



Research Article

Microcluster-Based Incremental Ensemble Learning for Noisy, Nonstationary Data Streams

Sanmin Liu,^{1,2} Shan Xue ,² Fanzhen Liu,² Jieren Cheng ,³ Xiulai Li,^{3,4} Chao Kong,¹ and Jia Wu ²

¹School of Computer and Information, Anhui Polytechnic University, Wuhu 241000, China

²Department of Computing, Macquarie University, Sydney 2109, Australia

³School of Computer Science & Cyberspace Security, Hainan University, Haikou 570228, China

⁴Hainan Hairui Zhong Chuang Technology Co. Ltd., Haikou 570228, China

Correspondence should be addressed to Shan Xue; emma.xue@mq.edu.au

Received 23 October 2019; Revised 26 December 2019; Accepted 1 February 2020; Published 5 May 2020

Guest Editor: Xuyun Zhang

Copyright © 2020 Sanmin Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Data stream classification becomes a promising prediction work with relevance to many practical environments. However, under the environment of concept drift and noise, the research of data stream classification faces lots of challenges. Hence, a new incremental ensemble model is presented for classifying nonstationary data streams with noise. Our approach integrates three strategies: incremental learning to monitor and adapt to concept drift; ensemble learning to improve model stability; and a microclustering procedure that distinguishes drift from noise and predicts the labels of incoming instances via majority vote. Experiments with two synthetic datasets designed to test for both gradual and abrupt drift show that our method provides more accurate classification in nonstationary data streams with noise than the two popular baselines.

1. Introduction

The velocity and voracity with which we are now producing data is making streaming data ubiquitous in real-world applications [1]. For example, intrusion detection [2], credit fraud detection [3], network traffic management [4], and recommendation system [5] all rely on data streams. However, data streams have some unique characteristics that make it more difficult to manipulate. First, the data can be generated at very fast speeds and in huge volumes. Second, there exists concept drift in data streams, and the existing models no longer work as effectively as they once did. Last, physical constraints mean that only a certain amount of knowledge can be used or extracted from a data stream at any point in time and, once elapsed, it can be very difficult to go back and retrieve more knowledge. Thus, data stream mining confronts many challenges.

Revealing the knowledge hidden in data streams is broadly known as data stream mining, which spans data stream classification, clustering, and other data analytics

tasks [6]. Data stream classification is, arguably, the most common analytics task in many practical applications. Due to the time-sequence characteristics, the related research studies of data stream classification confront lots of difficulties. For example, to keep track with concept drift, the model not only needs to be retrained frequently but also its processing and memory overheads must stay low to cope with the velocity and volume of the data. In a traditional data mining scenario, the model would merely need to extract knowledge from a static dataset with a joint distribution function that does not change. However, our model for data stream classification needs to extract knowledge from instances that are generated over time and where the joint distribution function is variable, i.e., in the presence of concept drift [7, 8]. According to many studies, concept drift is the main barrier to data stream classification.

To date, the solutions to classification in nonstationary data stream environment have been based on either online or ensemble learning, and those methods improve the performance of classification. Concerning concept drift in

imbalanced streams data setting, an ensemble learning model was presented with resampling technology [7]. A combined online ensemble method was used to simultaneously consider concept drift and the high-dimension problem [9]. Additionally, in the light of various classification scenarios, many supervised learning approaches recently have been widely explored [10–17], and some have been applied in data stream classification, such as support vector machine (SVM) and Bayesian technique.

In nonstationary streaming data environment, these investigations solved some of the problems, including concept drift, the curse of dimensionality, and imbalanced learning. However, there are still some open problems to be addressed. For example, few studies have considered how to effectively and simultaneously cope with both concept drift and noise in nonstationary data streams. To deal with these problems, we design a new classification approach that constructs microclusters to serve as a pool of base classifiers. Final prediction of incoming instance’s class label is made by a majority vote of the microclusters. At the same time, an incremental learning strategy combined with an ensemble learning and a smoothing operator does the work of adapting the model to concept drift, distinguishing noise, and maintaining stability.

In a word, there exist the three main contributions in our paper:

- (1) A technique for constructing a set of microclusters as base classifiers by redefining the concept of cluster feature previously used in hierarchical cluster analysis. Good classification results can be achieved with nonstationary data streams by combining numerous microclusters. Additionally, microcluster combined with incremental learning is a very convenient way to absorb new knowledge and keep track of concept drift.
- (2) A smoothing strategy designed to shift the centroids of microcluster and control the balance between historical and new instances. This approach makes the best use of historical knowledge and can also overcome problems with a shortage of drifted data.
- (3) A majority vote strategy and an incremental learning enhance the stability and adaptability of the model in nonstationary data streams with noise. Thus, the proposed model leverages the advantages of both ensemble and incremental learning to maintain high accuracy in class label prediction.

This paper is organized as follows. The background work is discussed in Section 2, and then Section 3 outlines the basic concept. Section 4 describes the proposed model and provides a complexity analysis of the algorithm. In Section 5, experimental schema and results are illustrated. Section 6 describes conclusion and future plans.

2. Related Work

An excellent data stream classification approach has the ability to learn incrementally and adapt to concept drift as

well [18]. In general, two important kinds of incremental learning method are concerned: instance-incremental learning [19, 20], which learns an instance at a time, and batch-incremental learning [21], which learns from instance set once. In the instance-incremental learning group, Cramer et al. [19] developed an online passive-aggressive algorithmic (PA) framework based on SVM that forces the classification hyperplane to move to satisfy the minimum loss constraint when the classifier misclassifies an instance. This framework has been widely explored for many practical settings [22, 23]. In work [24], it presented the instance-incremental method with weighted one-class SVM that could solve gradual drift in nonstationary data streams. Instance-incremental learning has also been based on extreme learning machine as a way to boost classification speeds [25]. When data stream is stable, incoming instance is used to update the classifier; however, when concept drift happens, a weakly performing classifier is deleted. This is a very flexible approach to classifying real-time data streams. In the batch-incremental learning domain, Lu et al. [26] provided a novel dynamic weighted majority approach to deal with imbalance problems and concept drift. This method uses dynamic weighted ensemble learning to keep the classification model stable and batch-incremental learning to track concept drift.

Between the two modes, instance-based incremental learning is more flexible and scalable for real-time data stream classification. It is also a more suitable approach for environments where it is difficult to label instances and understand concepts in advance [20]. Hence, we turn our attention to instance-incremental learning for the remainder of this paper, using the simple term incremental learning, hereafter.

