divergence between concept distributions.

Details of the above techniques are as follows.

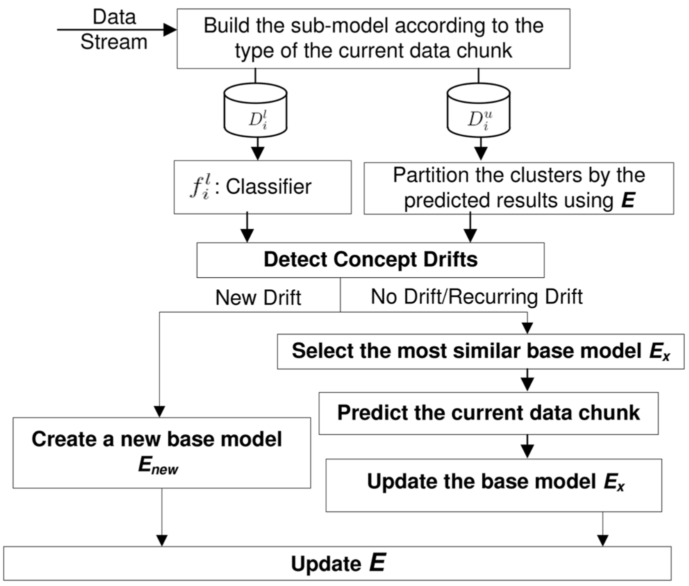
Data Stream
Build the sub-model according to the type of the current data chunk
$D_i^l$
$D_i^u$
$f_i^l$: Classifier
Partition the clusters by the predicted results using *E*
Detect Concept Drifts
New Drift
No Drift/Recurring Drift
Select the most similar base model $E_x$
Predict the current data chunk
Update the base model $E_x$
Create a new base model $E_{new}$
Update *E*

**Fig. 2** Generation of our ensemble model

## Building, Prediction and Updating for the Ensemble Model

Our ensemble model is built as shown in Fig. 2. More specifically, with the incoming of the new data chunk in $S$, if the current data chunk are labeled, denoted as $D_i^l$, we first generate a classifier $c_i$ on the labeled data chunk $D_i^l$, and then partition the clusters according to the class labels predicted by the current ensemble model, denoted as $cl_i = \{g_{i1}, g_{i2}, \dots, g_{iv}\}$, where $v$ indicates the number of class labels, $g_{ij}$ indicates the $j^{th}$ cluster ($1 \le j \le v$). Thus, we can get the model combined the classifier and the clusters together built on the current data chunk, denoted as $f_i^l = \{c_i, cl_i\}$. According to the recurring concept drifting detection method (for more details please refer to "Recurring Concept Drifting Detection"), if the concept distribution of the current data chunk represents a new concept, let the model $f_i^l$ as a new base model $E_{\text{new}}$ and add into the ensemble model $E$; Otherwise (namely non-drift or only recurring concept drift), in terms of the model selection method (for more details please refer to "Model Selection"), select the most similar base model $E_x$ from the ensemble model $E$ to predict the current data chunk and update the model $f_i^l$ into the base model $E_x$.

On the other hand, if the current data chunk is unlabeled, denoted as $D_i^u$, we use each base model $E_j$ in the current ensemble model $E$ to predict $D_i^u$ and partition the clusters according to the distribution of the predicted class labels. Correspondingly, we can get $n$ sets of clusters ($n$ indicates the count of base models). According to the concept drifting detection method, compute the similarity between each set of clusters and the concept distribution in the corresponding base model, select the most similar base model $E_x$ by the threshold $\delta$ and use the selected model (denoted as $f_i^u = \{cl_i\}$) to predict on the current data chunk, and update $f_i^u$ into the base model $E_x$. In this processing, all concepts hidden in data streams will be maintained in the ensemble model.

It is worth to mention that given an unlabeled data chunk, if we cannot find the most similar base model $E_x$ from the ensemble model $E$, this data chunk will be stored temporally. After a new base model is generated, we will partition the stored data chunk into clusters according to the distribution of class labels predicted by the new base model, called the cluster model $f_i^u$. Meanwhile, we will compute the similarity of the concept distributions between the base model and the $f_i^u$ model, if it is similar by the threshold, update the prediction results on the stored data chunk using that of the new base model.

Based on the ensemble model $E$, the prediction method is as follows. Given the testing data set $D$, the predicted label for each instance $x$ ($x \in D$) satisfies Eq. (1).

$$y^* = \text{argMax}_{y \in Y} P(y|x, E) = \text{argMax}_{y \in Y} \sum_{j=1}^{n} P(y|x, E_j), \tag{1}$$

where the probability on the instance $x$ predicted by each base model $E_j$ can be represented by the weighted probability sum on $x$ predicted by the classifier $f_{jk}^l$ and the cluster model $f_{jk}^u$ in $E_j$, denoted as Eq. (2).

$$P(y|x, E_j) = \sum_{k=1}^{a} w_k P(y|x, f_{jk}^l) + \sum_{k=a+1}^{a+b} w_k P(y|x, f_{jk}^u), \tag{2}$$

where $a$ and $b$ indicates the number of the classifier $f_{jk}^l$ and the cluster model $f_{jk}^u$, respectively, $w_k$ indicates the weight of the corresponding sub-model, it is set to an equal value (e.g., 1) in this paper.

In Eq. (2), the prediction in the model $f_{jk}^l$ indicates the prediction in the classifier $c_{jk}$, namely $P(y|x, f_{jk}^l) = P(y|x, c_{jk})$, while the prediction in the cluster model $f_{jk}^u$ uses the prediction method mentioned in [8], which can be rewritten in Eq. (3).

$$P(y|x, f_{jk}^u) = \sum_{i=1}^{v} P(y|g_{ji}) P(g_{ji}|x, f_{jk}^u), \tag{3}$$

where $v$ indicates the cluster count (equal to the number of class labels), the cluster $g_{ji} \in cl_j$ and $cl_j$ indicates the cluster set in the model $f_{jk}^u$. In our problem setting, each clustering model can only assign an instance a cluster ID that doesn't carry any class label information. Formally, for each test example $x$, a clustering model $f_{jk}^u$ will assign it a group ID with the probability $P(g_{ji}|x, f_{jk}^u)$, instead of the genuine class label $P(y|x, f_{jk}^u)$. To bridge these two different probabilities, the probability $P(y|g_{ji})$ is introduced, which reflects the mapping relationship between each cluster ID $g_{ji}$ and the genuine class label $y \in Y$. Thus, for each clustering model $f_{jk}^u$, the posterior probability $P(y|x, f_{jk}^u)$ can be estimated by integrating all the $v$ mappings together as shown in Eq. (3).

## Recurring Concept Drifting Detection

Before introducing details of the recurring concept drifting detection mechanism, we first give the definition on the formalization of the concept drift and recurring concept drift. More specifically, concept drift refers to the data distribution (denoted as $P$) evolving over time, i.e., $P_t(X,y) \neq P_{t+\Delta}(X,y)$, where $P_t(X,y)$ and $P_{t+\Delta}(X,y)$ indicates the data distribution at time $t$ and $t + \Delta$, respectively, where $X$ indicates the feature vector and $y$ indicates the label vector. A recurring concept drift occurs when the instances at time $t + \Delta$ are generated from the same distribution as that at the previously observed time $t$, i.e., $P_{t+\Delta}(X, y) = P_t(X, y)$. That is, a concept appears at a point of the past, then it has disappeared for a long time and now reappears again. For more details please refer to the survey in [49].

To improve the efficiency of our approach, we compare the concept distributions between two data chunks instead of those between two instances for concept drifting detection. In this paper, we represent the concept distribution of each data chunk using clusters partitioned by the predicted class labels in the ensemble model, thus the model built on the current data chunk $D_i^a$ can be represented by $f_i^a$, correspondingly the hidden concept distribution indicates the distribution of clusters, denoted as $cl_i = \{g_{i1}, g_{i2}, \dots, g_{iv}\}$, where $v$ indicates the label count, $g_{ij}$ indicates the $j$th cluster ($1 \leq j \leq v$). To determine whether concept drifts occur in the current data chunk, we adopt the distance evaluation method based on the divergence of the concept distributions. Thus, we require computing the divergence between the concept distributions of the current data chunk and the ensemble model with all historically learned new concepts. That is, we need to, respectively, compare the distance between clusters of the current data chunk with those of the base model $E_j$ from $E$. Therefore, we can formalize the distance mentioned above as shown in Eq. (4).

$$\mathrm{dist}(f_i^a, E_j) = \frac{1}{K}\sum_{k=1}^{K} \mathrm{dist}(f_i^a, f_{jk}^a), \tag{4}$$

where $f_i^a (a \in \{u, l\})$ and $f_{jk}^a$ indicates the classifier or the cluster model built on the data chunk $D_i^a$ and the $k$th data chunk in the base model $E_j$, respectively, and $K$ indicates the size of a base model $E_j$. Let $f_j^a = f_{jk}^a$, we can rewrite the distance between concept distributions of the above two models by the average distance between corresponding clusters as shown in Eq. (5).

$$\mathrm{dist}(f_i^a, f_j^a) = \mathrm{dist}(cl_i, cl_j) = \frac{1}{v}\sum_{k=1}^{v} \mathrm{Min}_{1 \leq t \leq v} \mathrm{dist}(g_{ik}, g_{jt}), \tag{5}$$

where $v$ indicates the cluster count, $cl_i$ and $cl_j$ indicates the cluster set in the corresponding models of $f_i^a$ and $f_j^a$, respectively, denoted as $cl_i = \{g_{ik} | 1 \leq k \leq v\}$ and $cl_j = \{g_{jt} | 1 \leq t \leq v\}$, $g_{ik}$ and $g_{jt}$ indicates the partitioned clusters on the data chunks of $D_i^a$ and $D_j^a$, respectively. According to Eq. (5), the distance between clusters can be represented in Eq. (6).

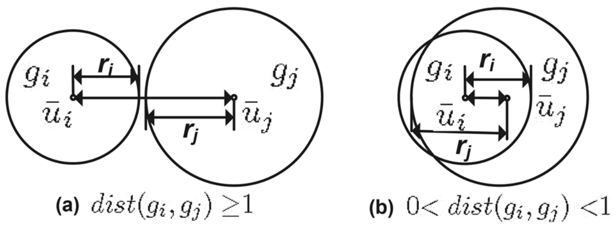

**Fig. 3** Illustration to the distance between clusters

$$\mathrm{dist}(g_{ik}, g_{jt}) = \frac{||\bar{u}_{ik}, \bar{u}_{jt}||_2}{(r_{ik} + r_{jt})}, \tag{6}$$

where $\bar{u}_{ik}$ and $\bar{u}_{jt}$ indicates the center of the clusters $g_{ik}$ and $g_{jt}$, respectively, denoted as $\bar{u}_{ik} = \frac{1}{|g_{ik}|}\sum_{j=1}^{g_{ik}} x_{ij}$, $||\cdot||_2$ indicates the Euclidean distance, $r_{ik}$ and $r_{jt}$ indicates the corresponding radius of the clusters $g_{ik}$ and $g_{jt}$, respectively, denoted as $r_{ik} = \frac{1}{|g_{ik}|}\sum_{j=1}^{g_{ik}} ||x_{ij}, \bar{u}_{ik}||_2$, and $x_{ij}$ indicates the instance consisted in the cluster $g_{ik}$.

