Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2021, Article ID 5533777, 13 pages
https://doi.org/10.1155/2021/5533777

Research Article

Recurrent Adaptive Classifier Ensemble for Handling Recurring Concept Drifts

Tinofirei Museba,[1] **Fulufhelo Nelwamondo**,[2] **Khmaies Ouahada**,[2] **and Ayokunle Akinola**[2]

[1]*Department of Applied Information Systems, University of Johannesburg, Johannesburg, South Africa*
[2]*Department of Electrical and Electronic Engineering Science, University of Johannesburg, Johannesburg, South Africa*

Correspondence should be addressed to Tinofirei Museba; tmuseba@uj.ac.za

Received 27 January 2021; Revised 7 May 2021; Accepted 29 May 2021; Published 10 June 2021

Academic Editor: Ridha Ejbali

For most real-world data streams, the concept about which data is obtained may shift from time to time, a phenomenon known as concept drift. For most real-world applications such as nonstationary time-series data, concept drift often occurs in a cyclic fashion, and previously seen concepts will reappear, which supports a unique kind of concept drift known as recurring concepts. A cyclically drifting concept exhibits a tendency to return to previously visited states. Existing machine learning algorithms handle recurring concepts by retraining a learning model if concept is detected, leading to the loss of information if the concept was well learned by the learning model, and the concept will recur again in the next learning phase. A common remedy for most machine learning algorithms is to retain and reuse previously learned models, but the process is time-consuming and computationally prohibitive in nonstationary environments to appropriately select any optimal ensemble classifier capable of accurately adapting to recurring concepts. To learn streaming data, fast and accurate machine learning algorithms are needed for time-dependent applications. Most of the existing algorithms designed to handle concept drift do not take into account the presence of recurring concept drift. To accurately and efficiently handle recurring concepts with minimum computational overheads, we propose a novel and evolving ensemble method called Recurrent Adaptive Classifier Ensemble (RACE). The algorithm preserves an archive of previously learned models that are diverse and always trains both new and existing classifiers. The empirical experiments conducted on synthetic and real-world data stream benchmarks show that RACE significantly adapts to recurring concepts more accurately than some state-of-the-art ensemble classifiers based on classifier reuse.

1. Introduction

Advances in technology in recent years have witnessed an upsurge in the number of applications that generate large amounts of data streams at unprecedented volumes and speed. Examples of such real-world applications include network intrusion detection [1], sensor networks, spam filtering systems [2], and credit card fraud detection [3].

One of the biggest challenges faced by machine learning tasks in data stream learning is concept drift [4], where the data generating mechanism is constantly evolving and the statistical properties of the target concept change over time. Changes that happen in the underlying distribution of the data lead to a significant drop in predictive performance of the learning model. Wang et el. [3] described the term concept in machine learning as the quantity that a learning model is trying to predict. Concept drift often occurs in real-world applications, for example, in weather prediction where prediction models may change due to changes in seasons and consumer preferences may change over time due to seasons, fashion, and economy. Changes that occur in the underlying distribution of the data often lead to a drastic drop in classification performance of the learning model.

An efficient and effective online learning model must have the ability to recognize and respond to such changes accordingly and accurately. In streaming data, different types of concept drifts can be identified. Concept drifts can be categorized based on their speed into sudden and gradual drifts [4]. Sudden concept drift is characterized by severe changes between the underlying class distribution

and the incoming instances in a relatively short amount of time. Gradual concept drift takes a relatively large amount of time for significant changes to be revealed in differences of underlying class distributions between the old instances and the incoming instances. Regardless of the type of drift currently occurring, an online learning model must be able to track the drift, recognize its type and adapt to changes accordingly. In many real-world applications, it is common that patterns or concepts recur over time. Context recurrence is a common situation concerning concept drift. Domains associated with context recurrence include weather prediction where learning models change according to seasons. Other domains include financial prediction and dynamic control. Recurring contexts may occur due to cyclic phenomena such as seasons of the year or may be associated with irregular phenomena such as inflation rates or market condition. This phenomenon of recurring concepts is one of the key challenges that online learning algorithms [5] need to deal with. In the event that concept drifts recur, previously learned models may be applied to handle recurring concepts. Existing algorithms consider recurring concepts as new concepts, thereby increasing computational overheads as more classification models are generated. If patterns or concepts recur, previously learned classification models should be reapplied; thus, the predictive performance of the learning model can be optimized. The application of previously learned models may impact both negatively and positively on learning the current concept. Preserving all previously learned classification models induces overheads in both storage and computation, for example, when repeatedly assessing the performance of previously learned classification models on new training data. For this reason, the number of preserved models should be subject to some constraints, instead of increasing indefinitely. A selection scheme is required to decide which previously learned classification models should be preserved. As learning algorithms work at handling different kinds of drift, they tend to better represent the last observed concepts and discard previously learned concepts. Two research questions need to be answered when designing an ensemble classifier to handle recurring concepts; that is, which previously learned classification models should be preserved for future use? And how to exploit the preserved classification models to facilitate adaptation to recurring concepts?

To address the above research questions, this paper first reviews the latest progress on machine learning algorithms for handling recurring concepts and then proposes the Recurrent Adaptive Classifier Ensemble (RACE), specifically designed to handle recurring concept drifts in dynamic environments. RACE employs J48 Decision Tree, Multilayer Perceptrons (MLPs), and Support Vector Machines (SVMs) as base learners in order to maximize diversity and create dynamic decision boundaries separating the training instances, a change detection algorithm, and a diversity based strategy for preserving previously learned models to handle recurring concepts. When a new data chunk arrives, classification models of high diversity are adapted to the new training data.

The rest of this paper is organized as follows. Section 2 presents a review of related work. Section 3 introduces the Recurrent Adaptive Classifier Ensemble (RACE). Section 4 presents the empirical analysis of the comparison between RACE and other state-of-the-art algorithms designed to handle recurring concepts using selected datasets considering the accuracies achieved and how the algorithms handle recurring concepts.

2. Related Work

Scenarios associated with recurring concepts are not uncommon, and a number of contemporary approaches have been proposed to address recurring concepts with minimum overheads. Many machine learning techniques have emerged in the literature as candidate solutions, and ensemble classifiers have demonstrated the ability to handle different types of drifting concepts in nonstationary environments. Hassan [6] proposed a concept drift adaptation technique in distributed environment for real-world data streams. The algorithm uses drift detection method; if concept drift is detected, it retrains the model, and knowledge of previously learned concepts is lost. The approach does not automatically identify the type of drift. Sarnovsky [7] proposed the heterogeneous adaptive ensemble model for data stream classification which utilizes the dynamic class weighting scheme and a mechanism to maintain the diversity of the ensemble members. The algorithm implicitly handles recurring concepts, and classifiers with lower weights are discarded, making it difficult to handle recurring concepts. Liu [8] proposed an instance based ensemble learning algorithm called the diverse instance weighting ensemble (DiwE). The algorithm weights classifiers according to their performance, and poorly performing classifiers are discarded. Heusinger [9] proposed a combination of the modified versions of Robust Soft Learning Vector Quantization (RSLVQ) and Generalized Learning Vector Quantization (GLVQ) to learn streaming data and adapt to all types of concept drift. The integration of Adadelta and Adamax into RSLVQ and GLVQ optimized the prediction performance over their vanilla versions. The combined algorithm does not detect drifts and does not handle concept drift explicitly. Zheng [10] proposed a semisupervised classification algorithm on data streams with recurring concept drift and concept evolution in data streams with partially labeled data. The framework uses the Jensen–Shannon divergence based change detection technique on classifier confidence score instead of classification error rate to detect recurring concept drift. The algorithm uses too many parameters that are difficult to tune. Namitha [11] proposed a novel algorithm to identify recurring concepts in data stream clustering. If concept drift is detected, the algorithm retrieves the most matching model from the repository. The algorithm has no strategy to prevent the repository from growing or increasing indefinitely. Wing [12] proposed a bagging ensemble that adapts to concept drift by using a dynamic cost-sensitive weighting scheme for component classifiers according to their classification performances and stochastic sensitivities. The