The impetus for studying ensemble learning in conjunction with data stream classification came from a desire to improve classification model’s stability [27–31]. These models include base classifier set and merged method which combines the base classifier’s output into a final output by the ensemble. SEA algorithm [29] is one of the early ensemble methods. When the SEA ensemble model is not full, each newly arriving data chunk is used to build a new base classifier. If the limit has been reached, the new classifier is still constructed for every newly arriving data chunk, but it replaces the classifier with the worst performance. According to a majority vote policy, the ensemble method, SEA, outputs the final predictions. Another similar work is weighted ensemble method based on accuracy [30], where the important point is to allocate a weight to each base classifier that is an estimate of its accuracy on the newest data chunk. This idea suggests that the newest data chunk could represent the target concept with high probability, so the classifiers with higher accuracy should be given more importance. Also, when the maximum ensemble scale has been reached, a base classifier with the worst performance is deleted and a new base classifier joins into the ensemble model. Another iterative ensemble method was developed based on boosting and batch-incremental learning [31]. This method adds a suitable number of base classifiers to the classification model with each newly arriving data chunk,

instead of adding just one. The experimental results suggest that the iterative boosting ensemble classification method is a promising way to perform classification task in nonstationary data stream environment. Beyond concept drift, imbalanced class distributions are another challenge with data stream classification that can be tackled with ensemble learning. Zhang et al.'s [27] method of dealing with this problem is a two-pronged approach. The first tack is to divide the majority into N subsets of roughly the same size as the minority and then construct N new balanced training subsets from the minority and divided subsets. Next, the ensemble model is created using a neural network with backpropagation as the base learning algorithm. The base classifiers' diversity is one of the important factors of learning system. Hence, Jackowski [32] introduced the idea of two error trend diversity measurements: pair errors and pool errors, to find and keep track with concept drift in streaming data setting. Experiments with this model show that the diversity measurements can not only be used to enhance the ensemble model's performance but also to hold effectively the scale of ensemble model. Based on the above analysis, we think that ensemble learning is currently the most promising research direction for data stream classification.

From this review, we distill several observations: incremental learning can dynamically reveal new knowledge in data streams. Ensemble learning can improve the stability of classification models for nonstationary data streams. The suitable algorithm can enhance a classification model's flexibility. These three observations form the basis of three integrated strategies in our method for simultaneously tackling concept drift and noise.

3. Basic Concept and Problem Definition

This section firstly begins with a description of the basic concepts used in this paper, and then a detailed analysis of the research problem is explored.

3.1. Data Stream. According to the related studies, in this paper, we think that data stream consists of a series of labeled instances, namely, $S = \{X_1, X_2, \dots, X_t, \dots\}$, where $X_t = (X, y)$, in which X stands for a feature vector which represents an instance characterizing the features of an object and y is X_t 's class label. When y is +1, X_t represents positive instance. On the contrary, X_t is negative instance.

According to the above definition, we explore a mapping function $f: X \rightarrow y$ with high accuracy which stands for classification model that can output the incoming instance X 's class label. Only supervised learning is considered in this paper. Therefore, the classification model f is constructed from a labeled dataset, and, once built, it can output the class label +1 or -1 for the incoming instance. In addition, for the purposes of this paper, the real label is acquired after the mapping function f outputs the prediction of incoming instance.

3.2. Concept Drift. According to the work [33], when a joint probability distribution P of data changes evolving over

time, there exists concept drift. In other words, $P_t(X, y) \neq P_{t+1}(X, y)$, where the subscript t stands for the time stamp, X suggests the vector which represents the value of feature attribute, and y is a class label. According to the changing rate of concept, gradual drift and abrupt drift [34] are discussed. Generally speaking, gradual drift is a slower rate of change from one concept to another one, and it is illustrated in Figure 1(a). When the distribution P_t is abruptly different from the distribution P_{t+1} at $t + 1$, we say that abrupt drift occurs and it is seen in Figure 1(b). In Figure 1, the difference between gradual drift and abrupt drift is clearly found, and these two kinds of drift are concerned in this paper.

3.3. Problem Definition. Noisy instances and concept drift appear to have similar distributions in nonstationary data streams. It is, therefore, critical to differentiate noise from concept drift and that is the motivation of this paper to build a classification model that can find and keep track with concept drift in nonstationary streaming data with noise. Meanwhile, in order to catch concept drift, the classification model should be updated by incremental learning. The research problem is demonstrated in Figure 2.

From Figure 2, we understand clearly the problem definition of this paper and identify the noisy instance in nonstationary streaming data. The dotted circle represents a microcluster, and the dotted straight line suggests the distribution for instances in Figure 2. The current case is shown in Figure 2(a). When time goes on, the instance is coming and the microcluster is updated at time stamp $t + 1$ as seen from Figure 2(b), which represents the case of incremental learning. At time stamp $t + 2$, in Figure 2(c), the incoming instance with positive class label lies in the old microcluster with a different class label. In this case, the new instance is regarded as a noisy instance and will be discarded; this is why this instance no longer exists at time stamp $t + 3$. In Figure 2(d), the incoming instance forms a concept drift and leads to a new microcluster construction.

Based on the above analysis, our solution involves three strategies to deal with the research problem as illustrated in Figure 2: incremental learning to track concept drift; ensemble learning to enhance the model's stability; and microclustering method to distinguish drift from noise and make the final label predictions. In the next section, we outline these strategies in detail and discuss the three scenarios illustrated in Figure 2.

4. Adaptive Incremental Ensemble Data Stream Classification Method

This section describes microcluster and data stream classification model, followed by the corresponding algorithm.

4.1. Definition of Microcluster. Microclusters as classifiers in our model are constructed by cluster features, which is a technique that was originally developed as part of hierarchical cluster analysis [35]. The structure of cluster feature is

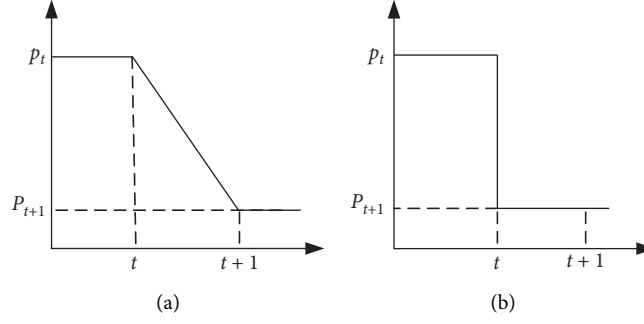


FIGURE 1: The two types of concept drift. (a) Gradual drift. (b) Abrupt drift.