According to the above definitions, we can write the divergence between the concept distribution of the current data chunk and that in the base model in Eq. (7).

$$\begin{aligned} \mathrm{dist}(f_i^a, E) &= \mathrm{argMin}_{E_j \in E} \mathrm{dist}(f_i^a, E_j) \\ &= \mathrm{argMin}_{E_j \in E} \frac{1}{Kv} \sum_{j=1}^{K} \sum_{k=1}^{v} \mathrm{Min}_{1 \leq t \leq v} \frac{||g_{ik}, g_{jt}||_2}{r_{ik} + r_{jt}} \end{aligned} \tag{7}$$

.

In terms of Eq. (7), we can see that the smaller the distance between clusters, the less difference between the concept distribution on the current data chunk and that in the ensemble model. In this case, it is considered as no concept drift or recurring concept drift occurring. Figure 3 gives an illustration to the simplest case, that is, the ensemble model $E$ only contains a base model and the base model is composed of a cluster model with only one cluster. Thus, the value of $\mathrm{dist}(f_i^a, E)$ depends on the possible relationship between the center distance and the radiuses of the two clusters. This figure shows two cases about the distance between clusters $\mathrm{dist}(g_i, g_j)$ below.

In case (a), we can get $\mathrm{dist}(f_i^a, E) \geq 1$, it indicates the current two clusters are not overlapping and even are separated. It is hence considered as a concept drift. In case (b), two clusters partially overlap or even completely overlap, namely $0 < \mathrm{dist}(f_i^a, E) < 1$. In this case, we need to distinguish how much the overlapping degree can be thought in the same concept. In another word, we consider the concept drift is occurring if the divergence between two concept distributions is larger enough. Because we know that a single data chunk only represents partial data distribution, there are variants of the data distributions in different data chunks. In addition, noisy data also impact the data distribution. It leads to the divergence between the data distributions of the two data chunks. Therefore, we introduce a threshold to

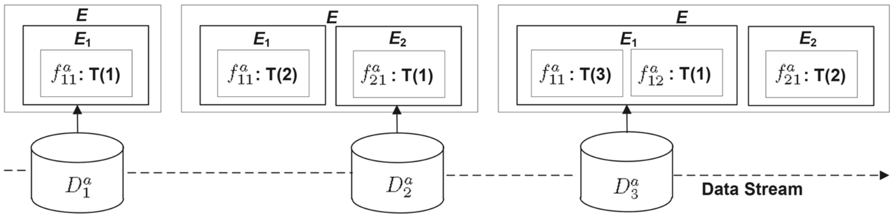

**Fig. 4** Time stamp update for sub-models in our ensemble model

determine the concept drifting cases. If $\mathrm{dist}(f_i^a, E) > \delta$ is met, it is considered as the concept drift, otherwise, it has no concept drift or recurring concept drift.

## Model Selection

We now introduce the model selection based on the time stamp weighting mechanism and the divergence between concept distributions. For better adapting to infinite concept drifting data streams, it is necessary to select a more suitable base model, especially if the concept distribution of the latest data chunk is similar to that of multiple base models in the current ensemble model, namely in the case with recurring concept drifts. Details of our model selection method are below.

First, according to the assumption the concept distributions hidden in the two neighbouring data chunks are more similar, we believe that the divergence of the concept distribution between the latest chunk and the latest base model in the ensemble model is less. Therefore, we define a time stamp-based weight for each base model $E_j$ in the ensemble model $E$ as shown in Eq. (8).

$$w_j = 1 + \sum_{k=1}^{K} \frac{1}{2^{T(k)}}, \tag{8}$$

where $K$ indicates the number of sub-models in each base model $E_j$, $T(i)$ indicates the time stamp of a sub-model $f_{jk}^a$ in the base model, it is initialized by 0. As a new data chunk arrives, the corresponding sub-model is generated, the time stamps for all sub-models in the current ensemble model are added by 1. Figure 4 illustrates the updating in the current ensemble model as the three new data chunks arrive. In this figure, the ensemble model $E$ is initialized by $\emptyset$, suppose the concept distributions hidden in data chunks $D_1^a$ and $D_2^a$ are different, and the concept distribution hidden in the data chunk $D_3^a$ is as same as that in the base model $E_1$. According to Eq. (8), we can get that the older the sub-model, the larger the value of $T(i)$, the smaller the weight sum of the base model $E_j$ in the ensemble model $E$.

Second, according to the concept drifting detection method mentioned in "Recurring Concept Drifting Detection", if the divergence of the concept distributions between the current data chunk and a base model $E_j$ in $E$ is lower than the threshold $\delta$, it is considered as no concept drift or recurring concept drift occurring. That is, the concept distributions are consistent, belonging to the same concept. To select the base model with the most similar concept distribution to that on the current data chunk, we take consideration of both the divergence between concept distributions and the weight of the base model, namely selecting the base model with the largest value defined in Eq. (9).

$$v(E_j) = argMax_{E_j \in E} \frac{w_j}{dist(f_i^a, E_j)}. \tag{9}$$

Finally, Algorithm 1 gives the framework of our Concept Drifting detection and Model Selection-based Ensembling classification approach for data streams with recurring concept drifts and unlabeled data, called CDMSE.

## Experiments

In this section, we first outline the experimental settings, including the benchmark data sets, evaluation measures and all competing approaches. Second, we give the parameter analysis for several important parameters involved in our approach. Finally, we evaluate the effectiveness and efficiency of our approach.

**Algorithm** Our CDMSE Approach

```
1: Input a streaming data $S$, the size of a data chunk $M$, the threshold in the concept
   drifting detection $\delta$ and the label count $v$;
2: Partition streaming data $S$ into labeled and unlabeled data chunks with equal size
   $M$;
3: for each data chunk $D_i^a$ $(1 \leq i \leq n)$
4:    if it is not the first data chunk
5:      for each instance $x \in D_i^a$
6:        Predict the label for $x$ using Eq. (1);
7:      Output the prediction accuracy on $D_i^a$;
8:    Build the corresponding model $f_i^a$ according to the labeling status on $D_i^a$;
9:    Detect concept drifts according to Eq. (6) and the threshold $\delta$;
10:   if $dist(f_i^a, E) > \delta$
11:      Build the new base model $E_{new}$ (namely $f_i^a \rightarrow E_{new}$) and add it into
         the ensemble mode;
12:   else
13:      Select the most similar base model $E_x$ by Eq. (9) and add the model $f_i^a$ into
         $E_x$ (namely $f_i^a \rightarrow E_x$);
```

### Experimental Settings

*Benchmark data sets* we use four benchmark synthetic data sets, such as Sea, Waveform21, Waveform40 and Hyperplane. Table 1 summarizes the details of these four data sets, including the size of the data set, the number of attributes, the label count and the ratio of unlabeled instances in the current data set. Our data sets are generated by the MOA (an experimental tool for Massive Online Analysis) [15] experimental tool. Each data set contains four different concepts

**Table 1** Data sets used in our experiments

| Data set | Sea | Waveform21 | Waveform40 | Hyperplane |
|---|---|---|---|---|
| Size (k) | 200 | 200 | 200 | 200 |
| #Attribute | 3 | 21 | 40 | 50 |
| #Label | 2 | 3 | 3 | 2 |
| Noise | 10% | 10% | 19 Irrelevant attributes | 10% |

and each concept contains 2000 instances. In our experiments, we specify the four concepts repeatedly occurring one by one, correspondingly each data set contains 100 concepts with 99 concept drifts. The first data chunk is considered as the initial training data, while the remaining data chunks will be first considered as the testing data and then as the training data.

In the following experiments of parameter evaluation and concept drifting detection, we generate three representative sets for each data set by specifying different ratios of unlabeled data (denoted as ulr), namely ulr = 0%, ulr = 50% and ulr = 80%. This is because in the case with ulr = 0%, all semi-supervised approaches used in experiments are conducted in an extreme case with all labeled data. In the case with ulr = 50%, our approach could beat all competing ones, while in the case with ulr = 80%, our approach could beat most of competing ones. For more details please refer to the following experimental analysis on Fig. 10. It is necessary to mention how to set the value of ulr. In our experiments, as each data chunk arrives, it has the probability of ulr as the unlabeled data chunk. In this case, we can make sure ulr unlabeled instances and 1-ulr labeled instances in all training data.

*Evaluation measures* we introduce four evaluation measures for concept drifting detection mentioned in [48], including (1) False Alarm: the rate that false alarms occur in the concept drifting detection; (2) missing: the rate of concepts missed in the drifting detection; (3) delay: the mean count of instances required to detect a concept drift after the occurrence of a concept drift. (4) MTFA (Mean Time between False Alarm) [50]: It characterizes how often we get false alarms when there is no change. We use the evaluation measures for the classification performance including accuracy with variance and the count of win/tie/lose. According to the prequential evaluation [14, 48], namely testing before training, in our approach, as each data chunk arrives, if it is the first data chunk, we will use the current data chunk to build a model. Otherwise, we first use the generated training model to predict the current data chunk and then use the current data chunk to build a model for the updating of ensemble model. Therefore, the accuracy on each data chunk can be obtained by the ratio of the number of correct instances to the total number of testing instances. And accuracy here indicates the average value over the prediction accuracies on all data chunks. Variance indicates the variance of accuracies over data chunks, the smaller the variance, the more stable the accuracy predicted in the ensemble model. win/tie/lose indicates the times our approach wins/ties/loses compared to the competing

**Table 2** Concept drifting detectors

| | |
|---|---|
| DDM [6] | Drift detection method |
| EDDM [7] | Early drift detection method |
| CusumDM [52] | Drift detection method based on Cusum (Cumulative Sum of Recursive residual) |
| PageHinkleyDM [14] | Drift detection using Page Hinkley test |
| HDDM_W_Test [23] | Online drift detection based on Hoeffding's bounds using the EWMA (Exponentially weighted moving average) statistic |
| SUN [9]/REDLLA [8] | Cluster distance-based drifting detection |
| RDDM [53] | Reactive drift detection method |
| d3 [54] | Unsupervised concept drift detection with a discriminative Classifier |

approach, respectively. In addition, we further investigate the significance difference between our approach and these four competing data stream classification approaches by the *Nemenyi test*.