algorithm discards classifiers whose weight is below a predefined threshold, making it unable to adapt to recurring concepts. Zang [13] presented the drift detection based incremental ensemble (DIE) that combines the operations of concept drift detection and component update mechanism to react to different types of concept drift. DIE assigns weights to classifiers and discards classifiers whose weight is below a predefined threshold, making it difficult to react to recurring concepts. Baidari [14] proposed the Accuracy Weighted Diversity based Online Boosting (AWDOB) which is based on an Adaptable Diversity based Online Boosting (ADOB). AWDOB uses an accuracy weighting scheme that exploits the accuracy of the current expert and the number of correctly classified and incorrectly classified instances of all experts to assign the current expert weight to the current instance in the data stream. Experts with lower weights are discarded from the ensemble. The process of calculating and assigning weights takes time and slows the learning process. Gu [15] presented the a novel self-organizing fuzzy inference ensemble framework (SOFEnsemble) which is capable of self-learning, processing streaming data on a chunk by chunk basis, and continuously self-updating the decision boundaries by identifying the more representative samples. SOFEnsemble has a high computational efficiency, and the use of fuzzy inference slows down the learning process. Zeng [16] proposed a chunk based incremental ensemble algorithm called Dynamic Updated Ensemble (DUC) for learning imbalanced data streams with concept drift. DUE periodically updates previous components to make the ensemble react to different kinds of concept drift, and the final decision of testing events is based on the weighting voting value of a certain number of best performing classifiers. DUE discards classifiers whose weight is below a predefined threshold making it unable to accurately react to recurring concepts. Liu et al. [17] proposed a comprehensive online active learning framework (CALMID) that includes an ensemble classifier, a drift detector, a label sliding window, sample sliding windows, and an initialization training sample sequence to learn concept drift. The algorithm has a sample weight formula that assigns weights to classifiers. CALMID was found to be effective and efficient when compared to other state-of-the-art algorithms.

Most of the proposed ensemble approaches in the literature handle recurring concepts by relearning them as if the concepts are new and not recurring. Existing ensemble classifiers for recurring concepts share a common weakness; that is, when a new data chunk arrives, all the ensembles utilize all previously learned concepts without adapting them to new training data. Neither of the proposed approaches explores the exploitation of highly diverse models previously learned to handle recurring concepts by firstly adapting them to the new training data. Therefore, in this paper, a novel and evolving ensemble learning approach called Recurrent Adaptive Classifier Ensemble (RACE) is presented. RACE stores highly diverse models and does not directly combine the prediction outputs of the models. Instead, each diverse model in the archive is first adapted to the new training data, and the model which further increases the diversity of the ensemble is removed from the archive.

In the next section, we present our proposed approach, the Recurrent Adaptive Classifier Ensemble (RACE), that explicitly exploits diversity to handle recurring concepts.

3. Recurrent Adaptive Classifier Ensemble (RACE)

The Recurrent Adaptive Classifier Ensemble (RACE) employs Support Vector Machines (SVMs) as the base learner. The algorithm first builds a support vector, denoted as f_1, with first streaming data chunk and stores the first support vector in an archive. When a new data chunk arrives, the drift detection algorithm checks if the data chunk is from the same distribution from the first created support vector. If the data chunk is from a different underlying distribution, the preserved support vector is adapted to the new data chunk and a new support vector is built from scratch from the new data chunk. The adapted support vector and new support vector are combined to constitute an ensemble to perform classification at time t. RACE does not directly combine the prediction outputs of the stored models in the library. Each preserved previously learned model is first adapted to fit the current data, and then the adapted models and the newly constructed model from the most recent data chunk are combined. Previously learned models are preserved according to a diversity based criterion as opposed to an accuracy based criterion, as the base classifiers have to perform diversely for the ensemble of classifiers to improve its prediction performance. RACE uses Yule's Q Statistic [18] as a diversity measure to minimize the ensemble error. The diversity measure is recommended due to its simplicity and ease of interpretation [19]. RACE stores highly diverse previously learned models. The previously learned diverse models are then adapted to the current concept via knowledge transfer. A diversity measure is used to measure model diversity to keep only previously learned diverse models [20]. The transfer learning is appropriate as it optimizes the learning process in terms of accuracy and learning efficiency. To learn new concepts, previously learned diverse models are employed as initial candidates of the ensemble for learning new concepts. RACE adapts each previously learned model in the archive to the new training data. The adapted models and the model learned from new training data are combined to predict incoming instances. The newly built model is stored in the archive if it is not full. The model whose removal will lead to the largest diversity among the remaining models is removed from the archive. Algorithm 1 provides a description of the ensemble framework.

Algorithm 2 provides a description of the RACE algorithm. The detailed steps of the Recurrent Adaptive Classifier Ensemble (RACE) algorithm are presented in Algorithm 2 with the assumption that data arrives sequentially.

The Recurrent Adaptive Classifier Ensemble (RACE) uses the Early Drift Detection Method [21] to detect drift. If concept drift is detected, the preserved models are adapted to fit the current data. EDDM is an online learning system since it does

not store the training instances for posterior use. The detailed steps of the Recurrent Adaptive Classifier Ensemble (RACE) algorithm are presented in Algorithm 2, with the assumption that t data chunks $D_1, \ldots, D_t$ arrive sequentially.

3.1. Model Preservation. Preserving previously learned models induces overheads in terms of both storage and computation. For example, iteratively assessing the predictive performance of previously learned models on new data is computationally prohibitive. To prevent the ensemble from growing indefinitely, the size of the ensemble is dynamic. Previously learned models are preserved in an archive of size n. When a data chunk arrives at step t, the preserved models in the archive are adapted to fit the current data. The drift detection helps to detect if the new data chunk is drawn from a different data distribution. The newly generated model from the current data chunk, f_t, will be directly stored in the archive if the size of the archive is less than n. To optimize diversity, the model whose removal will increase diversity among the remaining models in the archive will be discarded from the archive. RACE combines the prediction outputs of previously learned diverse models that are representative of the current concept with the prediction output of a new model built with the first data chunk to form final decisions on testing training instances of the current concept.

3.2. Archive Size and Transfer Operation. The goal is to minimize computational overheads by creating a dynamic pool size of previously learned models from where the ensemble to learn recurring concepts and sudden and gradual concepts is generated. RACE performs a transfer of every previously learned model with the new streaming data chunks. To improve the time efficiency of RACE, we implement the transfer operation in a parallel processing manner. By parallelizing the transfer operations, the speedup ratio is optimized and the runtime level is satisfactory for nonstationary environments. In line with transfer operation of knowledge is the archive size that is dynamic to cater for other different types of concepts. Parallelization of transfer operation is best optimized with a reasonable dynamic archive size which does not grow indefinitely, since models that cause diversity among models to decrease are removed from the archive. The implementation of a drift detection mechanism facilitates detection of recurring concepts. To reduce overheads, a dynamic pool size from which models are drawn serves as a better starting point. The goal is to capitalize on the accuracy as the ensemble size fluctuates. To validate the behavior of the RACE algorithm, we conduct two experiments. The first experiment evaluates the validity of RACE using knowledge transfer. In the second experiment, the behavior of RACE is evaluated using Hidden Markov Models (HMM).