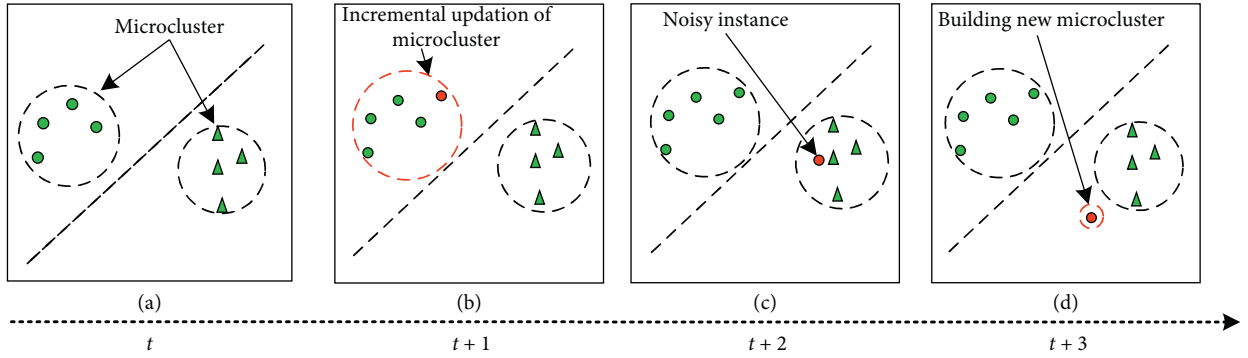


FIGURE 2: A demonstration of problem definition. In data stream, (a) microclusters (shape: dotted circle) are developed by historical instances (color: green) with positive class (shape: circle) and negative class (shape: triangle); (b) when new instance (color: red, shape: circle) comes, microcluster is updated (color: red, shape: dotted circled), and (c) the old microcluster involves noisy instance (color: red, shape: circle), and (d) concept drift (color: red, shape: circle) is detected and a new microcluster (color: red, shape: dotted circle) is built.

defined as $CF = \langle SS, LS, n \rangle$. Based on the cluster feature, we give the definition of microcluster used in this paper.

Definition 1. Microcluster (MC) is represented as $\langle SS, LS, n, Cid, CL, \alpha \rangle$, where SS and LS are used to compute the boundary of MC that SS denotes the square sum of the attributes of the instances in MC as calculated in equation (1) and LS is a vector saves the sum of each attribute as in equation (2), n suggests the number of instances, Cid presents MC 's centroid which changes over time as shown in equation (3), CL is MC 's class label, and α counts the number that MC correctly classifies incoming instance and α is initiated as 0.

$$SS = \sum_i^n \sum_j^l x_{ij}^2, \quad (1)$$

$$LS = \left(\sum_i^n x_{i1}, \dots, \sum_i^n x_{ij}, \dots, \sum_i^n x_{il} \right), \quad (2)$$

where l is the dimension of the instance.

$$Cid = (1 - \sigma) \times Cid_{t-1} + \sigma \times (LS/n), \quad (3)$$

where Cid_{t-1} is MC 's centroid on the previous time stamp $t - 1$ and $\sigma \in [0, 1]$ stands for smoothing parameter.

The size of MC is represented by cluster's radius r which is calculated as follows:

$$r = \sqrt{\frac{SS}{n} - \left\| \frac{LS}{n} \right\|^2}, \quad (4)$$

where $\|LS/n\|$ represents the length of vector.

4.2. Data Stream Classification Model Based on Microcluster.

Classification model consists of three phases: classification, incremental learning, and updating. A framework of the model is given in Figure 3. The processes and calculations are presented in detail in this part and summarized into the corresponding algorithm presented as Algorithm 1.

4.2.1. Phase 1 (Classification): The k -Nearest Microclusters Classify the Incoming Instance.

When an incoming instance arrives, Euclidean distance is computed between the incoming instance and each microcluster in pool. Based on Euclidean distances, the k -nearest microclusters are selected, and then each microcluster will assign its own label to the incoming instance. According to equation (5), the final label of incoming instance is voted by the merged method.

$$y = \arg \max_j \sum_{i=1}^c \sum_{i=1}^k f_i(x), \quad (5)$$

where k stands for the number of microclusters participating in the classification and c denotes the number of class.

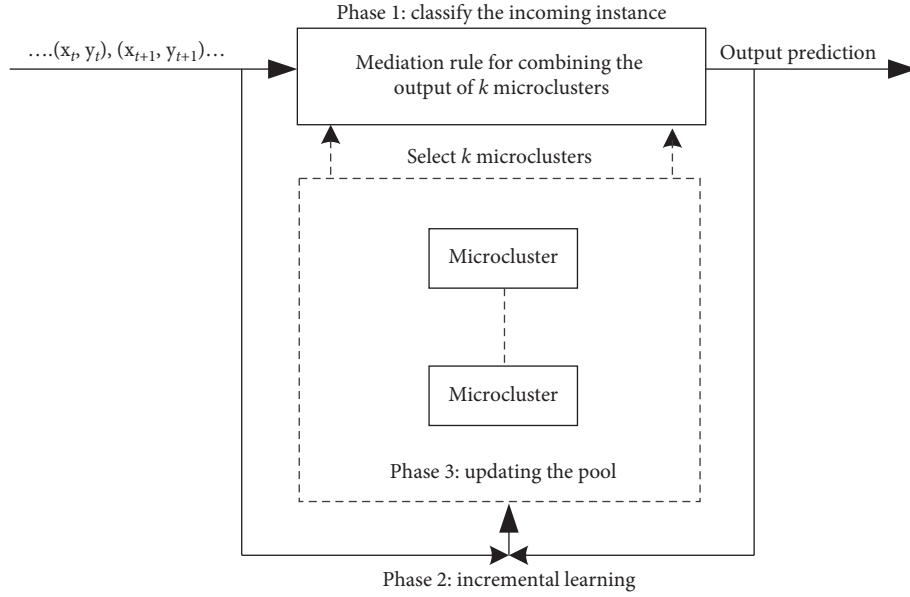


FIGURE 3: The framework of data stream classification model.

Input: The instances $S = \{X_1, X_2, \dots, X_t, \dots\}$,
the pool maximum limit M , and
the smoothing parameter σ .

Output: The pool of microcluster P^*

- (1) $P^{(0)} = \cup_{i=1}^L MC_i \leftarrow$ the pool of initial microclusters (MC) which is formed by k -means
- (2) **for** each instance $X_t = (X, y)$ **do**
 - (3) **Phase 1: Classification**
 - (4) $d \leftarrow$ distance between X and MC
 - (5) $MC^{(t)} \leftarrow$ select the k -nearest microclusters to classify the instance X
 - (6) $\hat{y} \leftarrow$ the predicted class label of instance X gained by majority vote in equation (5)
 - (7) $\alpha \leftarrow$ update the parameter of the k -nearest microcluster
 - (8) **Phase 2: Incremental Learning**
 - (9) **if** Scenario 1 **then**
 - (10) $MC^{(t)*} \leftarrow$ update the structure of nearest microcluster by equations (1)–(3) and the number of the instances in microcluster will be incremented by 1
 - (11) **else if** Scenario 2 **then**
 - (12) $X \leftarrow$ consider the instance as a noisy point and neglect it
 - (13) **else if** Scenario 3 **then**
 - (14) $MC_X^{(t)} \leftarrow$ build a new microcluster on instance X
 - (15) **Phase 3: Updating Pool**
 - (16) **if** $L < M$ **then**
 - (17) $P^{(t)} = P^{(t)} + MC_X^{(t)}$
 - (18) $L = L + 1$
 - (19) **else**
 - (20) $MC_{worst}^{(t)} \leftarrow$ the worst microcluster
 - (21) $MC_X^{(t)} \leftarrow$ replace $MC_{worst}^{(t)}$
 - (22) **end if**
 - (23) **end if**
 - (24) **end for**
 - (25) **return** $P^* \leftarrow$ microcluster pool at required time stamp t

ALGORITHM 1: MCBIE.