*Competing approaches* We will investigate the performance of our approach in two directions. One is to investigate the performance in the concept drifting detection. We select eight state-of-the-art concept drifting detectors in data streams as the baselines. Details are shown in Table 2. And the other is to investigate the performance in the classification. We select four competing approaches for data streams with unlabeled data, including the Ensemble classification approach combining both Classifiers and clUsters together called ECU [8] , a Single incremental decision tree-based classification approach for data streams with concept drifts and UNlabeled data called SUN [9], a Semi-supervised classification algorithm for data streams with REcurring concept Drifts and Limited LAbeled data called REDLLA [44], and an online learning algorithm based on micro-clusters and error-based representative learning called MC [43]. Since the MC method requires a certain amount of instances to be initialized, the first data chunk is selected for initialization, other parameter settings follow the settings of the original paper [43]. In our experiments, we use C4.5 decision tree as a base classifier and the k-means as the base clustering algorithm in ECU. Both algorithms are from the Weka experimental platform [51]. And the number of sub-models in the ensemble model for the ECU approach is set to 10. All experiments are performed on a P4, 2.5 GHz PC with 4 G main memory, running Windows7 with the program platform of Eclipse Jdk1.7.

## Parameter Evaluation

In this subsection, we aim to select the optimum values of all important parameters involved in our approach, including the size of a base model $E_j$ namely $K$, the weight ratio of a classifier vs. a cluster model in our approach, the size of a data chunk $M$ and the threshold used in the recurring concept drifting detection $\delta$. All experiments are conducted in cases varying with values of the specified parameter while keeping others unchanged. Details of parameter settings are as follows.

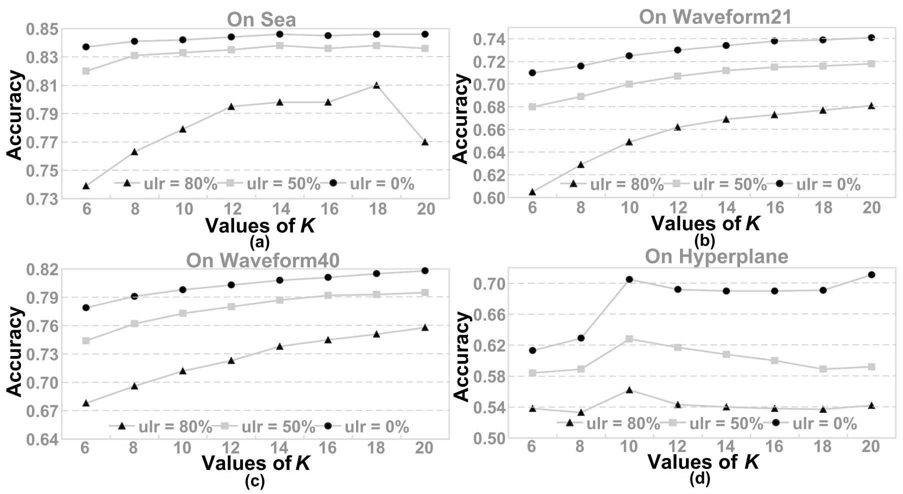

**Fig. 5** Performance of our approach varying with different sizes of ensemble model ($K$)

We first take the prediction accuracy of our CDSME approach varying with values of $K$ from 6 to 20 on four data sets to show the selection on the optimal value of $K$. In the observation of experimental results as shown in Fig. 5, we can see that as values of $K$ increase, the prediction accuracy of our approach is first increasing and and then maintaining stably on data sets of Sea and Hyperplane, while it is continuously increasing by a narrow range on data sets of Waveform21 and Waveform40. We can get a high prediction accuracy on other three data sets even up to the highest accuracy on Hyperplane if specifying $K \geq 10$. In the meanwhile, considering the larger the number of $K$ is, the more the resource consumes. Therefore, in the following experiments, we select $K = 10$ as a candidate optimal value for our approach. This value also follows the setting in ECU [8].

Second, Fig. 6 reports the prediction accuracy of our approach in three cases of ulr varying with weight ratios of a classifier vs. a cluster model in [2:1, 1,5:1, 1:1, 1:1.5, 1:2] on four benchmark data sets. From experimental results we can observe that our approach performs the best in the case with the weight ratio of a classifier vs. a cluster model as 1:1. That is, the base models in our approach have the equal weights. Thus, we specify $w_k = 1$ as an optimal value involved in Eq. (2).

Third, Fig. 7 reports the prediction accuracy of our approach in three cases of ulr varying with values of $M$ from 100 to 300 with a step 50 on four benchmark data sets. In the observation of experimental results, we can see that as values of $M$ increase, the prediction accuracy of our approach is firstly increasing up to a peak value at $M = 200$, then decreasing to a relatively stable point on three benchmark data sets including Sea, Hyperplane and Waveform21. While the prediction accuracy of our approach on Waveform40 is continuously increasing. This is because the data set of Waveform40 contains 19 irrelevant attributes, as the size of a data chunk varies from 200 to 300, the useful information will be

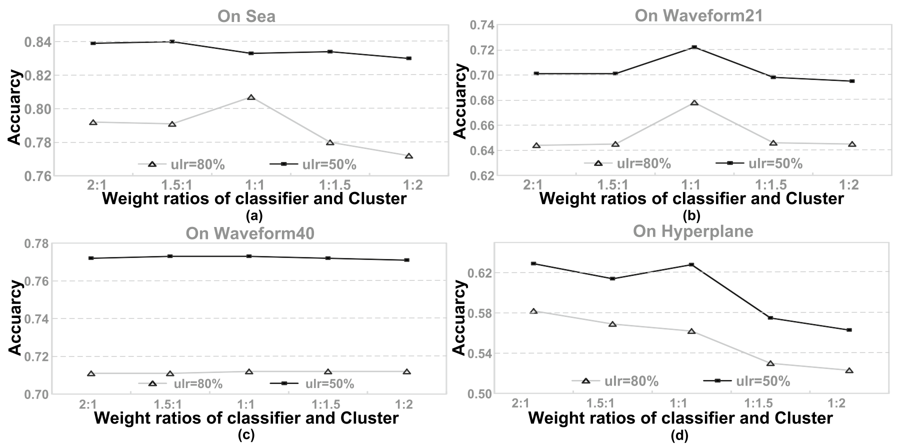

**Fig. 6** Performance of our approach varying with different weights of classifiers and clusters

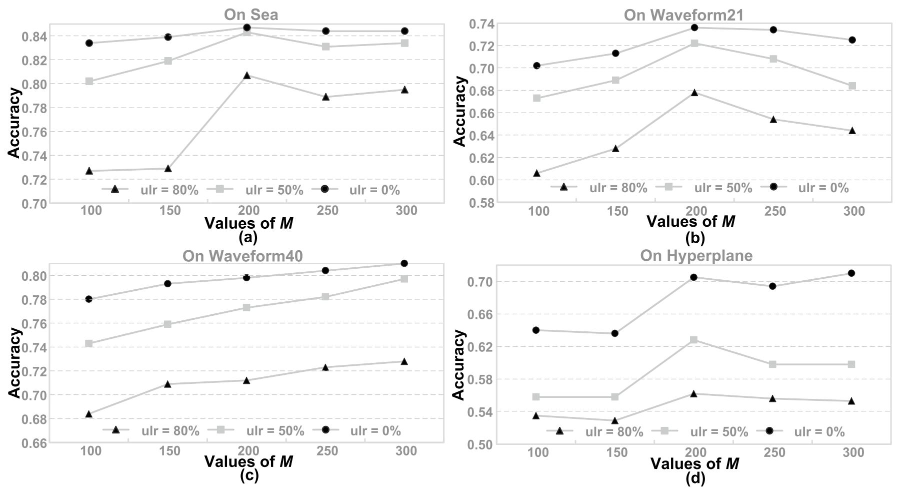

**Fig. 7** Performance of our approach varying with sizes of a data chunk ($M$)

more sufficient. The prediction accuracy is correspondingly increasing. For unity, $M = 200$ is hence set to a candidate optimal value.

Finally, we investigate the prediction accuracy of our approach in three cases of ulr varying with values of $\delta$ from 0.12 to 0.18 with a step 0.01 on four benchmark data sets. According to experimental results as shown in Fig. 8, we can see that as values of $\delta$ increase, the prediction accuracy of our approach is firstly increasing then decreasing to a relatively stable value on data sets of Sea and Waveform21, while the prediction accuracies of our approach are increasing and then maintains stably on data sets of Waveform40 and Hyperplane. All prediction accuracies are either up to a peak value or maintain a higher value at $delta = 0.15$.

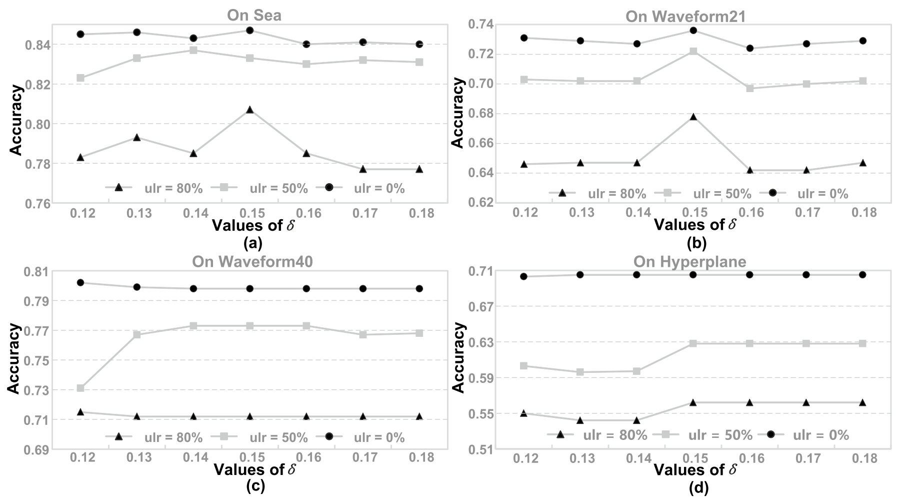

**Fig. 8** Performance of our approach varying with values of delta

Thus, we select *delta* = 0.15 as a candidate optimal value in our concept drifting detection.

In summary, the optimal parameter settings in our approach are specified below. The number of sub-models in each base model for our approach is set to $K = 10$ and the weights of all sub-models are specified equally. The size of a data chunk is set to $M = 200$ and the threshold used in the recurring concept drifting detection is set to $\delta = 0.15$. The following experimental results of our approach are obtained using these optimal values of all important parameters.