4. Experimental Configuration

The empirical experiments to assess the performance of RACE were conducted on the Massive Online Analysis (MOA) framework, a software environment for implementing machine learning algorithms and running experiments for online learning. MOA is an open source framework for data streaming mining in evolving environments. The generalization performance of RACE is compared to other state-of-the-art algorithms designed to handle recurring concepts such as the comprehensive online active learning framework (CALMID) [17], Dynamic Updated Ensemble (DUE) [16], Self-Organizing Fuzzy Ensemble Inference System (SOFEnsemble) [15], and Accuracy Weighted Diversity based Online Boosting (AWDOB) [14].

4.1. Datasets Used in the Experiments. We evaluate the performances of the algorithms with data created by five synthetic dataset generators. All data stream generators are available in MOA. The synthetic datasets contain three types of concept drift, namely, gradual, sudden, and recurring concept drift.

The Hyperplane dataset [22] is represented by the set of points x that satisfy $\sum_{i=1}^{d} w_i x_i = w_0$, where x_i is the ith coordinate of x. Two classes are distinguished in the following way: instances for which $\sum_{I=1}^{d} w_i x_i > w_0$ are labeled positive, and instances for which $\sum_{i=1}^{d} w_i x_i < w_0$ are labeled negative. Drifts are simulated by changing each weight attribute $w_i = w_i + d\alpha$, where is the probability that the direction of change is reversed and d is the change applied to every instance. This generator was adopted to create a dataset that contains 1,000,000 instances.

The LED dataset [23] is used to predict the digit displayed on a seven-segment LED display. The particular configuration of the generator used for the experiment produces 24 binary attributes, 17 of which are irrelevant. Concept drift is simulated by interchanging relevant attributes. A stream of 1,000,000 instances was generated.

The Random Tree dataset [24] is generated by the Random Tree generator. The dataset contains 1,000,000 instances and 10 attributes. The dataset has four recurring concepts which are evenly distributed among the instances.

The SEA dataset [25] consists of three attributes, where only two are recognized as relevant attributes. All three attributes have values between 0 and 10. The points of the dataset are divided into four blocks with different concepts. In each block, the classification is done using $f_1 + f_2 \leq \theta$, where f_1 and f_2 represent the first two attributes and θ is a threshold value. The dataset contains 1,000,000 instances.

The last artificial dataset adopted for this study is the STAGGER Boolean Concepts. The dataset presents enough variety of drifts to perform principled studies. It allows a proper analysis considering several types of drift with different amounts of severity and speed. STAGGER Boolean Concepts dataset generates the data with categorical features using a set of rules to determine the class label. The dataset contains three nominal attributes, namely, size = {small, medium, large}, color = {red, green}, and shape = {circular, noncircular}. Concept drift is simulated by changing the items in the rules. Before the first drift, instances are labeled positive if (color = red) and (size = small). Before the occurrence of the second drift, instances are classified as

positive if (color = green) and (shape = circular), and after the second drift, instances are classified as positive only if (size = medium) and (size = large).

Table 1 provides a description of the real-world datasets used in the experiments. The datasets include Airlines [26], KDD99 Cup [27], Covertype [28], Poker Hand [29], and Sensor Data [30].

4.2. Evaluation of RACE. This section investigates the proposed algorithm and compares its predictive accuracy and drift handling capabilities with existing ensemble based approaches: CALMID, DUE, SOFEnsemble, and AWDOB. We also investigate in the second experiment the effect of Hidden Markov Model on the predictive performance and its recurrent drift handling capabilities.

The predictive performance and the recurrent drift handling capabilities of RACE were tested on both artificial and real-world datasets, and corresponding ranks of all algorithms are determined in such a way that higher averages represent lower ranks. Significance tests and post hoc comparisons on ranks are performed to determine significance levels and critical differences. The prediction accuracies and average ranks of RACE, CALMID, DUE, SOFEnsemble, and AWDOB are shown in Table 2.

It is evident from the table that shows accuracy measures that RACE performed significantly better than CALMID, DUE, SOFEnsemble, and AWDOB. The Nemenyi test [31] was applied for pairwise comparison. The critical difference is 1.432. From the figure that provides the average ranks of algorithms compared, it is evident that RACE performed significantly better than the other four algorithms. Figure 1 shows the critical difference plots from post hoc Nemenyi tests of average rankings for experiments on all datasets.

To further evaluate the drift handling capabilities of RACE against the other four representative and current algorithms designed to handle concept drift, we introduce the two Kappa evaluation measures, Kappa Temporal and Kappa M, on all the five algorithms designed to handle recurring concepts. The Kappa evaluation measure is widely used in data stream learning and can handle both multiclass and imbalanced class problems. The larger the Kappa value, the more generalized the classifier, and a negative Kappa value is an indication of low predictive accuracy. Kappa Temporal values are shown in Table 3.

Table 4 shows the Kappa M values of all the datasets used.

Kappa values for both Temporal and M are positive as the attributes in the datasets are averagely balanced.

The statistical tests applied on Kappa Temporal on artificial and real-world data streams showed significance differences at any specified level of significance. Statistical tests for Kappa M on both artificial and real-world datasets also showed significance differences at a specified level of significance, and for this experiment, we chose 0.05. The Nemenyi test [31] was applied for Kappa Temporal and Kappa M for pairwise comparison. The critical difference (CD) is 1.421. RACE performed significantly better than CALMID, DUE, SOFEnsemble, and AWDOB.

4.3. Resources Comparison. To analyze the benefits in terms of resources usage, we compare CPU time and memory consumption of RACE, CALMID, DUE, SOFEnsemble, and AWDOB using real-world data streams since they have large numbers of attributes. The ensemble sizes of all the algorithms are dynamic; that is, they vary in size given the task at hand. Lower values generated in the two scenarios are considered to be the best for each algorithm. Corresponding ranks are determined such that higher averages are representing lower ranks.

Table 5 shows the memory consumption (MB) of each algorithm on each dataset.

According to Table 5, in most cases, RACE achieved minimal memory consumption while AWDOB consumed the most memory. The insertion and deletion of models make memory usage lower for RACE when compared to other algorithms.

Table 6 shows the CPU processing time(s) for each algorithm on each real-world dataset.

As shown in Table 4, through the comparative analysis, we found that RACE consumed the least processing time, followed by CALMID, and SOFEnsemble has the longest CPU processing time.

4.4. Accuracy over Time. Graphical plots are generated for each dataset to describe the performance curves of all the tested algorithms at each time step. The x-axis represents the number of processed observations, and the average accuracy is presented on the y-axis. The graphical plots allow adaptation abilities of all comparative algorithms under different streaming conditions to be analyzed. As shown in the accuracy over time plots, RACE achieved the highest predictive accuracies on the Hyperplane 81.67%, Stagger 79.34%, Covertype 81.56%, and Sensor Data 80.34%. In total, the RACE average ranking in both artificial and real-world data streams is 1.4, CALMID is 3.2, DUE is 3.9, SOFEnsemble is 2.5, and AWDOB is 4.0.

Figure 2 shows the accuracy over time plots of the five algorithms on the Hyperplane dataset that exhibits gradual concept drift. The accuracy of all the algorithms shows the same trend. RACE performs the best, followed by DUE, and CALMID performs the worst. RACE is designed to adapt to all types of concept drift.