Once incoming instance is classified, microcluster is immediately updated. If the final prediction is correct, i.e., if all the microclusters who voted have the same class label as the final prediction, the value of α increases by 1; otherwise, it decreases by 1.

4.2.2. Phase 2 (Incremental Learning): The Nearest Microcluster Will Be Updated Based on the Incoming Instance. Following the first-test-and-then-train principle, the nearest microcluster is immediately updated to ensure the model quickly adapts to the new concept or the new

microcluster is constructed in this phase, which is depicted in Figure 4.

Scenario 1: *when incoming instance's label is the same as the nearest microcluster's label*, incoming instance is used to retrain this microcluster. The terms SS , LS , and Cid of the nearest microcluster are recalculated by equations (1)–(3). The number of instances in this microcluster is incremented by 1. The radius of microcluster is also updated by equation (4). This scenario is shown in Figure 4(a). As a matter of fact, when the incoming instance drops into the nearest microcluster, we carry out the same operation, that is, the incoming instance is merged into the nearest microcluster.

Scenario 2: *incoming instance's label varies from the nearest microcluster's label and incoming instance lies inside the boundary of the nearest microcluster*, as seen from Figure 4(b). In this paper, there exists the fundamental assumption that two adjacent instances are highly likely to represent the same concept, i.e., the probability that they share the same class label is very high. According to the fundamental assumption, the incoming instance will be treated as noise and deleted.

Scenario 3: *in contrast to Scenario 2, incoming instance's label is different from the nearest microcluster's label and incoming instance does not drop into the nearest microcluster*, as shown in Figure 4(c). This scenario suggests that incoming instance is derived from the different joint probability distribution. Under this circumstance, we think new concept happens, and a microcluster will be constructed with incoming instance by the method described in Section 4.1. Because there is only one instance in this new microcluster when it is constructed, its label CL will be the same as the incoming instance and its centroid will be the incoming instance itself. The terms SS and LS of the new microcluster are computed by equations (1) and (2), and the value of α is 0.

4.2.3. Phase 3 (Updating): The Pool of Microcluster Is Updated. As time passes, new microclusters are continuously being created and, eventually, the pool will reach its limit. Once full, the microcluster with the worst performance will be replaced with new microcluster. By this cyclical update, the classification model can effectively catch concept change, and it leads to improve the classification accuracy. Generally speaking, the smaller the value of α , the worse the performance of the microcluster. Therefore, the microcluster with the smallest α is selected for replacement.

4.3. Algorithm and Complexity Analysis. In summary of the above phases and scenarios in data stream classification model, the algorithm of microcluster-based incremental ensemble classification named as MCBIE is expressed in Algorithm 1.

The algorithm of MCBIE includes three phases which achieve three functions, namely, classification, incremental learning, and updating pool. Line 1 is to train the initial

microcluster and build a pool of microclusters. Lines 3 to 6 achieve the classification for an incoming instance X and update the performance of the microcluster. According to the three different scenarios, the function of Phase 2 is accomplished in lines 7 to 12. Finally, the size of base classifier reaches the upper-bound M , the worst microcluster will be deleted, and the new microcluster is added to microcluster pool. On the contrary, the new microcluster is directly put into microcluster pool. It is illustrated in lines 13 to 19.

In terms of complexity, through the analysis of Algorithm 1, we know the core operation included by the algorithm MCBIE is to calculate the distance in classification phase. The complexity here depends on mainly two aspects: the dimensions of the instance (l) and the number of microclusters as base classifier (M) in the ensemble model. Thus, the presented algorithm's time complexity is approximately $O(l \cdot M)$. In the presented algorithm, the previous instances are not reserved over time and the statistical information of microcluster is recorded, such as SS , LS , and Cid , which can save the storage memory by this way.

5. Experiments

5.1. Datasets. To evaluate MCBIE, we conduct simulation experiments with two synthetic datasets. The two datasets selected are the Hyperplane data stream and the SEA data stream taken from Massive Online Analysis (MOA) [36]. Hyperplane data stream is designed to test for gradual drift, while SEA data stream is designed to test for abrupt drift. These are the most popular datasets in the data stream classification domain. Further details are as follows.

Hyperplane data stream [37]: in the l -dimensional space, a hyperplane includes the point set X which satisfies $\sum_{i=1}^l a_i x_i = a_0 = \sum_{i=1}^l a_i$, where x_i represents the i -th dimension of X . Instances for which $\sum_{i=1}^l a_i x_i \geq a_0$ represent positive class, and instances for which $\sum_{i=1}^l a_i x_i < a_0$ represent negative class. A hyperplane in l -dimensional space may slowly rotate by changing the parameters for simulating time-changing concepts. In this paper, the value of l is 10 and there are 6 attributes with concept drift, and it generates 20,000 instances. Three different noise ratios (respectively, 20%, 25%, and 30%) are injected into data stream.

SEA data stream [29]: the instances in this data stream are generated from three attributes with continuous values $x_1, x_2, x_3 \in [0, 10]$. When it satisfies $x_1 + x_2 \geq \theta$, the instance is positive class; otherwise, the label of instance is negative. To simulate concept drift, the threshold value $\theta = \{8, 9, 7, 9.5\}$ will change over time. It generates 5000 instances with each threshold value, and the whole SEA data stream includes 20,000 instances. SEA data stream with two different noise ratios (20% and 30%) is applied in this experiment to test the abrupt drift.