## Performance Analysis

This section will investigate the performance of our approach in two dimensions. First, we compare our approaches with and without the time stamp-based weight in the model selection as shown in Eq. (8), and then compare our approach with well-known semi-supervised approaches in the classification performance. Second, we compare our approaches with well-known concept drifting detection methods in the recurring concept drifting detection. Details of our experiments are as follows.

### Classification Performance

We first compare our approaches with and without the time stamp based weight in the model selection as shown in Fig. 9. It can be seen from experimental results that our approach with the time stamp-based weight in the model selection actually outperforms that without this handling. And then we investigate the effectiveness and efficiency of our approach compared to four well-known semi-supervised data stream classification approaches as follows.

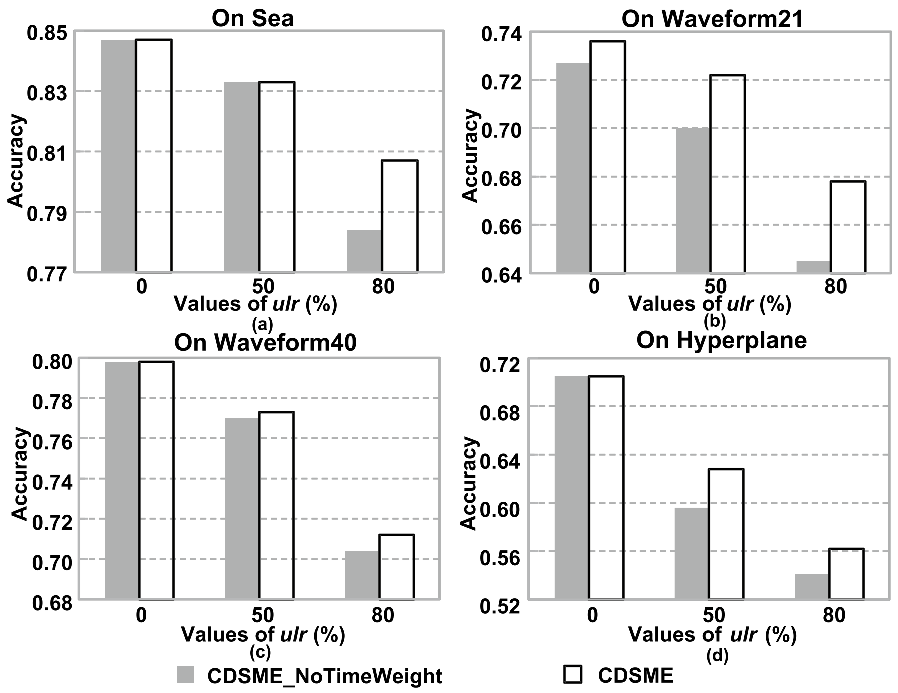

**Fig. 9** Performance of our approaches without and with time weights in the model selection

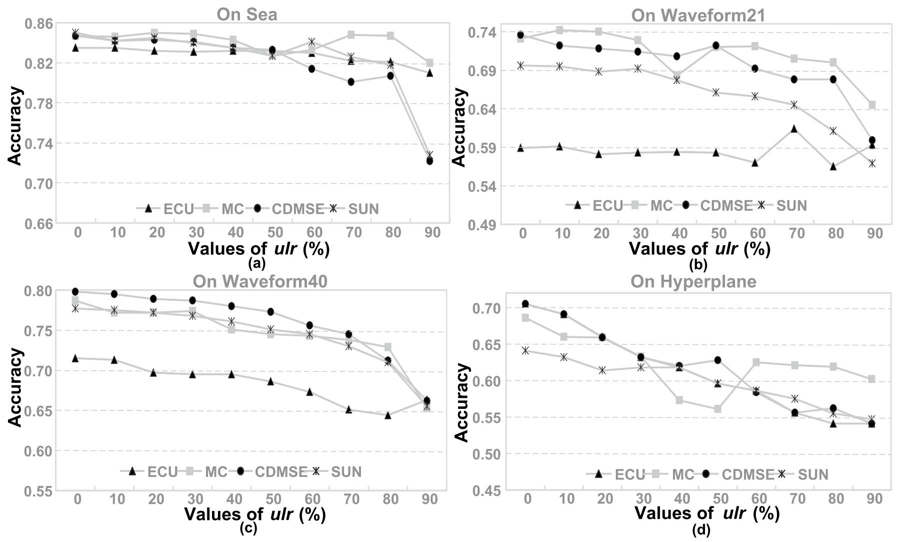

**Fig. 10** Performance of our approach varying with values of ulr

On the one hand, Fig. 10 reports the prediction accuracies of our approach compared to competing ones varying with values of ulr ranging from 0 to 90% with a step 10%. From experimental results, we can observe the followings.

First, with the increasing of the ratios of unlabeled data, the performance of all approaches in the classification accuracy is deteriorating. This is because the more unlabeled data indicate the less useful information especially for labels used in the classification, which leads to the deterioration in the classification accuracy.

Second, as compared to three competing approaches of ECU, SUN and REDLLA, our approach is superior or comparable to them in the case with ulr no more than 80%. This is because instead of using the k-means algorithm in ECU, clusters are built in our approach according to the distribution of class labels predicted in the ensemble model. It cannot significantly reduce the classification accuracy, while it probably enables improving the classification accuracy. Meanwhile, it validates that introducing the concept drifting detection and the model selection is actually conducive to improve the classification accuracy of the ensemble model. In addition, the ensemble model in our approach is also beneficial to the improvement of the classification accuracy compared to the single model in SUN and REDLLA.

Third, as compared to the MC approach, our approach can beat it in case of ulr no more than 50% while it is inferior to MC in case of ulr > 50%. This is because the MC method maintains a group of 1000 flexible micro-clusters. These micro-clusters may contain more recurring concepts due to their large sizes. But these micro-clusters are sensitive to noise, as the values of ulr increases more than 50%, the prediction accuracy of the method MC may be increasing on several data sets such as Sea and Hyperplane. This anomaly is because the MC method is based on error-based representative learning. With the increasing value of ulr, the amount of labeled noise instance is decreasing, which means that the probability of noise data interfering with the reliability of the micro-clusters is reduced. And the probability of unnecessary and false micro-clusters deleting is reduced.

In summary, our approach outperforms all competing ones in the case with ulr no more than 50%, while our approach could beat other three competing ones except of the MC approach in the case with ulr ≤ 80%. Thus, in terms of the prediction accuracy, we further select three representative cases (namely ulr = 0%, ulr = 50% and ulr = 80%) to compare the statistics of win/tie/lose as shown in Tables 3, 4 and 5. In these tables, the statistics of win/tie/lose in our approach indicate the average values over four competing approaches, and all best results are highlighted in bold. In the case of ulr = 0%, there is only a set of experimental results for SUN and REDLLA, because both approaches are based on the same single decision tree. In the observation of experimental results, we can obtain the same conclusions mentioned above. These data further validate the effectiveness of our approach compared to well-known competing ones in the handling of data streams with unlabeled data.

On the other hand, Fig. 11 shows the comparison experiments of all competing approaches in time consumptions. According to experimental results in Fig. 11, our CDMSE approach is comparable to approaches of SUN, REDLLA and ECU, while all of them are superior to the MC method in the time performance. The analysis is below. The ensemble model in our CDMSE approach is developed from the ECU algorithm, the time complexity of the basic classifier is mainly denoted as O

**Table 3** Performance comparison between our approach and competing ones on synthetic data sets with ulr = 0%

| Data set | Approach | Accuracy ± variance | Win/tie/lose |
|---|---|---|---|
| Sea | ECU | 0.835 (3) ± 0.002 | 77/643/279 (4) |
| | SUN/REDLLA | **0.850** (1) ± **0.001** | 509/70/420 (2) |
| | MC | 0.847 (2) ± 0.001 | **519/15/465** (1) |
| | CDMSE | 0.847 (2) ± 0.001 | 396/200/403 (3) |
| Waveform21 | ECU | 0.589 (4) ± 0.021 | 110/39/850 (4) |
| | SUN/REDLLA | 0.696 (3) ± 0.001 | 135/31/833 (3) |
| | MC | 0.731 (2) ± 0.001 | 489/13/497 (2) |
| | CDMSE | **0.736** (1) ± **0.001** | **753/29/217** (1) |
| Waveform40 | ECU | 0.715 (4) ± 0.009 | 123/33/843 (4) |
| | SUN/REDLLA | 0.777 (3) ± 0.001 | 194/35/770 (3) |
| | MC | 0.787 (2) ± 0.001 | 304/9/686 (2) |
| | CDMSE | **0.798** (1) ± **0.005** | **767/28/204** (1) |
| Hyperplane | ECU | 0.705 (1) ± 0.003 | 0/999/0 (4) |
| | SUN/REDLLA | 0.641 (3) ± 0.001 | 163/17/819 (3) |
| | MC | 0.686 (2) ± 0.001 | 327/10/662 (2) |
| | CDMSE | **0.705** (1) ± **0.003** | **575/261/163** (1) |
| Average Ranking | ECU | 3 | 4 |
| | SUN/REDLLA | 2.5 | 2.75 |
| | MC | 2 | 1.75 |
| | CDMSE | **1.25** | **1.5** |

$(k \cdot M^2 + v \cdot M \cdot d)$, where $k$ indicates the number of clusters on each data chunk, $v$ indicates the number of class labels, $d$ indicates the number of dimensions, and $M$ indicates the size of a data chunk, which is set to 200 in our experiments. The difference in the time consumptions as shown in Fig. 11 lies that our approach will update model according to the results of recurring concept drifting detection, while the latter always update the model as the new data chunk arrives. Thus, the whole time consumptions of our approach are lower. As compared to SUN and REDLLA approaches, they are built on a single Hoeffding decision tree with clusters at leaves. The used models are similar as ours, the time consumptions are hence similar. As compared to the MC method, its time complexity of basic classifier is denoted as O($k^*/v \cdot |D_{\text{init}}|^2 + maxMC^2$), where $k^*$ indicates the number of clusters, $D_{\text{init}}$ indicates the total number of instances in the initial data set (which is set to 1000), and maxMC indicates the maximum number of micro-clusters in the model (which is set to 1000). As the data stream flows, our method will update as the new data chunk arrives (the size of a data chunk is only $M$), while the MC method will update as the new instance arrives (the number of instances in the data stream is $m$). Obviously, the value of $m$ is more larger than the value of $M$. Thus, the total time consumption of the MC method is much larger than that of our method.