Figure 3 demonstrates the accuracy over time plots of the five algorithms on the Stagger dataset which exhibits sudden concept drift. As can be observed, RACE performs the best, followed by DUE, and CALMID is the third, while SOFEnsemble and AWDOB are the worst.

Figure 4 shows the accuracy over time plots of the five algorithms on the LED dataset which is devised to evaluate the ability to handle sudden concept drift. RACE performs the best, followed by AWDOB and then CALMID. SOFEnsemble and DUE perform poorly.

Figure 5 shows the prediction accuracy of the five algorithms on the SEA dataset which is devised to evaluate the ability to handle sudden and gradual drifts. The trend of all the five algorithms is basically the same. Among them,

Input: $(\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_t)$ chunks of streaming data
M: a set of diverse models previously learned
Output : $\mathbf{E}_t$ **:** the generalized ensemble model at time step t
(1) For each data chunk D_t do
(2) Learn a new base model f_t with D_t
(3) Select transferred models f_i^t by transferring the highly diverse stored models $f_i \in M$
(4) Build the generalized ensemble E_t using the transferred models f_i^t and the newly learned model f_t
(5) Update M with f_t to maximise diversity
(6) Endfor

Algorithm 1: Framework of RACE.

Input: $(D_1, D_2, \ldots.., D_t)$ the streaming data chunks
E_t archive of ensemble models at time step t
Diversity measure : Q Statistic
Drift Detection Method *Detect Drift*
Output: F_t**:** the generalized ensemble model at each time step t
(1) For each incoming data chunk D_t do
(2) Train new model f_t with data chunk D_t
(3) Test E_t with f_t
(4) drift $\leftarrow$ Detect Drift ( )
(5) if drift == true
(6) adapt models to current data
(7) else
(8) Update E_t with f_t to maximize diversity
(9) End if
(10) If $|E_t - 1| < t$ then
(11) $E_t \leftarrow E_{t-1} \bigcup \{f_t\}$
(12) *else*
(13) $S_t \leftarrow S_{t-1}$
(14) Endif
(15) Calculate diversity of models
(16) Output F_t
(17) Endif

Algorithm 2: RACE.

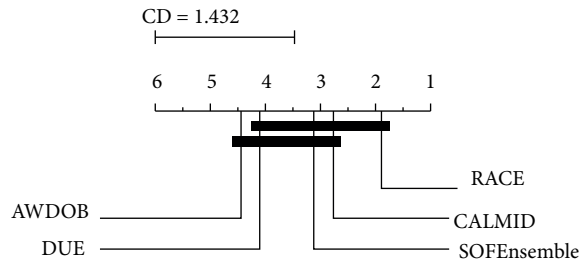

Figure 1: Average rank diagram of the five algorithms.

RACE performs the best, followed by DUE and AWDOB, and SOFEnsemble performs the worst.

Figure 6 shows the accuracy over time plots of the five algorithms on the Random Tree dataset which is devised to evaluate the ability to handle recurring concepts. AWDOB performs well in the first observed instances, but as the number of observed instances increases, RACE outperforms all the four algorithms.

Table 1: Description of real-world datasets.

Dataset	Classes	Attributes	Attribute types	Instances
Airlines	2	7	Numeric	539,383
KDD99 Cup	23	42	Numeric	494,021
Covertype	10	54	Binary	581,012
Poker Hand	10	10	Numeric	629,012
Sensor Dataset	54	5	Numeric	2,219,803

Artificial data streams are typically designed for controlled environments. When handling real-world classification problems, several challenges emerge. The major issue is that of the identification and location of the concept drifts. Accordingly, RACE was evaluated on real-world data streams, namely, Airlines, Forest Covertype, KDD99 World Cup, Poker Hand, and Sensor Data. With the five real datasets and the five observations, significance tests were performed and the obtained results showed improvements. Figures 7–11 show the accuracy over time plots of the five algorithms on five real-world datasets.

Table 2: Prediction accuracies and average ranks of the five algorithms.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Hyperplane	81.67 (1)	69.85 (3)	60.74 (4)	76.61 (2)	61.42 (5)
Stagger	79.34 (1)	74.39 (2)	64.62 (4)	73.56 (3)	63.52 (5)
LED	83.46 (1)	70.35 (3)	69.39 (4)	79.62 (2)	66.47 (5)
SEA	76.34 (2)	73.49 (3)	64.43 (5)	80.28 (1)	66.78 (4)
Random Tree	84.4 (1)	68.76 (5)	71.53 (3)	81.62 (2)	71.38 (4)
Airlines	78.59 (2)	71.46 (3)	61.34 (5)	86.73 (1)	63.32 (4)
KDD99	69.28 (3)	67.54 (4)	81.36 (2)	63.48 (5)	83.12 (1)
Covertype	81.56 (1)	73.37 (3)	66.87 (4)	81.36 (2)	64.39 (5)
Poker Hand	84.31 (1)	70.48 (3)	69.38 (4)	66.49 (5)	82.34 (2)
Sensor Data	80.34 (1)	72.57 (3)	70.43 (4)	79.67 (2)	69.36 (5)
Average ranks	**1.4**	**3.2**	**3.9**	**2.5**	**4.0**

Table 3: Kappa temporal values for all the ten datasets.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Hyperplane	76.43 (1)	63.84 (3)	66.44 (2)	62.61 (4)	60.48 (5)
Stagger	76.34 (1)	71.42 (3)	64.62 (4)	74.56 (2)	62.58 (5)
LED	76.49 (2)	71.45 (3)	68.38 (4)	82.67 (1)	65.67 (5)
SEA	72.24 (3)	76.42 (1)	64.43 (5)	74.27 (2)	69.48 (4)
Random Tree	81.42 (1)	66.36 (5)	73.24 (2)	68.32 (4)	72.43 (3)
Airlines	86.72 (1)	71.46 (3)	67.45 (4)	76.43 (2)	64.39 (5)
KDD99	78.48 (2)	69.34 (4)	86.16 (1)	64.58 (5)	70.32 (3)
Covertype	83.36 (1)	71.47 (3)	68.27 (5)	78.46 (2)	68.45 (4)
Poker Hand	82.37 (1)	76.47 (2)	71.58 (4)	64.49 (5)	74.36 (3)
Sensor Data	76.54 (2)	67.57 (5)	70.43 (3)	79.32 (1)	69.45 (4)
Average ranks	**1.5**	**3.2**	**3.4**	**2.8**	**4.1**

Table 4: Kappa M values for all the ten datasets.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Hyperplane	78.37 (1)	69.54 (2)	62.56 (4)	66.48 (3)	61.42 (5)
Stagger	76.48 (3)	74.39 (2)	67.42 (4)	79.56 (1)	65.42 (5)
LED	81.36 (1)	69.35 (4)	72.49 (3)	80.48 (2)	67.43 (5)
SEA	75.34 (3)	77.49 (2)	66.43 (5)	83.28 (1)	69.78 (4)
Random Tree	82.37 (1)	74.36 (3)	67.58 (5)	69.42 (4)	78.48 (2)
Airlines	80.59 (2)	72.46 (3)	62.34 (5)	83.73 (1)	64.32 (4)
KDD99	66.28 (4)	64.54 (3)	79.36 (2)	62.48 (5)	84.42 (1)
Covertype	77.36 (1)	73.37 (3)	67.37 (4)	74.46 (2)	65.29 (5)
Poker Hand	82.34 (1)	75.38 (3)	67.38 (4)	64.43 (5)	78.44 (2)
Sensor Data	78.64 (2)	68.54 (5)	73.48 (3)	80.47 (1)	71.46 (4)
Average ranks	**1.9**	**3.0**	**3.9**	**2.5**	**3.7**

RACE achieved the highest predictive accuracies: Covertype 81.56%; Poker Hand, 84.31%; Sensor Data, 80.34%. The overall average ranking of RACE is 1.4, CALMID 3.2, SOFEnsemble 3.9, DUE 2.5, and AWDOB 4.0.