5.2. Baselines. The PA algorithmic framework [19] and Hoeffding tree [38] are selected as baselines to compare with the presented method MCBIE, and these two approaches are

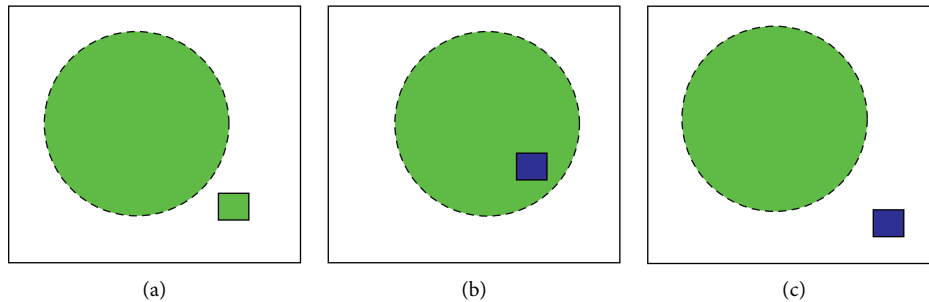


FIGURE 4: Three different scenarios: (a) incoming instance's label is the same as the nearest microcluster's label; (b) incoming instance's label varies from the nearest microcluster's label, which represents noisy instance; and (c) incoming instance as a new concept does not drop into the nearest microcluster and its label is different from the nearest microcluster's label. Note: the color represents the class label, the rectangle suggests the incoming instance, and the circle represents the nearest microcluster.

frequently chosen as the benchmark in many studies [20, 22, 23, 38]. Moreover, as a well-known classical algorithm, the Hoeffding tree algorithm is integrated into the MOA platform [36]. Therefore, we have followed suit in our paper.

The PA algorithmic framework [19] is an online incremental learning framework for binary classification based on SVM. Given instance X , the classification model outputs the prediction as follows:

$$\hat{y} = \text{sign}(\vec{w} \cdot X), \quad (6)$$

where \vec{w} represents a vector of weights and \hat{y} is the prediction of instance X .

After the \hat{y} is output, it acquires the ground truth class label y and computes a loss value resulting from the following equation:

$$l_s = \max\{0, 1 - y \cdot (\vec{w} \cdot X)\}. \quad (7)$$

The vector of weights \vec{w} is then updated using

$$\vec{w} = \vec{w} + \tau \cdot y \cdot X, \quad (8)$$

where $\tau \geq 0$ is a Lagrange multiplier, whose value is calculated by equation (9) in three different methods, namely, PA, PA-I, and PA-II.

$$\begin{aligned} \tau &= \frac{l_s}{\|X\|^2} \text{ (PA)}, \\ \tau &= \min\{C, (l_s/\|X\|^2)\} \text{ (PA-I)}, \\ \tau &= \frac{l_s}{\|X\|^2 + (1/(2 \cdot C))} \text{ (PA-II)}, \end{aligned} \quad (9)$$

where C is a positive parameter and referred to as aggressiveness parameter of the algorithm. A detailed outline of the derivation procedure can be found in [19].

Hoeffding tree [38] is a decision tree for online learning from the high-volume data stream, and it is built from each instance in constant time. According to Hoeffding bound, we can estimate the number of instances which are needed to build the tree node. The Hoeffding bound has nothing to do with the distribution function that

generates the instances. Moreover, the Hoeffding bound is used to construct Hoeffding tree which is approximated to the one produced by batch learning. In the light of its incremental nature of the Hoeffding tree, it is used widely in data stream classification.

5.3. Experiment Setup. Following the first-test-and-then-train principle [39], each incoming instance is first tested, and then the model is retrained with the incoming instance under an incremental paradigm. To assess the classification model's performance in this paper, the classification accuracy is computed every one hundred instances during the process of data stream classification.

Both our MCBIE and the baselines are initialized on the first 100 instances, and the model resulting from that initialization is used to predict the following instance in data stream. In MCBIE, we use these 100 instances to train 6 initial microclusters as base classifiers by using the k -means algorithm. At every time stamp, the three nearest microclusters of each incoming instance are selected to assert the label information. The maximum scale of microcluster pool is 30, and once full, a new microcluster which takes the place of the worst-performing microcluster joins in the pool. We use Weka package to implement the MCBIE algorithm. Hoeffding tree algorithm (named as HT) is run in MOA platform with the parameters set to their default values. PA, PA-I, and PA-II with Gaussian kernel are executed in MATLAB and the constant C is equal to 1.

5.4. Experiment Result and Analysis. The simulation experiments are designed to evaluate MCBIE in two sides. First, we want to assess the sensitivity of the smoothing parameter σ ; second, we want to justify the feasibility and validity of MCBIE.

5.4.1. Experiment 1: Sensitivity Analysis of the Smoothing Parameter. Following the experimental setup in Section 5.3, we verify the function of smoothing parameter σ in MCBIE from 0.1 to 1. When the smoothing parameter σ is either too big or too small, the MCBIE's average accuracy and corresponding standard deviation do not reach the desired result on the Hyperplane data stream and SEA data stream.

TABLE 1: Average classification accuracy and standard variance.

	Hyperplane data stream			SEA data stream	
	20% noise	25% noise	30% noise	20% noise	30% noise
PA	0.622 ± 0.056	0.581 ± 0.054	0.548 ± 0.048	0.661 ± 0.061	0.577 ± 0.056
PA-I	0.678 ± 0.049	0.626 ± 0.050	0.582 ± 0.045	0.726 ± 0.060	0.617 ± 0.061
PA-II	0.646 ± 0.053	0.599 ± 0.053	0.558 ± 0.047	0.688 ± 0.062	0.595 ± 0.059
HT	0.582 ± 0.067	0.567 ± 0.061	0.556 ± 0.060	0.705 ± 0.068	0.625 ± 0.059
MCBIE	0.696 ± 0.051	0.648 ± 0.046	0.618 ± 0.047	0.708 ± 0.053	0.632 ± 0.046

Through the observation and analysis, we find the smoothing parameter σ could regulate the balance between the historical and new instances used to compute the centroid of the microcluster. When $\sigma = 0$, the centroid of the microcluster will not move, and only its radius changes. On the contrary, when $\sigma = 1$ reaches the maximum value, the microcluster’s centroid is a mean of instances. It suggests all instances have the same importance to the centroid. However, because concept drift will occur in nonstationary data stream environment, instance at different time stamps should have different contributions to the centroid of microcluster. Experiment results justify this viewpoint. According to the analysis of experiment results, a conclusion is made that the best value of σ is located at an interval $[0.6, 0.7]$; hence, we chose $\sigma = 0.65$ for subsequent experiments.

5.4.2. Experiment 2: Feasibility and Validity of MCBIE.

All the experimental results with both the Hyperplane and SEA data streams are shown in Table 1. At the same time, the maximum value in each column is marked in bold.

From Table 1, we see the average accuracy of MCBIE reaches the highest value of 69.6%, 64.8%, and 61.8% on Hyperplane data stream with the noise ratio of 20%, 25%, and 30%, respectively. The corresponding standard variance of the three average accuracies is 0.051, 0.046, and 0.047, and the standard variances about accuracy are relatively low compared with the baselines. On average, MCBIE provides the most accurate classifications with the least standard variance among all the baselines with the Hyperplane data stream. On the SEA data stream, the average classification accuracy of MCBIE is 70.8% at 20% noise and 63.2% at 30% noise, respectively. Again, the standard variances are the smallest compared to all baselines, which demonstrate MCBIE’s stability in nonstationary data streams with noise. Based on the above experiment results, we may draw a conclusion that MCBIE is an effective and well-performing classification approach.