**Table 4** Performance comparison between our approach and competing ones on synthetic data sets with ulr = 50%

| Data set | Approach | Accuracy ± variance | Win/tie/lose |
|---|---|---|---|
| Sea | ECU | 0.832 (2) ± 0.002 | 411/104/484 (2) |
| | SUN | 0.827 (3) ± 0.001 | 353/51/595 (3) |
| | REDLLA | 0.826 (4) ± 0.001 | 333/49/617 (5) |
| | MC | 0.827 (3) ± 0.001 | 340/11/649 (4) |
| | CDMSE | **0.833** (1) ± **0.002** | **586/54/359** (1) |
| Waveform21 | ECU | 0.583 (5) ± 0.019 | 78/18/903 (4) |
| | SUN | 0.661 (4) ± 0.001 | 68/15/916 (5) |
| | REDLLA | 0.685 (3) ± 0.001 | 147/39/813 (3) |
| | MC | 0.720 (2) ± 0.002 | 490/9/500 (2) |
| | CDMSE | **0.722** (1) ± **0.002** | **783/20/196** (1) |
| Waveform40 | ECU | 0.686 (5) ± 0.009 | 123/61/815 (5) |
| | SUN | 0.751 (2) ± 0.001 | 241/49/709 (2) |
| | REDLLA | 0.746 (3) ± 0.001 | 196/33/770 (4) |
| | MC | 0.745 (4) ± 0.002 | 231/10/758 (3) |
| | CDMSE | **0.773** (1) ± **0.003** | **763/38/198** (1) |
| Hyperplane | ECU | 0.596 (2) ± 0.007 | 325/32/642 (4) |
| | SUN | 0.596 (2) ± 0.001 | 377/32/590 (3) |
| | REDLLA | 0.596 (2) ± 0.001 | 397/23/579 (2) |
| | MC | 0.561 (3) ± 0.003 | 196/3/800 (5) |
| | CDMSE | **0.628** (1) ± **0.007** | **653/22/324** (1) |
| Average ranking | ECU | 3.5 | 3.75 |
| | SUN | 2.75 | 3.25 |
| | REDLLA | 3 | 3.5 |
| | MC | 3 | 3.5 |
| | CDMSE | **1** | **1** |

Obviously, the time complexity of MC is much greater than the method CDMSE, therefore our CDMSE is better than MC method in time performance. In particular, the cause of above phenomenon is that our CDMSE approach uses an efficient ensemble method for classification, which can learn from the previous concepts in a very short time. Therefore, our method achieves a comprehensive consideration of time and accuracy performance.

In addition, we employ the *Nemenyi test* to further analyze the classification performance among comparing approaches in three representative cases. We treat our approach as the dominating approach, the difference between the average ranks of our approach and one comparing approach is compared with the critical difference (CD): For the Nemenyi test, we have CD = 2.749 ($q_{0.10} = 2.459$, the number of competing approaches = 5, the number of used data sets = 4) on data stream classification at significance level $\alpha = 0.10$. Accordingly, the performance between our approach and one comparing approach is deemed to be significantly different if

**Table 5** Performance comparison between our approach and competing ones on synthetic data sets with $ulr = 80\%$

| Data set | Approach | Accuracy ± variance | Win/tie/lose |
|---|---|---|---|
| Sea | ECU | 0.821 (2) ± 0.002 | 484/56/459 (2) |
| | SUN | 0.818 (3) ± 0.001 | 464/43/492 (4) |
| | REDLLA | 0.792 (5) ± 0.001 | 303/18/678 (5) |
| | MC | **0.847** (1) ± **0.001** | **745/8/246** (1) |
| | CDMSE | 0.807 (4) ± 0.004 | 469/31/499 (3) |
| Waveform21 | ECU | 0.565 (5) ± 0.012 | 91/27/881 (5) |
| | SUN | 0.611 (4) ± 0.002 | 116/14/869 (4) |
| | REDLLA | 0.658 (3) ± 0.001 | 326/35/638 (3) |
| | MC | **0.700** (1) ± **0.002** | **700/6/293** (1) |
| | CDMSE | 0.678 (2) ± 0.002 | 670/21/308 (2) |
| Waveform40 | ECU | 0.644 (4) ± 0.010 | 172/98/729 (5) |
| | SUN | 0.710 (3) ± 0.001 | 394/42/563 (4) |
| | REDLLA | 0.710 (3) ± 0.001 | 415/44/540 (3) |
| | MC | **0.729** (1) ± **0.003** | **580/8/411** (1) |
| | CDMSE | 0.712 (2) ± 0.005 | 561/48/390 (2) |
| Hyperplane | ECU | 0.541 (5) ± 0.005 | 328/88/583 (5) |
| | SUN | 0.555 (3) ± 0.001 | 507/29/463 (2) |
| | REDLLA | 0.554 (4) ± 0.001 | 533/21/445 (3) |
| | MC | **0.619** (1) ± **0.002** | **783/3/213** (1) |
| | CDMSE | 0.562 (2) ± 0.007 | 426/35/538 (4) |
| Average ranking | ECU | 4 | 4.25 |
| | SUN | 3.25 | 3.5 |
| | REDLLA | 3.75 | 3.5 |
| | MC | **1** | **1** |
| | CDMSE | 2.5 | 2.75 |

their average ranks over all data sets differ by at least one CD as shown in Fig. 12. From the experimental results, we can observe that on the accuracy, all competing approaches have no significant differences, but our CDMSE approach takes the first place in the cases of ulr = 0% and ulr = 50%. And in the case of ulr = 80%, our approach presents the significant advantage compared to the ECU approach. On the time consumption, our CDMSE approach is comparable to the single model-based approaches such as SUN and REDLLA, and it is superior to the ensemble approach ECU. Meanwhile, all of them present the significant advantages compared to the MC approach in all three cases of ulr. These data also reveal the advantages of our approach in the effectiveness and efficiency.

### Recurring Concept Drifting Detection

In this subsection, we want to evaluate whether the concept drifting detection technique in our approach could handle scenarios of recurring concept drifts. Table 6 summarizes the statistics on three evaluation measures for the concept drifting detection in our approach varying with different values of ulr and eight competing

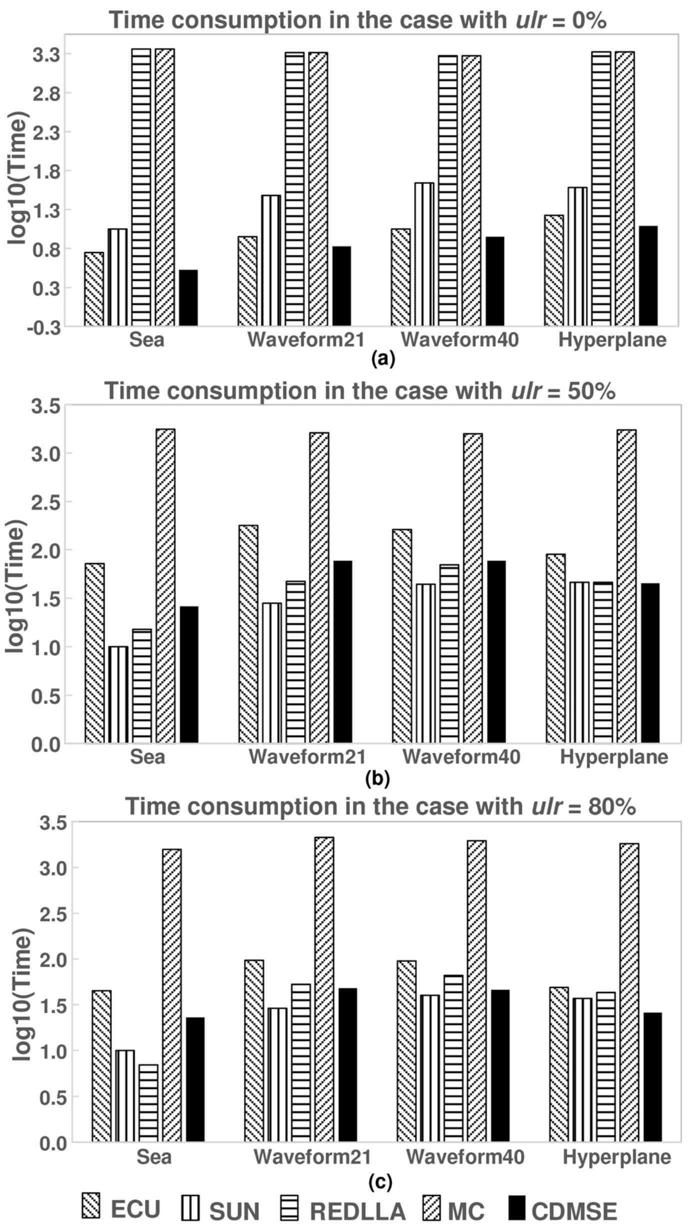

**Fig. 11** Time consumptions between our approach and four classification methods over four data sets

approaches. In this table, if there are no drifts detected in the concept drifting detection, we mark the corresponding values of evaluation measures in '/'. For the simplification, we only give the best statistics of all competing drifting detectors using all labeled data over three base classifiers, including Naïve Bayes, Perceptron and

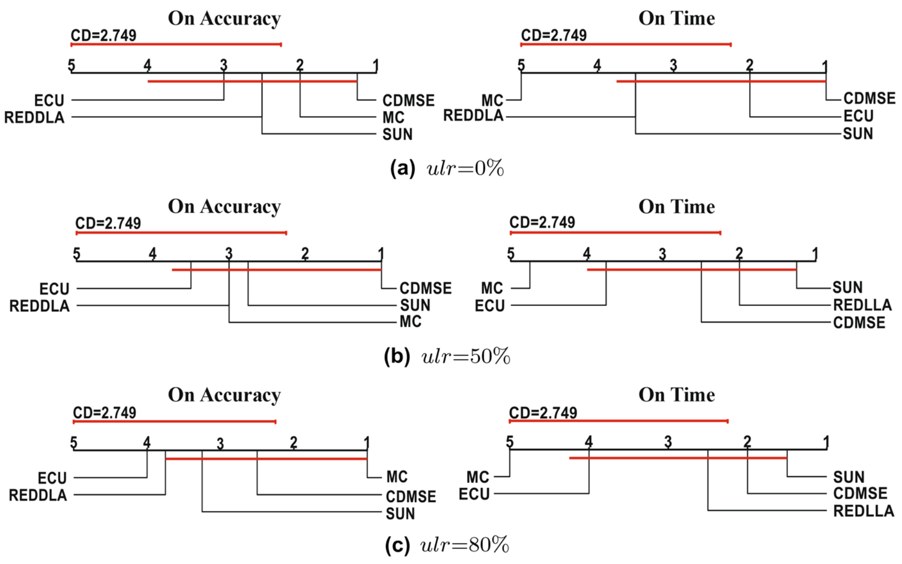

**Fig. 12** The Nemenyi test on four benchmark data sets with different values of ulr

OzaBagASHT based on ensemble of adaptive Hoeffding trees. All competing drifting detectors and the base classifiers are from the open source experimental platform of MOA [15]. From the statistical results, we can see the followings.