Figure 7 shows the accuracy over time plots of the five algorithms on the Airlines dataset. DUE performs well in the first observed instances, but as more instances are observed, RACE performs the best. SOFEnsemble performs the worst.

Figure 8 shows the accuracy over time plot of the five algorithms on the KDD99 dataset. RACE performs the best, followed by DUE. SOFEnsemble performs the worst, and the trend is the same for CALMID and AWDOB.

Figure 9 demonstrates the accuracy over time plots of the five algorithms on the Covertype dataset. RACE performs the best, followed by DUE. AWDOB performs the worst.

Figure 10 demonstrates the accuracy of the five algorithms on the Poker Hand dataset. The prediction performance of all the algorithms fluctuates with time. As more instances are observed, RACE performs the best, followed by AWDOB. DUE and SOFEnsemble perform the worst.

Figure 11 shows the accuracy over time plots of the five algorithms on the Sensor Data to evaluate gradual concept drift. RACE performs the best, followed by DUE. SOFEnsemble is the third, and AWDOB and CALMID perform the worst. RACE manages recurrent change detection mechanism by reusing previously learned concepts and generalizes well in different situations especially in different concept drift environments. However, other existing ensemble methods do not store previously learned knowledge and lack detection mechanisms, and for that they adapt poorly to different types of drifts.

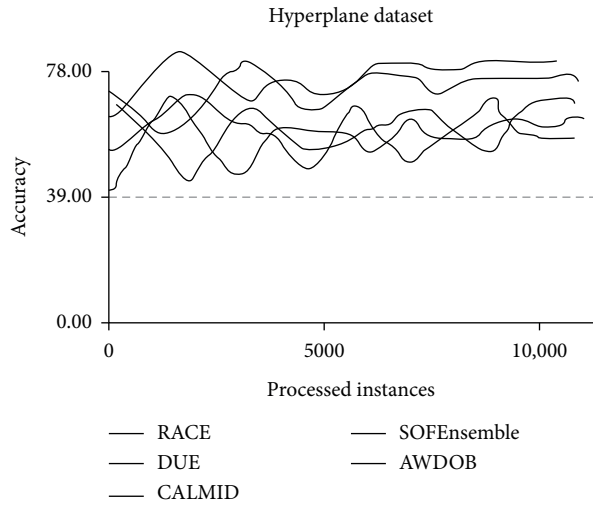

Figure 2: Prediction accuracy of the five algorithms on the Hyperplane dataset.

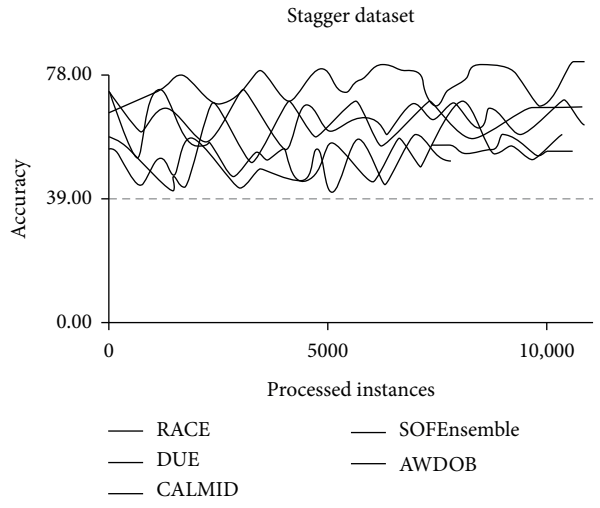

Figure 3: Prediction accuracy of the five algorithms on the Stagger dataset.

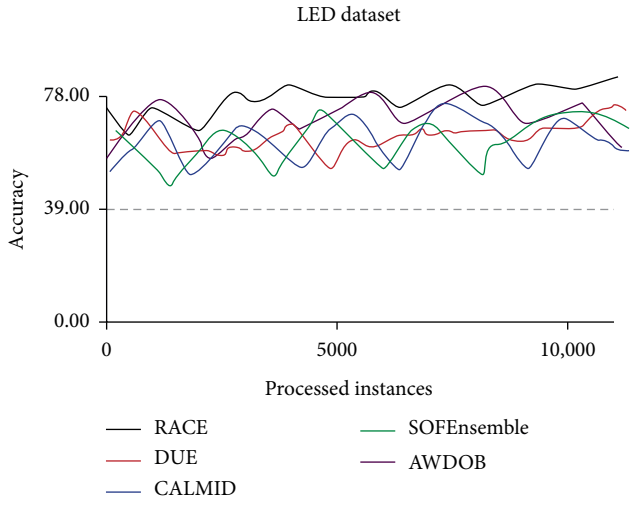

Figure 4: Prediction accuracy of the five algorithms on the LED dataset.

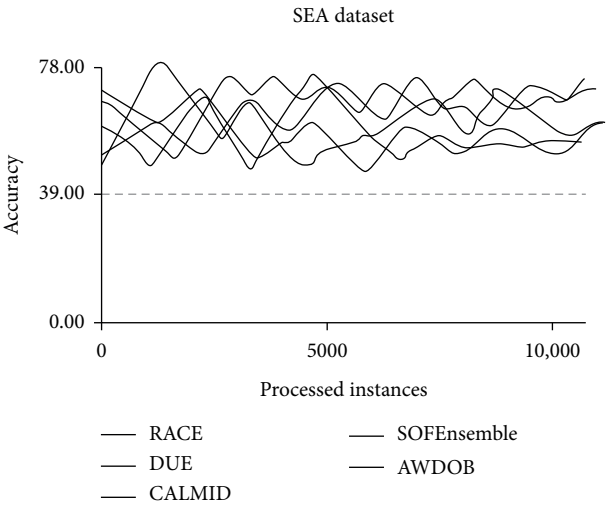

Figure 5: Prediction accuracy of the five algorithms on the SEA dataset.

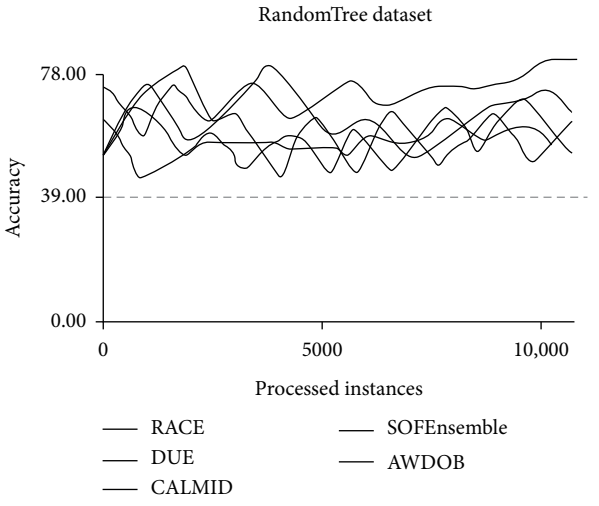

Figure 6: Prediction accuracy of the five algorithms on the Random Tree dataset.