Through the further analysis of experiment results in Table 1, some interesting phenomena exist. As the noise ratio grows, the performance of MCBIE is improved to a greater degree than the other methods. For instance, with a noise ratio of 20% on the SEA data stream, MCBIE ranks only the second lead behind PA-I. However, at 30% noise, MCBIE becomes the most accurate model. Given the same noise ratios, the experiment results show that the classification model on SEA data stream performs better than on Hyperplane data stream. This suggests it is more difficult for classification model to learn knowledge from gradual drift

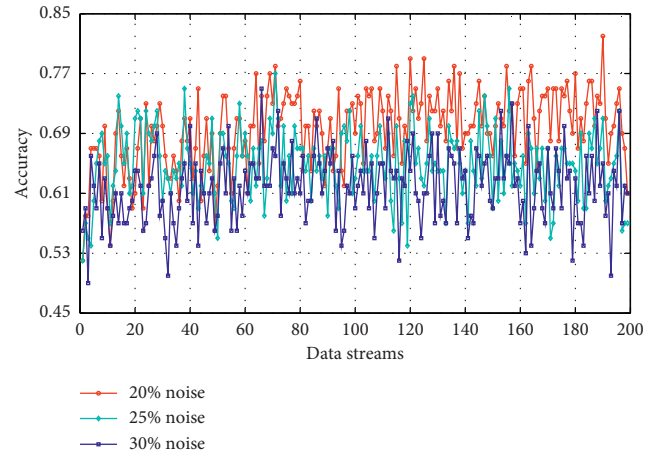


FIGURE 5: The accuracy curve for the MCBIE method with Hyperplane data stream.

than from abrupt drift. Of all the baselines, PA-I provides the best performance, which indicates that selecting an appropriate learning ratio τ is very important for incremental learning. The Hoeffding tree baseline has the largest standard variance, and it shows that Hoeffding tree has instability.

Last but not least, we want to show MCBIE adapts well to concept drift; Figures 5 and 6 illustrate accuracy curve for the MCBIE method with Hyperplane and SEA data streams. Figure 5 suggests that MCBIE can tackle concept drift in time on the Hyperplane data stream with the different noise ratios. When concept drift occurs, the curve plot in Figure 5 sharply descends and it indicates the concept included by the model is inconsistent with the current concept. MCBIE’s accuracy decreases when concept changes. However, the performance of MCBIE improves immediately after the model is retrained and updated with the incoming instance through incremental learning. The intensity of ascent and descent in Figure 5 reflects that the classification model has the ability to catch concept drift. In Figure 6, we easily understand that the similar phenomena are presented with the SEA data stream.

To demonstrate MCBIE’s superiority, based on the above analysis, we choose the two best methods MCBIE and PA-I to illustrate the ability to perform the prediction task in streaming data setting with noise and concept drift. The accuracy curve is plotted in Figures 7 and 8. From Figure 7, the accuracy curve suggests that these two methods have the ability to keep track with concept drift and shows clearly that our method is superior to the PA-I in terms of accuracy over the Hyperplane data stream with the three different noise

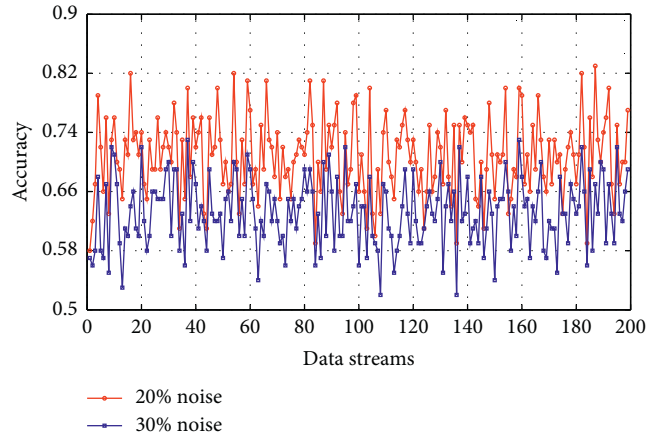
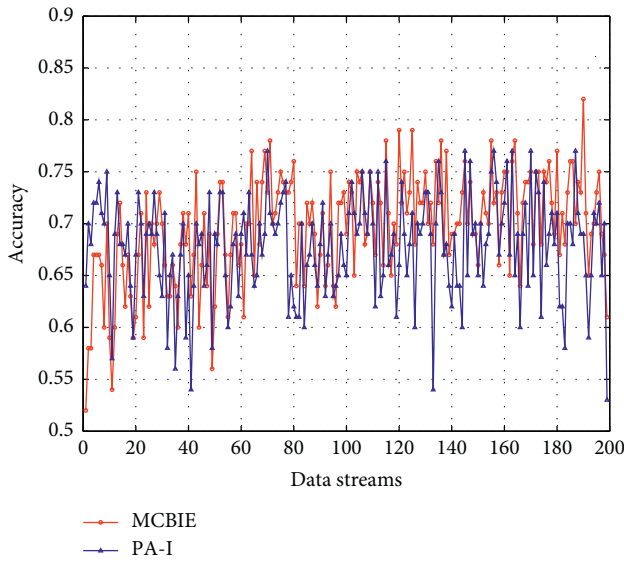
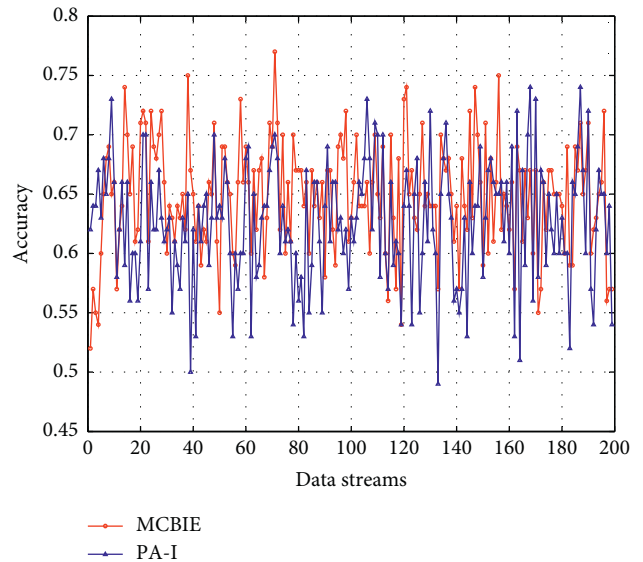