First, considering the performance on the evaluation measures of False alarm and Missing, our approach can beat all baselines in the performance of Missing, while it is comparable to others in the performance of False alarm. This is because our approach uses the similarity-based clusters to detect recurring concept drifts instead of the classification error-based approaches in all competing detectors. It is conducive to distinguish recurring concept drifts caused by changes of conditional probabilities of attribute-values in our experimental data, while it can reduce the impact from class labels. Second, considering the performance on the evaluation measure of Delay, our approach wins the four place at least in eight approaches. Third, considering the performance on the evaluation measure of MTFA, our approach wins the first place in eight approaches except on Waveform40. These data validate the concept drifting detection mechanism in our approach is effective even in the case with unlabeled data up to 80%. It is comparable or superior to the detectors in the case with all labeled data.

It is necessary to mention that our CDMSE approach is not the best on the data set of Waveform40 and cannot detect any drifts on the data set of Hyperplane in the case with ulr = 0%. In fact, if the value of $\delta$ decreases, namely in the case of $\delta \leq 0.12$, our approach can beat all competing approaches in the recurring concept drifting detection. The reason is analyzed below. The Waveform40 data set adds 19 irrelevant attributes compared to the Waveform21 data set. More noisy data indicate the increasing of the distance between cluster centers and the radii of corresponding concept clusters. According to Eq. (6), both values of numerator and denominator are increasing, but the increasing ratio of radius is larger than that of distance between cluster center, thus, our approach can distinguish the recurring concept drifts only if the threshold of $\delta$ decreases. Considering the data set of Hyperplane in the case with ulr = 0%, we know that the more labeled data indicate the lower drift rate according to the concept drifting theory in [55]. To detect concept drifts, the threshold of $\delta$ should be less in the handling of Hyperplane without unlabeled data than those with more unlabeled data.

**Table 6** Statistics of drift detection

| Data set | Approach | False alarm | Missing | Delay | MTFA |
|---|---|---|---|---|---|
| Sea | DDM | 0.14 | 98.99 | 709 | 8400 |
| | EDDM | 0.36 | 96.97 | 378 | 4088 |
| | CusumDM | **0.00** | 98.99 | **17** | / |
| | PageHinkleyDM | / | / | / | / |
| | HDDM_W_Test | 1.28 | 97.98 | 284 | 9815 |
| | REDLLA | 45.00 | 92.93 | 257 | 440 |
| | RDDM | / | / | / | / |
| | d3 | / | / | / | / |
| | CDMSE(ulr = 0%) | 0.20 | 66.66 | 170 | 91200 |
| | CDMSE(ulr = 50%) | 0.40 | 78.78 | 229 | 41750 |
| | CDMSE(ulr = 80%) | 3.00 | **58.58** | 371 | 6593 |
| Waveform21 | DDM | **0.00** | 73.74 | 300 | / |
| | EDDM | 0.08 | 96.97 | 0 | 24800 |
| | CusumDM | **0.00** | 73.74 | 169 | / |
| | PageHinkleyDM | 0.00 | 84.85 | 1160 | / |
| | HDDM_W_Test | 1.60 | 66.67 | 166 | 4034 |
| | REDLLA | 79.58 | 1.01 | 380 | 249 |
| | RDDM | 0.4 | 21.22 | 29 | 37650 |
| | d3 | 13.70 | 3.04 | 131 | 1539 |
| | CDMSE(ulr = 0%) | **0.00** | **0.00** | **0.00** | / |
| | CDMSE(ulr = 50%) | 0.80 | **0.00** | **0.00** | 22900 |
| | CDMSE(ulr = 80%) | 0.70 | **0.00** | **0.00** | 23914 |
| Waveform40 | DDM | 0.30 | 93.94 | 433 | 4933 |
| | EDDM | / | / | / | / |
| | CusumDM | 0.00 | 50.51 | 216 | / |
| | PageHinkleyDM | **0.00** | 94.95 | 600 | / |
| | HDDM_W_Test | 0.01 | 17.18 | 202 | 2100 |
| | REDLLA | / | / | / | / |
| | RDDM | 0.8 | 21.22 | 586 | 16475 |
| | d3 | 0.14 | **2.03** | **12** | 1410 |
| | CDMSE(ulr = 0%) | 0.4 | 80.81 | 330 | 1750 |
| | CDMSE(ulr = 50%) | 0.4 | 80.81 | 330 | 1750 |
| | CDMSE(ulr = 80%) | 0.4 | 80.81 | 330 | 1750 |

**Table 6** (continued)

| Data set | Approach | False alarm | Missing | Delay | MTFA |
|---|---|---|---|---|---|
| Hyperplane | DDM | / | / | / | / |
| | EDDM | / | / | / | / |
| | CusumDM | **0.00** | 78.79 | 1107 | / |
| | PageHinkleyDM | / | / | / | / |
| | HDDM_W_Test | 0.87 | 92.93 | 700 | 7911 |
| | REDLLA | 83.95 | 0 | **326** | 236 |
| | RDDM | **0.00** | 94.95 | 1447 | / |
| | d3 | **0.00** | 95.96 | 496 | / |
| | CDMSE(ulr = 0%) | / | / | / | / |
| | CDMSE(ulr = 50%) | 1.80 | 62.6 | 859 | 9855 |
| | CDMSE(ulr = 80%) | 3.70 | **41.41** | 576 | 5354 |

In addition, since approaches of ECU, SUN and MC have no statistical results of concept drifting detection, we now give the accuracy curves to show the performance in the recurring concept drifting environment in the case with ulr = 50%. Figures 13, 14, and 15 draw the curves of the classification accuracy in our approach and competing approaches with the concept drifting detection mechanisms. In these figures, the dotted line indicates the real recurring concept drifting point in the corresponding data set, the solid line indicates the concept drifting point detected by our CDMSE approach. From the experimental results, we can observe the following conclusion. As compared with competing approaches ECU, SUN and MC, our CDMSE approach can quickly adapt to the recurring concept drifts. That is, the classification accuracy in our approach is higher than that of the competing approaches, especially when the concept drifts are detected correctly in our approach. The reason is analyzed below.

Regarding the competing approach ECU, it only adapts to the concept drifts by updating the ensemble model, which leads to the adaptation to concept drifts slowly. However, our CDMSE approach adopts the recurring concept drifting detection mechanism based on the divergence between concept distributions. When the concept drifts occur as the incoming data chunks arrive, our approach can distinguish recurring concept drifts in most cases and update the base model in time. Regarding the competing approach SUN, it adopts a similar concept drifting detection mechanism based on the clusters partitioned in k-Modes, but it is built on a single classifier. Thus, the reaction to the concept drifts is probably more sensitive and causes the unnecessary updating. Therefore, the classification accuracy in SUN is usually lower than ours in the concept drifting detection. Regarding the MC method, it is also susceptible to noise interference due to k-means, which means it is easy to delete a micro-cluster as a concept and cause unnecessary deleting and adding, this may cause that concept drift cannot be correctly identified. All of the above experimental