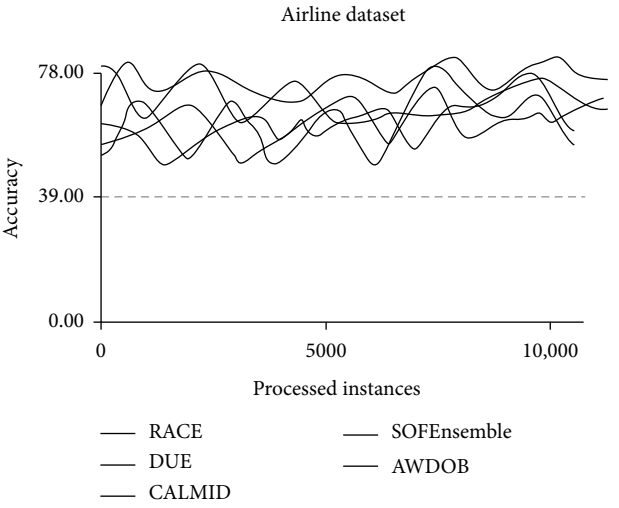

Figure 7: Prediction accuracy of the five algorithms on the Airlines dataset.

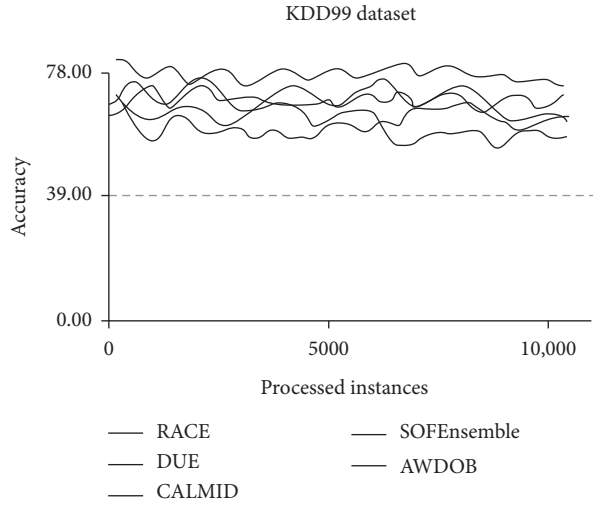

Figure 8: Prediction accuracy of the five algorithms on the KDD99 dataset.

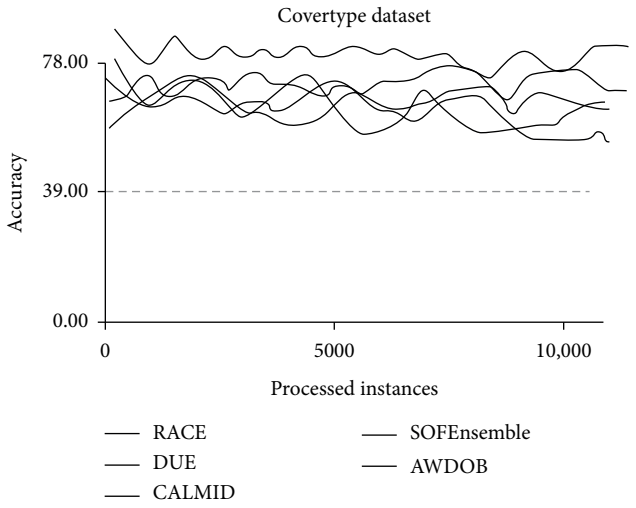

Figure 9: Prediction accuracy of the five algorithms on the Covertype dataset.

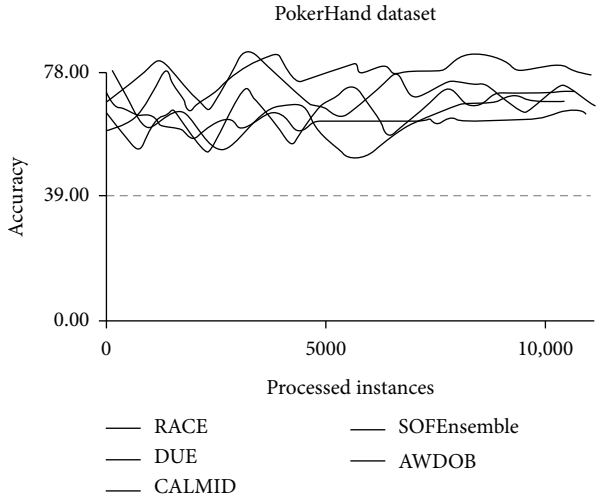

Figure 10: Prediction accuracy of the five algorithms on the Poker Hand dataset.

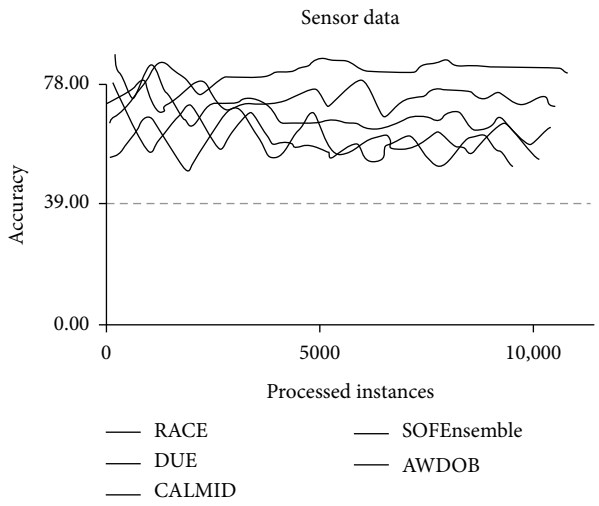

Figure 11: Prediction accuracy of the five algorithms on the Sensor Data.

For all the five real-world datasets, RACE subjects all classifiers to a diversity and accuracy evaluation after each iteration. If they are not representative of the current concept, they are discarded, and classifiers that are representative of the current concept and those with higher amounts of diversity are retained, which allows RACE to appropriately deal with recurring concepts. Poker Hand (84.31%) and DUE (81.36) on the KDD99 dataset are able to deal with concept drifts appropriately and this can only be attributed to the periodic inclusion of new base learners, while CALMID and SOFEnsemble do not maintain dynamic pools due to their static ensemble size.

5. Hidden Markov Model-Based RACE

In our next experiment, we investigate the behavior of RACE when we replace the knowledge transfer process with Hidden Markov Model, a metalearner. Hidden Markov Models (HMM) are known to work extremely well in practice as prediction, recognition, and identification systems in a very efficient manner. Hidden Markov Models are based on the assumption that consecutive observations are independent and therefore the probability of a sequence of observations can be expressed as the probabilities of individual observations.

The Hidden Markov Model is a metalearner that is able to predict when recurring concepts will occur. We can then anticipate that recurrent drifts choose also the most appropriate model for the incoming data chunk. The implementation of RACE using Hidden Markov Models allows the algorithm to better handle recurrent situations in classification problems in dynamic environments, thus enabling the evolving base learner to adapt to recurring concepts in a timely manner. This is made possible by predicting when the drift will happen from training examples at a given time and also getting a similarity level between concepts from a fuzzy similarity function.