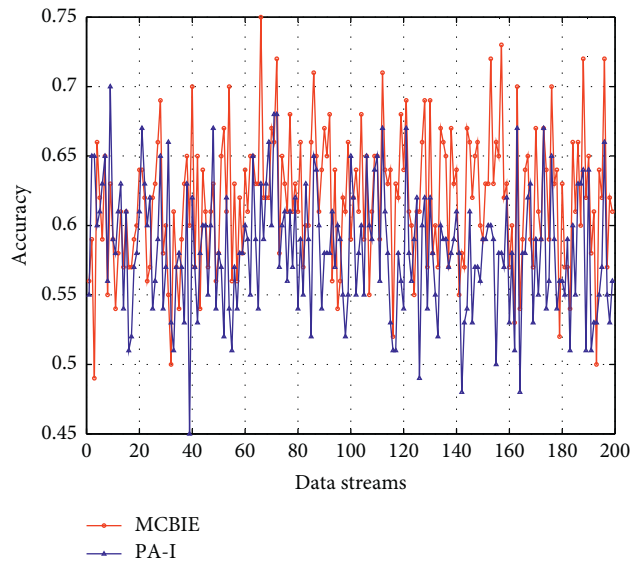
FIGURE 6: The accuracy curve for the MCBIE method with SEA data stream.



(a)



(b)



(c)

FIGURE 7: The accuracy curve for the two best methods over Hyperplane data streams. (a) Hyperplane with 20% noise. (b) Hyperplane with 25% noise. (c) Hyperplane with 30% noise.

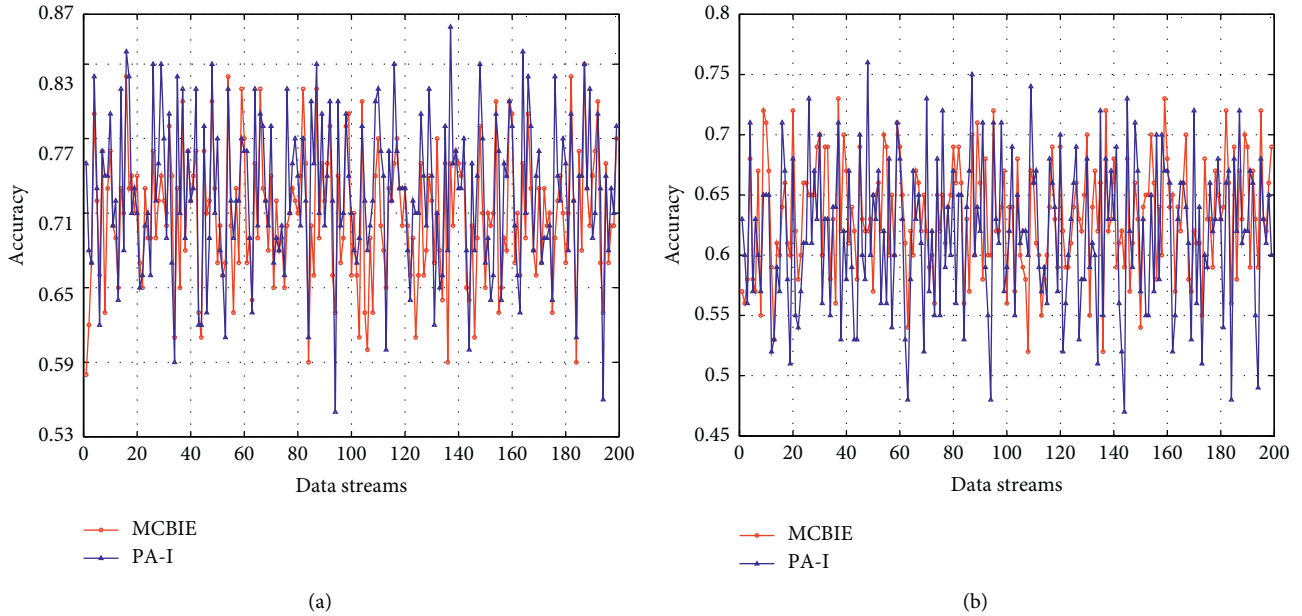


FIGURE 8: The accuracy curve for the two best methods over SEA data streams. (a) SEA with 20% noise. (b) SEA with 30% noise.

ratios. Moreover, in three cases, the maximum and minimum accuracy of MCBIE is higher than that of the PA-I. Through the analysis of the accuracy curve over SEA data stream, concerned with the ability to adapt to concept drift, these two methods seem to have the same function to deal with nonstationary data stream classification, as demonstrated in Figure 8. Moreover, with the growth of noise ratio, the MCBIE has a better performance than PA-I, such as stability.

From these analyses, we conclude that the MCBIE method is able to conduct nonstationary data stream classification with high accuracy in environments characterized by both concept drift and noise.

6. Conclusions

Classification task in nonstationary data streams faces the two main problems: concept drift and noise, which require the classification model to not only cope with concept drift but also differentiate noise from concept drift. In order to deal with these problems, a novel method named MCBIE was proposed, which can achieve the classification task in nonstationary data streams with noise. Aiming to enhance MCBIE's performance, the three strategies are used to alleviate the influence of concept drift and noise. In this paper, incremental learning can help microcluster as classifier to catch the concept change fast and ensemble strategy alleviates the disturbance between noise and concept drift. The function of smoothing parameter is to absorb the useful information from historical knowledge. Compared with the baseline methods, experiment results justify that our method, MCBIE, has the ability to perform classification in nonstationary streaming data setting. However, the three problems are worthy to be further concerned: (1) how to improve the noise recognition ability of our method in

abrupt drift environment needs to be further strengthened; (2) in addition to accuracy, the stability of model needs to be improved; (3) when concept reoccurs, it is important to design more appropriate strategies for the replacement of microcluster.

Data Availability

The data used to support the findings of this study have been deposited in the GitHub repository (<https://github.com/FanzhenLiu/ComplexityJournal>).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the Natural Science Foundation of Anhui Province (nos. 1608085MF147 and 1908085MF183), the Humanities and Social Science Foundation of the Ministry of Education (no. 18YJA630114), a Major Project of Natural Science Research in the Colleges and Universities of Anhui Province (no. KJ2019ZD15), MQNS (no. 9201701203), MQEPS (no. 96804590), MQRSG (no. 95109718), and the Investigative Analytics Collaborative Research Project between Macquarie University and Data61 CSIRO.

References

- [1] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary environments: a survey," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 12–25, 2015.