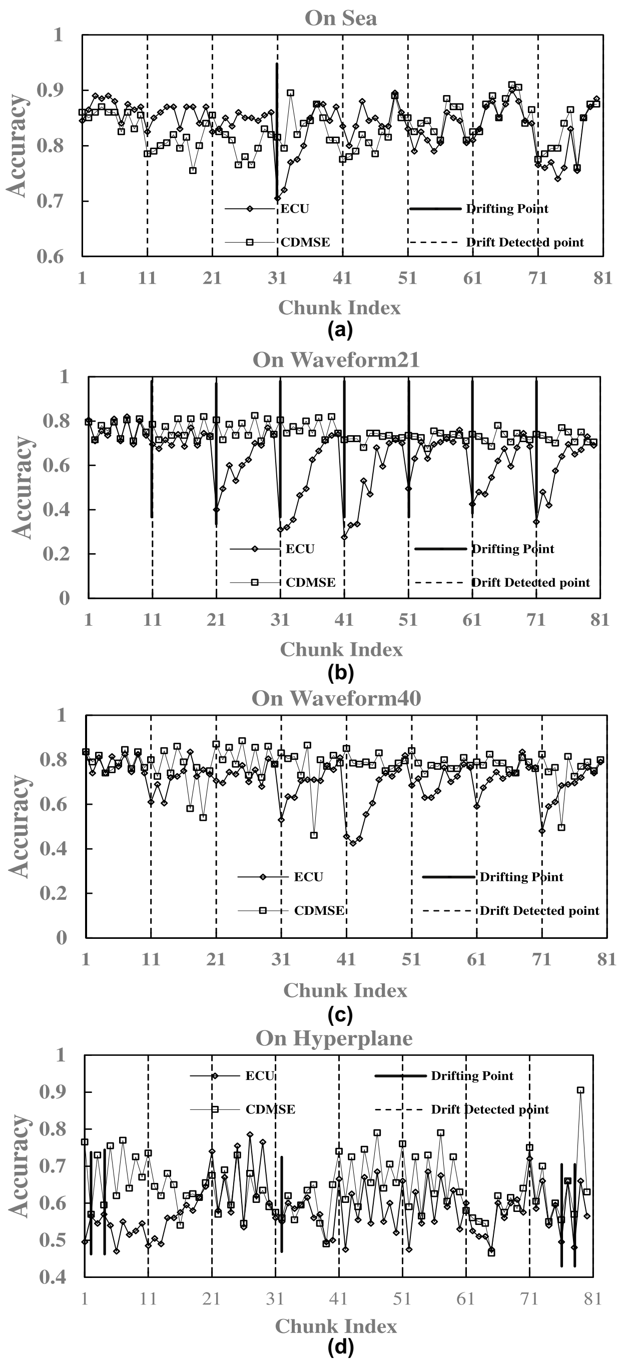

**Fig. 13** Concept drifting detection curves of CDMSE and ECU on four data sets

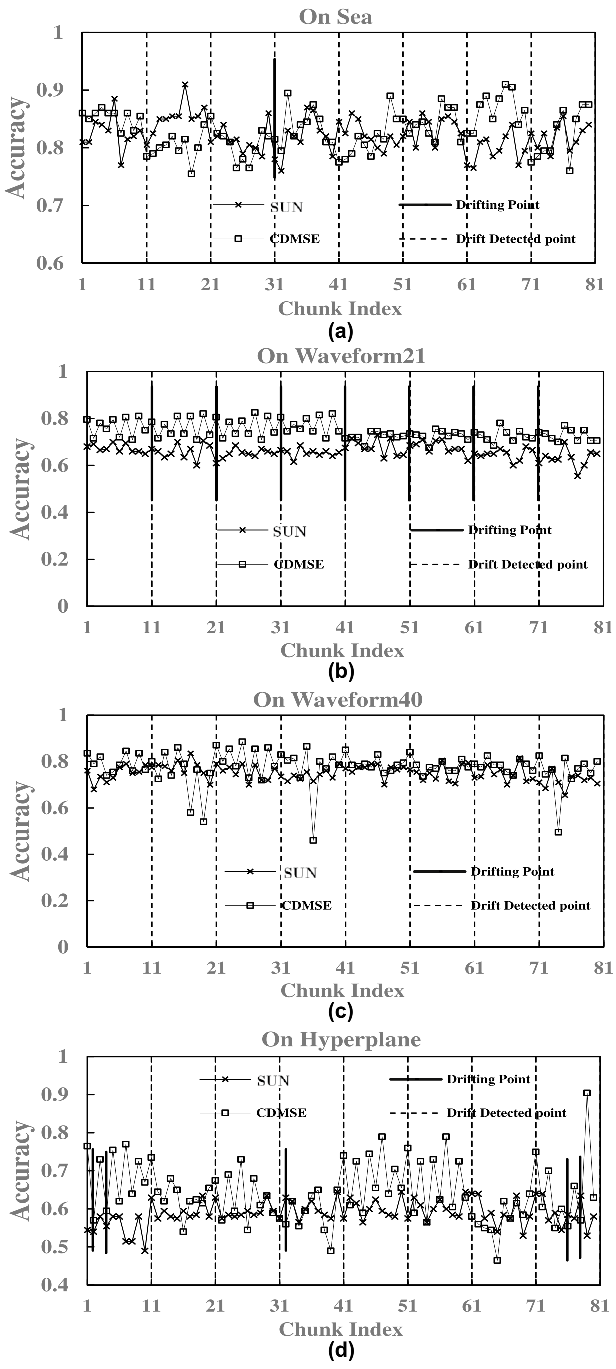

**Fig. 14** Concept drifting detection curves of CDMSE and SUN on four data sets

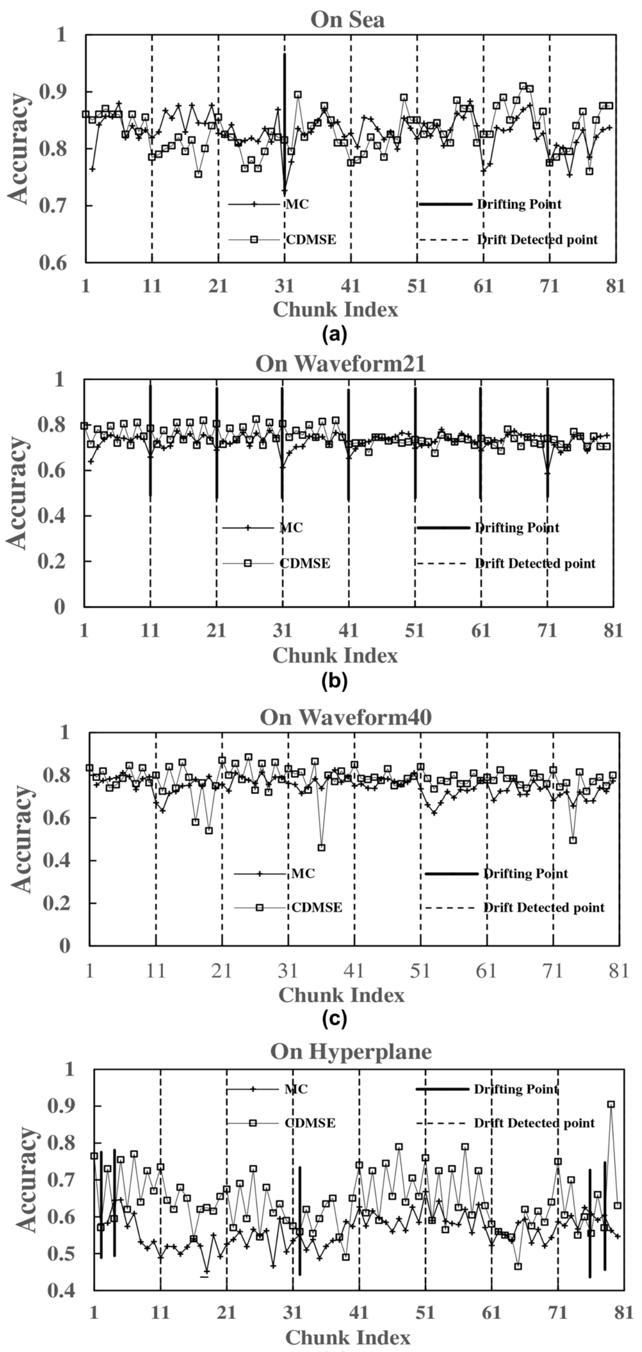

**Fig. 15** Concept drifting detection curves of CDMSE and MC on four data sets

results involved in Figs. 13, 14 and 15 are given in the case of ulr = 50%, we can draw the similar conclusions in cases with the value ulr no more than 80%.

## Application on Real Data Sets

Finally, we also conduct experiments on two real data sets to investigate the effectiveness and efficiency of our approach. These two real data sets are from the UCI Repository[1] as follows. Adult is known as "Census Income" dataset. It contains 48842 samples with two labels, and each sample has 14 features, such as age, education-num and capital-gain. We do not know whether it contains the concept drift. Occupancy data set contains 20560 samples. Each sample has five features, such as temperature, humidity, light and $CO_2$, and these features are used to classify whether the room is occupied or not. Because we cannot know the concept drifts hidden in the real data sets. Thus, we only compare our approach with well-known semi-supervised data stream classification approaches in the classification accuracy and in the time consumption. In the observation of experimental results of Table 7 and Fig. 16, we can see that our approach is not always the first place on these two real data sets in three cases of ulr, but in general, our approach is comparable or superior to competing approaches in the classification accuracy. While our approach has obvious advantages compared to other competing algorithms in the time consumptions.

## Conclusions

In the real-world applications, recurring concept drifting and label missing in data streams impose the challenges for classic data streaming classification approaches due to the lower effectiveness. Therefore, we proposed an ensemble classification method based on the recurring concept drifting detection and model selection for data streams with concept drifts and unlabeled data. First, we built an ensemble model composed of classifiers and clusters. Meanwhile, we divided the data chunk and built the clusters based on the distribution of class labels predicted in the ensemble model. Second, we adopted a new concept drifting detection method based on the divergence between concept distributions and a base model selection method for adapting to the recurring concept drifts. Finally, experimental studies demonstrated that our approach is effective and efficient in the classification on data stream with recurring concept drifts and unlabeled data compared to several well-known competing algorithms. However, how to improve the performance in the handling of gradual concept drifts will be our future work.

[1] https://archive.ics.uci.edu/ml/datasets.php

**Table 7** Performance comparison between our approach and competing ones on real data sets

| Data set | Approach | Accuracy ± variance | Win/tie/lose |
|---|---|---|---|
| In the case of ulr = 0% | | | |
| Adult | ECU | 0.827 (2) ± 0.001 | 0/162/0 (4) |
| | SUN/REDLLA | **0.834 (1) ± 0.001** | **110/13/39** (1) |
| | MC | 0.784 (3) ± 0.001 | 5/4/153 (3) |
| | CDMSE | 0.827 (2) ± 0.001 | 58/48/56 (2) |
| Occupancy | ECU | 0.988 (2) ± 0.001 | 7/85/10 (3) |
| | SUN/REDLLA | **0.989 (1) ± 0.001** | 30/19/53 (2) |
| | MC | 0.988 (2) ± 0.001 | 7/81/14 (4) |
| | CDMSE | 0.988 (2) ± 0.001 | **32/51/19** (1) |
| In the case of ulr = 50% | | | |
| Adult | ECU | **0.822 (1)** ± 0.001 | 0/162/0 (5) |
| | SUN | 0.818 (2) ± 0.001 | 70/9/83 (2) |
| | REDLLA | 0.818 (2) ± 0.001 | 70/6/86 (3) |
| | MC | 0.777 (3) ± 0.001 | 8/4/150 (4) |
| | CDMSE | **0.822** (1) ± **0.001** | **80/45/37** (1) |
| Occupancy | ECU | 0.987 (2) ± 0.001 | 5/88/9 (5) |
| | SUN | 0.981 (4) ± 0.001 | 22/20/60 (4) |
| | REDLLA | 0.982 (3) ± 0.001 | 26/48/28 (3) |
| | MC | **0.989** (1) ± **0.001** | **42/30/30** (1) |
| | CDMSE | 0.987 (2) ± 0.001 | 32/47/23 (2) |
| In the case of ulr = 80% | | | |
| Adult | ECU | **0.807** (1) ± **0.001** | 0/162/0 (5) |
| | SUN | 0.797 (2) ± 0.001 | 61/4/97 (2) |
| | REDLLA | 0.796 (3) ± 0.001 | 58/7/97 (3) |
| | MC | 0.763 (4) ± 0.001 | 31/10/121 (4) |
| | CDMSE | **0.807 (1) ± 0.001** | **79/46/38 (1)** |
| Occupancy | ECU | **0.983** (1) ± **0.001** | 12/71/19 (5) |
| | SUN | 0.973 (4) ± 0.001 | 29/24/49 (4) |
| | REDLLA | 0.98 (3) ± 0.001 | **40/22/40** (1) |
| | MC | **0.983** (1) ± **0.001** | 31/50/21 (3) |
| | CDMSE | 0.981 (2) ± 0.001 | 32/42/28 (2) |

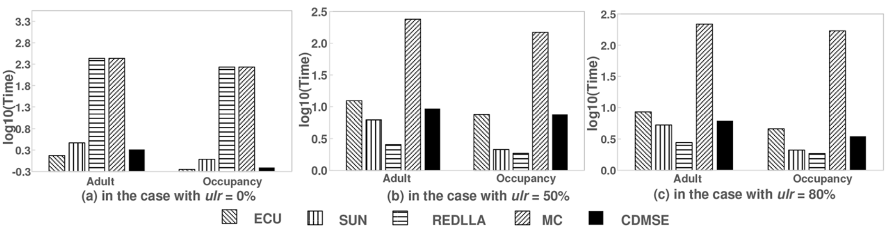

**Fig. 16** Time consumptions of all competing approaches on two real data sets in three cases of ulr

**Acknowledgements** This work is supported in part by the National Key Research and Development Program of China under Grant 2016YFB1000901, the Natural Science Foundation of China under Grants 61976077, 62076085, and 91746209, and the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) of the Ministry of Education under Grant IRT17R32.