Table 5: Memory consumption of each algorithm on each dataset.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Airlines	36.25 (1)	42.39 (2)	44.29 (3)	57.31 (4)	61.23 (5)
Covertype	43.17 (2)	28.34 (1)	63.46 (5)	59.63 (4)	51.42 (3)
Poker Hand	66.37 (1)	71.26 (3)	69.47 (2)	79.45 (5)	77.36 (4)
KDD99	83.43 (4)	76.39 (2)	78.36 (3)	73.43 (1)	88.34 (5)
Sensor Data	59.34 (1)	87.63 (5)	67.34 (3)	79.23 (4)	62.41 (2)
Average rank	**1.8**	**2.6**	**3.2**	**3.6**	**3.8**

Table 6: CPU processing time for each algorithm on real-world datasets.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Airlines	13.49 (1)	15.43 (2)	29.43 (3)	54.03 (5)	35.38 (4)
Covertype	17.36 (2)	36.48 (3)	33.42 (2)	53.48 (4)	65.43 (5)
Poker Hand	43.56 (1)	78.45 (4)	83.37 (5)	73.28 (3)	56.32 (2)
KDD99	81.43 (3)	78.32 (2)	76.43 (1)	88.46 (5)	83.52 (4)
Sensor Data	85.46 (2)	81.34 (1)	91.23 (3)	99.38 (5)	97.41 (4)
Average rank	**1.8**	**2.4**	**2.8**	**4.4**	**3.8**

5.1. Description of the Algorithm. Multilayer Perceptrons (MLPs), J48 Decision Trees, and Support Vector Machines are used as the base learners, processing the training instances from the time series data by means of an incremental learning algorithm to generate a classifier from the data chunk that represents the underlying concept. A pool that stores all concept representation is created. The drift detection mechanism (DDM) is continuously monitoring the error rate generated by learning algorithm; a warning is generated by the DDM if the error rate exceeds a predefined threshold, and a new classifier is learned. A metamodel is trained from the information provided by the drift detection mechanism and the metamodel evolves as new concepts are detected. The fuzzy concept similarity approach determines whether the underlying concept is recurrent, and previously learned models are applied.

In this case, previously learned highly diverse models are no longer trained as they are stable models that adequately represent specific concepts.

5.2. Experimental Analysis. To compare the performance of RACE that uses knowledge transfer and the RACE that uses Hidden Markov Models, we use the same synthetic datasets and real-world datasets used to compare the predictive performance of RACE with recent state-of-the-art algorithms designed to handle recurring concepts in dynamic environments.

Using the MOA framework, the performance of the analyzed algorithms is evaluated with respect to accuracy, time efficiency, and memory usage on both synthetic datasets and real-world datasets. Table 7 shows the prediction accuracy of RACE using Markov Models.

The performance of RACE is also evaluated with respect to CPU processing time in seconds. Table 8 shows the CPU processing time in seconds.

Concerning runtime, online ensembles like AWDOB require the most time for classification, followed by ARF and DP. RACE is the least time-consuming. This is partly because the combination of Hidden Markov Models with a drift detection mechanism offers quicker reactions to sudden and recurring concept drift compared to other methods. For this reason, RACE is in a better position to capture changes with Hidden Markov Models much more efficiently and adapt to different types of drifts accurately and timeously.

Memory consumption on the real-world datasets that have many attributes is shown in Table 9.

The memory consumption of SOFEnsemble, CALMID, and AWDOB is more than that of RACE and DUE. The three algorithms maintain a large pool of historical concepts which are checked for reuse. RACE and DUE require the least memory storage due to their pruning strategy.

5.3. Comparison of Accuracy Performance. To compare the accuracy of the five algorithms over multiple datasets, we follow the methodology proposed by Demsar [32]. We firstly use the nonparametric Friedman test to determine if there is a statistically significant difference between the rankings of the compared algorithms. We then perform the Nemenyi post hoc test with average rank diagrams. The rankings are depicted on the axis such that the best ranking algorithms are at the rightmost part of the diagram. The algorithms that do not differ significantly are connected with a line. The critical difference (CD) is indicated above the graph.

As can be observed, from the critical difference (CD) plots, RACE outperforms the other algorithms most of the time.

Figure 12 shows the critical difference plots from post hoc tests of rankings for experiments on the datasets used.

The nonparametric Friedman test was carried out to extend the analysis of comparing multiple classifiers over multiple datasets. The null hypothesis for the test was that there is no difference between the performances of all the tested algorithms. In the event of rejecting the null hypothesis, the Nemenyi test could have been employed to verify whether the performance of our algorithm, RACE, is statistically different from the rest of the algorithms used for comparative purposes. The critical difference (CD) from the average rank diagram shows that our algorithm is significantly better than the four recent representative algorithms on nonstationary time series data.

Table 7: Prediction accuracy of the algorithms using Hidden Markov Models.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Hyperplane	76.47 (2)	72.53 (3)	63.24 (5)	79.63 (1)	66.38 (4)
Stagger	72.32 (1)	66.58 (4)	70.33 (2)	68.28 (3)	62.48 (5)
LED	69.56 (3)	72.37 (2)	66.49 (4)	62.37 (5)	73.54 (1)
SEA	73.48 (1)	64.47 (5)	72.43 (3)	70.27 (2)	66.45 (4)
Random Tree	69.49 (1)	64.38 (3)	63.29 (4)	67.53 (2)	60.36 (5)
Airlines	79.43 (2)	73.29 (4)	76.35 (3)	81.25 (1)	70.32 (5)
KDD99	72.49 (3)	76.38 (1)	74.19 (2)	69.57 (5)	70.26 (4)
Covertype	81.24 (1)	75.24 (3)	72.49 (4)	78.34 (2)	69.57 (5)
Poker Hand	68.38 (2)	62.54 (5)	71.37 (1)	66.43 (3)	64.56 (4)
Sensor Data	73.48 (1)	69.42 (3)	66.38 (4)	70.43 (2)	63.27 (5)
Average ranks	**1.7**	**3.3**	**3.2**	**2.6**	**4.2**

Table 8: CPU processing time of the algorithms using Hidden Markov Models.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Hyperplane	16.47 (1)	23.45 (2)	57.23 (5)	29.31 (3)	35.46 (4)
Stagger	17.38 (2)	33.45 (5)	22.26 (4)	20.13 (3)	9.16 (1)
LED	22.98 (5)	15.68 (3)	13.68 (2)	12.52 (1)	17.98 (4)
SEA	13.87 (1)	24.05 (3)	20.03 (2)	26.43 (4)	38.45 (5)
Random Tree	30.26 (1)	38.18 (4)	37.42 (3)	31.43 (2)	47.53 (5)
Airlines	63.24 (2)	60.48 (1)	82.47 (4)	80.25 (3)	84.26 (5)
KDD99	16.68 (1)	28.14 (4)	30.53 (5)	20.23 (2)	26.68 (3)
Covertype	14.64 (3)	13.48 (2)	26.48 (4)	10.41 (1)	36.17 (5)
Poker Hand	20.25 (2)	28.35 (4)	16.67 (1)	26.69 (3)	30.23 (5)
Sensor Data	72.36 (1)	96.45 (5)	90.12 (4)	80.12 (2)	86.57 (3)
Average ranks	**1.9**	**3.3**	**3.4**	**2.3**	**4.0**

Table 9: Memory consumption of the five algorithms on real-world datasets.

Dataset	RACE	CALMID	DUE	SOFEnsemble	AWDOB
Airlines	36.25 (1)	44.29 (3)	42.39 (2)	61.23 (4)	57.31 (5)
Covertype	43.17 (2)	28.34 (1)	51.42 (3)	63.48 (5)	59.63 (4)
Poker Hand	66.37 (1)	69.47 (2)	79.45 (5)	77.36 (4)	71.26 (3)
KDD99	78.36 (3)	76.39 (2)	73.43 (1)	83.43 (4)	88.34 (5)
Sensor Data	59.34 (1)	67.34 (3)	62.41 (2)	87.63 (5)	79.23 (4)
Average rank	**1.6**	**2.2**	**2.6**	**4.4**	**4.2**

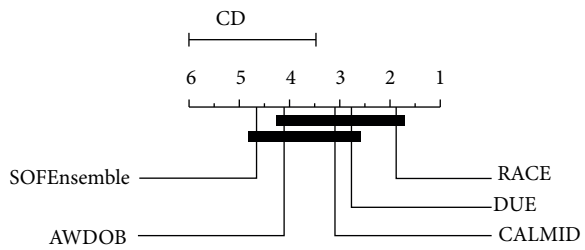

Figure 12: Average rank diagram of compared algorithms.