- [2] A. Jadhav, A. Jadhav, P. Jadhav, and P. Kulkarni, "A novel approach for the design of network intrusion detection system (NIDS)," in *Proceedings of 2013 International Conference on Sensor Network Security Technology and Privacy Communication System*, IEEE, New York, NY, USA, pp. 22–27, December 2013.
- [3] A. Salazar, G. Safont, A. Soriano, and L. Vergara, "Automatic credit card fraud detection based on non-linear signal processing," in *Proceedings of 2012 IEEE International Carnahan Conference on Security Technology*, IEEE, Newton, MA, USA, pp. 207–212, October 2012.
- [4] T. Bujlow, T. Riaz, and J. M. Pedersen, "A method for classification of network traffic based on c5. 0 machine learning algorithm," in *Proceedings of the 2012 International Conference on Computing, Networking and Communications*, IEEE, Maui, HI, USA, pp. 237–241, February 2012.
- [5] L. Gao, J. Wu, C. Zhou, and Y. Hu, "Collaborative dynamic sparse topic regression with user profile evolution for item recommendation," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, New York, NY, USA, February 2017.
- [6] S. Xue, J. Lu, and G. Zhang, "Cross-domain network representations," *Pattern Recognition*, vol. 94, pp. 135–148, 2019.
- [7] S. Ren, W. Zhu, B. Liao et al., "Selection-based resampling ensemble algorithm for nonstationary imbalanced stream data learning," *Knowledge-Based Systems*, vol. 163, pp. 705–722, 2019.
- [8] J. Sun, H. Fujita, P. Chen, and H. Li, "Dynamic financial distress prediction with concept drift based on time weighting combined with adaboost support vector machine ensemble," *Knowledge-Based Systems*, vol. 120, pp. 4–14, 2017.
- [9] T. Zhai, Y. Gao, H. Wang, and L. Cao, "Classification of high-dimensional evolving data streams via a resource-efficient online ensemble," *Data Mining and Knowledge Discovery*, vol. 31, no. 5, pp. 1242–1265, 2017.
- [10] W.-X. Lu, C. Zhou, and J. Wu, "Big social network influence maximization via recursively estimating influence spread," *Knowledge-Based Systems*, vol. 113, pp. 143–154, 2016.
- [11] Y. Zhang, J. Wu, C. Zhou, and Z. Cai, "Instance cloned extreme learning machine," *Pattern Recognition*, vol. 68, pp. 52–65, 2017.
- [12] J. Wu, Z. Cai, S. Zeng, and X. Zhu, "Artificial immune system for attribute weighted naive bayes classification," in *Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN)*, IEEE, Dallas, TX, USA, pp. 1–8, August 2013.
- [13] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, "Multi-instance learning with discriminative bag mapping," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1065–1080, 2018.
- [14] P. ZareMoodi, S. K. Siahroudi, and H. Beigy, "A support vector based approach for classification beyond the learned label space in data streams," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, ACM, Pisa, Italy, pp. 910–915, April 2016.
- [15] S. Ramirez-Gallego, B. Krawczyk, S. Garcia, M. Wozniak, J. M. Benitez, and F. Herrera, "Nearest neighbor classification for high-speed big data streams using spark," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 10, pp. 2727–2739, 2017.
- [16] J. Gama, R. Fernandes, and R. Rocha, "Decision trees for mining data streams," *Intelligent Data Analysis*, vol. 10, no. 1, pp. 23–45, 2006.
- [17] H. L. Hammer, A. Yazidi, and B. J. Oommen, "On the classification of dynamical data streams using novel "Anti-Bayesian" techniques," *Pattern Recognition*, vol. 76, pp. 108–124, 2018.
- [18] M. A. Maloof and R. S. Michalski, "Incremental learning with partial instance memory," *Artificial Intelligence*, vol. 154, no. 1–2, pp. 95–126, 2004.
- [19] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, "Online passive-aggressive algorithms," *Journal of Machine Learning Research*, vol. 7, pp. 551–585, 2006.
- [20] M. Tennant, F. Stahl, O. Rana, and J. B. Gomes, "Scalable real-time classification of data streams with concept drift," *Future Generation Computer Systems*, vol. 75, pp. 187–199, 2017.
- [21] J. Read, A. Bifet, B. Pfahringer, and G. Holmes, "Batch-incremental versus instance-incremental learning in dynamic and evolving data," in *Proceedings of International Symposium on Intelligent Data Analysis*, Springer, Helsinki, Finland, pp. 313–323, October 2012.
- [22] J. Lu, D. Sahoo, P. Zhao, and S. C. Hoi, "Sparse passive-aggressive learning for bounded online kernel methods," *ACM Transactions on Intelligent Systems and Technology*, vol. 9, no. 4, p. 45, 2018.
- [23] M. Oide, A. Takahashi, T. Abe, and T. Suganuma, "User-oriented video streaming service based on passive aggressive learning," *International Journal of Software Science and Computational Intelligence*, vol. 9, no. 1, pp. 35–54, 2017.
- [24] B. Krawczyk and M. Wozniak, "One-class classifiers with incremental learning and forgetting for data streams with concept drift," *Soft Computing*, vol. 19, no. 12, pp. 3387–3400, 2015.
- [25] S. Xu and J. Wang, "A fast incremental extreme learning machine algorithm for data streams classification," *Expert Systems with Applications*, vol. 65, pp. 332–344, 2016.
- [26] Y. Lu, Y.-M. Cheung, and Y. Y. Tang, "Dynamic weighted majority for incremental learning of imbalanced data streams with concept drift," in *Proceedings of the 2017 International Joint Conference on Artificial Intelligence*, pp. 2393–2399, Melbourne, Australia, August 2017.
- [27] Y. Zhang, J. Yu, W. Liu, and K. Ota, "Ensemble classification for skewed data streams based on neural network," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 26, p. 08, 2018.
- [28] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Wozniak, "Ensemble learning for data stream analysis: a survey," *Information Fusion*, vol. 37, pp. 132–156, 2017.
- [29] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, San Francisco, CA, USA, pp. 377–382, August 2001.
- [30] H. Wang, W. Fan, P. S. Yu, and 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, Washington, DC, USA, pp. 226–235, December 2003.
- [31] J. R. B. Junior and M. do Carmo Nicoletti, "An iterative boosting-based ensemble for streaming data classification," *Information Fusion*, vol. 45, pp. 66–78, 2019.
- [32] K. Jackowski, "New diversity measure for data stream classification ensembles," *Engineering Applications of Artificial Intelligence*, vol. 74, pp. 23–34, 2018.
- [33] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 964–994, 2016.
- [34] A. Tsymbal, "The problem of concept drift: definitions and related work," *Computer Science Department*, vol. 106, no. 2, 2004.

- [35] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch," *ACM SIGMOD Record*, vol. 25, no. 2, pp. 103–114, 1996.
- [36] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "Moa: Massive online analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010.
- [37] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, San Francisco, CA, USA, pp. 97–106, December 2001.
- [38] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 2, p. 4, Boston, MA, USA, April 2000.
- [39] A. Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà, "Improving adaptive bagging methods for evolving data streams," in *Proceedings of 2009 Asian Conference on Machine Learning*, Springer, Berlin Germany, pp. 23–37, November 2009.