## References

1. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Mach. Learn. **23**, 69–101 (1996)
2. Kolter, J.Z., Maloof, M.A., Domingos, P.: Dynamic weighted majority: a new ensemble method for tracking concept drift. In: Proceedings of ICDM'03, Melbourne, FL, United states, pp. 123–130 (2003)
3. Ikonomovska, E., Gama, J., Džeroski, S.: Online tree-based ensembles and option trees for regression on evolving data streams. Neurocomputing **150**, 458–470 (2015)
4. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of KDD'01, Boston, MA, United states, pp. 71–80 (2000)
5. Domeniconi, C., Gunopulos, D.: Incremental support vector machine construction. In: Proceedings of ICDM'01, San Jose, CA, United states, pp. 589–592 (2001)
6. Gama, J., Medas, P., Castillo, G., Rodrigues, P.: Learning with drift detection. In: Proceedings of SBIA Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, pp. 286–295 (2004)
7. Baena-Garća, M., del Campo-Avila, J., Fidalgo, R., Bifet, A., Gavaldà, R., Morales-Bueno, R.: Early drift detection method. In: Proceedings of the 4th international workshop on knowledge discovery from data streams, Berlin, Germany (2006)
8. Zhang, P., Zhu, X., Tan, J., Guo, L.: Classifier and cluster ensembles for mining concept drifting data streams. In: Proceedings0 of ICDM'10, Sydney, NSW, Australia, pp. 1175–1180 (2010)
9. Wu, X., Li, P., Hu, X.: Learning from concept drifting data streams with unlabeled data. Neurocomputing **92**, 145–155 (2012)
10. Masud, M.M., Woolam, C., Gao, J., Khan, L., Han, J., Hamlen, K.W., Oza, N.C.: Facing the reality of data stream classification: coping with scarcity of labeled data. Knowl. Inf. Syst. **33**(1), 213–244 (2012)
11. Loo, H.R., Marsono, M.N.: Online data stream classification with incremental semi-supervised learning. In: Proceedings of CODS'15, Bangalore, India, pp. 132–133 (2015)
12. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of KDD'01, San Francisco, CA, United states, pp. 97–106 (2001)
13. Rutkowski, L., Jaworski, M., Pietruczuk, L., Duda, P.: Decision trees for mining data streams based on the gaussian approximation. IEEE Trans. Knowl. Data Eng. **26**(1), 108–119 (2014)
14. Gama, J., Sebastião, R., Holmes, P.P.: On evaluating stream learning algorithms. Mach. Learn. **90**, 317–346 (2013)
15. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: Moa: massive online analysis. Mach. Learn. Res. **11**, 1601–1604 (2010)
16. Pears, R., Sakthithasan, S., Koh, Y.S.: Detecting concept change in dynamic data streams. Mach. Learn. **97**(3), 259–293 (2014)
17. Almeida, E., Ferreira, C., Gama, J.: Learning model rules from high-speed data streams. In: Proceedings of IJCAI'13, Beijing, China, pp. 10–16 (2013)
18. Kpotufe, S., Orabona, F.: Regression-tree tuning in a streaming setting. In: Proceedings of NIPS'13, Lake Tahoe, NV, United states, pp. 1788–1796 (2013)
19. Shao, J., Ahmadi, Z., Kramer, S.: Prototype-based learning on concept-drifting data streams. In: Proceedings of KDD'14, New York, NY, United states, pp. 412–421 (2014)
20. Kosina, P., Gama, J.: Very fast decision rules for classification in data streams. Data Min. Knowl. Discov. **29**(1), 168–202 (2015)
21. Mena-Torres, D., Aguilar-Ruiz, J.S.: A similarity-based approach for data stream classification. Expert Syst. Appl. **41**(9), 4224–4234 (2014)

22. Rosa, R.D., Orabona, F., Cesa-Bianchi, N.: The abacoc algorithm: a novel approach for nonparametric classification of data streams. In: Proceedings of ICDM'15, Atlantic City, NJ, United states, pp. 733–738 (2015)
23. Frias-Blanco, I., del Campo-Avila, J., Ramos-Jimenez, G., Morales-Bueno, R., Ortiz-Diaz, A., Caballero-Mota, Y.: Online and non-parametric drift detection methods based on Hoeffding's bound. IEEE Trans. Knowl. Data Eng. **27**(3), 810–823 (2015)
24. Chen, D.Z., Yang, Q.L., Liu, J.M., Zeng, Z.: Selective prototype-based learning on concept-drifting data streams. Inf. Sci. **516**, 20–32 (2020)
25. Kolter, J.Z., Maloof, M.A.: Using additive expert ensembles to cope with concept drift. In: Proceedings of machine learning, Bonn, Germany, pp. 449–456 (2005)
26. Fan, W.: Systematic data selection to mine concept-drifting data streams. In: Proceedings of KDD'04, Seattle, WA, United states, pp. 128–137 (2004)
27. Sun, Y., Mao, G., Liu, X., Liu, C.: Mining concept drift from data streams based on multi-classifiers. Acta Automatica Sinica **34**(1), 93–97 (2008)
28. Ramamurthy, S., Bhatnagar, R.: Tracking recurrent concept drift in streaming data using ensemble classifiers. In: Proceedings of international conference on machine learning and applications, Cincinnati, Ohio, pp. 404–409 (2007)
29. Li, P., Wu, X., Hu, X., Liang, Q., Gao, Y.: A random decision tree ensemble for mining concept drifts from noisy data streams. Appl. Artif. Intell. **24**(7), 680–710 (2010)
30. Zhu, Q., Zhang, Y., Hu, X., Li, P.: A double-window-based classification algorithm for concept drifting data streams. Acta Automatica Sinica **37**(9), 1077–1084 (2011)
31. Bardda, J.P., Gomes, H.M., Enembreck, F.: Sfnclassifier: A scale-free social network method to handle concept drift. In: Proceedings of SAC'14, Gyeongju, Korea, Republic of, pp. 786–791 (2014)
32. Islam, M.R.: Recurring and novel class detection in concept-drifting data streams using class-based ensemble. In: Proceedings of PAKDD'14, Tainan, Taiwan, pp. 425–436 (2014)
33. Zahra, A., Kramer, S.: Modeling recurring concepts in data streams: a graph-based framework. Knowl. Inf. Syst. **55**, 1–30 (2017)
34. Anderson, R., Koh, Y.S., Dobbie, G., Bifet, A.: Recurring concept meta-learning for evolving data streams. Expert Syst. Appl. **138**, 112832 (2019)
35. Chiu, C.W., Minku, L.L.: Diversity-based pool of models for dealing with recurring concepts. In: Proceedings of IJCNN'18 (2018)
36. Gomes, J.B., Gaber, M.M., Sousa, P.A.C., Menasalvas, E.: Mining recurring concepts in a dynamic feature space. IEEE Trans. Neural Netw. Learn. Syst. **25**(1), 95–110 (2014)
37. Sakthithasan, S., Pears, R., Bifet, A., Pfahringer, B.: Use of ensembles of fourier spectra in capturing recurrent concepts in data streams. In: Proceedings of IJCNN'15, Killarney, Ireland, pp. 1–8 (2015)
38. Patil, P., Fatangare, Y., Kulkarni, P.: Semi-supervised learning algorithm for online electricity data streams. In: Proceedings of ICAEES'14, Kumaracoil, India, pp. 349–358 (2014)
39. Sethi, T.S., Kantardzic, M., Hu, H.: A grid density based framework for classifying streaming data in the presence of concept drift. J. Intell. Inf. Syst. **46**(1), 179–211 (2016)
40. Silva, C.A.S., Krohling, R.A.: Semi-supervised online elastic extreme learning machine with forgetting parameter to deal with concept drift in data streams. In: Proceedings of IJCNN'19, Budapest, Hungary (2019)
41. Ferreira, R.S., Zimbrao, G., Alvimb, L.G.M.: AMANDA: semi-supervised density-based adaptive model for non-stationary data with extreme verification latency. Inf. Sci. **488**, 219–237 (2019)
42. Haque A., Khan L., Baron M.: SAND: Semi-supervised adaptive novel class detection and classification over data stream, In: 30th AAAI conference artificial intelligence, pp.1652–1658 (2016)
43. Din, S.U., Shao, J.M., Kumar, J.: Online reliable semi-supervised learning on evolving data streams. Inf. Sci. **525**, 153–171 (2020)
44. Li, P.P., Wu, X.D., Hu, X.G.: Mining recurring concept drifts with limited labeled streaming data. ACM Trans. Intell. Syst. Technol. **3**(2), 1–32 (2012)
45. Gonçalves, P.M., Barros, R.S.M.: RCD: a recurring concept drift framework. Pattern Recognit. Lett. **39**(4), 1018–1025 (2013)
46. Hosseini, M.J., Gholipour, A., Beigy, H.: An ensemble of cluster-based classifiers for semi-supervised classification of non-stationary data streams. Knowl. Inf. Syst. **46**(3), 567–597 (2016)
47. Ren, S.Q., Liao, B., Zhu, W., Can, F.: Knowledge-maximized ensemble algorithm for different types of concept drift. Inf. Sci. **430–431**, 261–281 (2018)

48. Gama, J., Sebastião, R., Rodrigues, P.P.: Issues in evaluation of stream learning algorithms. In: Proceedings of KDD'09, Paris, France, pp. 329–337 (2009)
49. Liu, J., Liu, A., Dong, F., Gu, F., Gama, J., Zhang, Z.: Learning under concept drift: a review. IEEE Trans. Knowl. Data Eng. **31**(12), 2346–2363 (2019)
50. Bifet, A.: Classifier concept drift detection and the illusion of progress. In: Artificial intelligence and soft computing, pp. 715–725. CRC Press, Boca Raton (2017)
51. "Weka tool...". http://www.weka.net.nz/. Accessed 8 Mar 2020
52. Severo, M., Gama, J.: Change detection with Kalman filter and CUSUM. In: Proceedings of international conference on discovery science, Barcelona, Spain, pp. 243–254 (2006)
53. Barros, R.S.M., Cabral, D.R.L., Goncalves, P.M., Santos, S.G.T.C.: RDDM: reactive drift detection method. Expert Syst. Appl. **90**(C), 344–355 (2017)
54. Gozuacik, O., Buyukcakir, A., Bonab, H., Can, F.: Unsupervised concept drift detection with a discriminative classifier. In: Proceedings of the 28th ACM international conference on information and knowledge management, Beijing, China (2019)
55. Helmbold, D.P., Long, P.M.: Tracking drifting concepts by minimizing disagreement. Mach. Learn. **14**, 27–45 (1994)

## Authors and Affiliations

**Peipei Li**[1,2] · **Man Wu**[1,2] · **Junhong He**[3] · **Xuegang Hu**[1,2]

Man Wu
manwu@mail.hfut.edu.cn

Junhong He
johnhea@163.com

Xuegang Hu
jsjhuxg@hfut.edu.cn

[1] Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei University of Technology, Hefei 230009, China

[2] School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China

[3] CloudWalk Technology Co., Ltd., Guangzhou, China