6. Conclusion

This paper presented a novel and evolving algorithm called Recurrent Adaptive Classifier Ensemble (RACE) to handle recurring concepts. RACE stores previously learned highly diverse models that are adapted using a new data chunk. We conducted two empirical experiments to evaluate the effectiveness of RACE in streaming environments associated with recurring concepts. In the first experiment, we created an ensemble of previously learned high diverse classifiers and used the concept of knowledge transfer to select diverse classifiers that are representative of the current concept from the latest data chunk. The drift detector was used in the algorithm to determine whether a drift has occurred or not. Results show that incorporating knowledge transfer and drift detection improves the prediction accuracy of the algorithm for nonstationary time series data.

In the second experiment, we investigated the behavior of the RACE algorithm when knowledge transfer is replaced by the Hidden Markov Model to predict upcoming drift with previously trained classifiers used to test similarity of past concepts to a present concept. Results show that using Hidden Markov Models to anticipate drift does not make the algorithm run efficiently enough for use in nonstationary time series data streams.

This paper has opened new avenues or directions for research, where recurring concepts are learned in a timely manner in nonstationary time series data with the least computational overheads. It is evident from the literature review that this area has not been fully explored. Even

though the RACE algorithm exudes novelty, it has its own weaknesses. The RACE algorithm can be computationally expensive as it requires large memory to store all the highly diverse classes and storage during concept transfer. Furthermore, as the ensemble increases in size, it slows down the convergence to recurring concepts as the concept transfer process will require more time, thus compromising its usability in nonstationary series data where a classification delay can prove costly. However, regardless of the weaknesses identified, this paper has uniquely opened new avenues of research in this area. The expectation is that many more approaches to handling recurring concepts in nonstationary time series data can be explored and developed, so that a comparison of prediction performance with the unique and novel RACE algorithm proposed in this research paper can be made.

Data Availability

The research used five artificial datasets, namely, (1) Random Tree generator, (2) SEA generator, (3) LED generator, (4) Stagger, and (5) Hyperplane. The real-world datasets used are (1) Covertype dataset, (2) Sensor Data, (3) KDD99 Cup dataset, (4) Poker Hand dataset, and (5) Airlines dataset. The artificial and real-world data used to support the findings of this study have been deposited in the following repositories and sources: (1) Random Tree generator: Cunningham P., Nowlan N., Delany S. J., and Haahr M., 2003, "A Case-Based Approach to Spam Filtering that Can Track Concept Drift", in the proceedings of ICCBR-2003 Workshop on Long-Lived CBR Systems. (2) SEA generator: Wang H., Fan W., Yu P.S., and Han J., 2003, "Mining Concept-Drifting Data Streams Using Ensemble Classifiers," in the Proceedings of 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining KDD-2003, ACM Press, pp. 226–235. (3) LED generator: Cunningham P., Nowlan N., Delany S.J., and Haahr M., 2003, "A Case-Based Approach to Spam Filtering that Can Track Concept Drift," in the Proceedings of ICCBR-2003 Workshop on Long-Lived CBR Systems. (4) Hyperplane: A. Bifet and R. Kirkby, Tutorial 1. Introduction to MOA Massive Online Analysis (Accessed 10.04.17). (5) Stagger dataset: J.C. Schlimmer and R.H. Granger Jr., "Incremental Learning from Noisy Data," Vol. 1, 1986, pp. 317–354. Real-world datasets used are (1) Covertype dataset, (2) Airlines dataset, (3) KDD99 dataset, (4) Poker Hand dataset, and (5) Sensor Data-Intel Lab Data.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] T. Lane and C. Brodley, "Approaches to online learning and concept drift for user identification in computer security," in *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp. 259–263, Menlo Park, CA, USA, September 1998.

[2] S. Jane Delany, P. Cunningham, A. Tsymbal, and L. Coyle, "A case-based technique for tracking concept drift in spam filtering," *Knowledge-Based Systems*, vol. 18, no. 4-5, pp. 187–195, 2005.

[3] 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 (KDD'03)*, pp. 226–235, Washington, DC, USA, August 2003.

[4] A. Tsymbal, "The problem of concept drift: definitions and related work," Technical Report, Department of Computer Science, Trinity-College, Dublin, Ireland, 2004.

[5] I. Zliobaite, M. Pechenizkiy, and J. Gama, "An Overview of concept drift applications," in *Big Data Analysis: New Algorithms for a New Society*Springer International Publishing, Berlin, Germany, 2016.

[6] M. Hassan, P. Kostakos, M. Cortes, T. Anagnostopoulos, S. Pirttikangas, and E. Gilman, "Concept drift adaptation techniques in distributed environment for real-world data streams," *Smart Cities*, vol. 4, pp. 349–371, 2021.

[7] M. Sarnovsky and M. Kolarik, "Classification of the drifting data streams using heterogeneous diversified dynamic class-weighted ensembles," *PeersJ Computer Science*, vol. 7, no. 2, 2021.

[8] A. Liu, J. Lu, and G. Zhang, "Diverse instances-weighting ensemble based on region drift disagreement for concept drift adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 22, no. 1, pp. 293–307, 2021.

[9] M. Heusinger, C. Raab, and F. M. Schleif, "Passive Concept Drift handling via variations of learning vector quantization," *Neural Computing and Applications*, 2020.

[10] X. Zheng, P. Li, X. Hu, and K. Yu, "Semi-supervised classification on data streams with recurring concept drift and concept evolution," *Knowledge Based Systems*, vol. 215, 2021.

[11] K. Namitha and G. Santhosh Kumar, "Learning in the presence of concept recurrence in data stream clustering," *Journal of Big Data*, vol. 7, no. 1, p. 75, 2020.

[12] W. W. Y. Ng, J. Zhang, C. S. Lai, W. Pedrycz, L. L. lai, and X. Wang, "Cost Sensitive weighting and imbalance-reversed bagging for streaming imbalanced and concept drifting in electricity pricing classification," *IEEE Transations on Industrial Informatics*, vol. 15, no. 3, pp. 1585–1597, 2019.

[13] L. Zang, Y. Xiang, and W. Huang, "Drift-detection based incremental ensemble for reacting to different kinds of concept drift," in *Proceedings of the 2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*, pp. 107–114, Qingdao, China, 2019.

[14] I. Baidari and N. Honnikoll, "Accuracy weighted diversity-based online boosting," *Expert Systems with Applications*, vol. 160, 2020.

[15] X. Gu, P. Angelov, and Z. Zhao, "Self-organising fuzzy inference ensemble system for big streaming data classifications," *Knowledge Based Systems*, vol. 218, 2021.

[16] L. Zeng, W. Huang, Y. Xiong, S. Ren, and T. Zhu, "Incremental learning imbalanced data streams with concept drift: the dynamic updated ensemble algorithm," *Knowledge-Based Systems*, vol. 195, Article ID 105694, 2020.

[17] W. Liu, H. Zhang, Z. Ding, Q. Liu, and C. Zhu, "A comprehensive active learning method for multiclass imbalanced data streams with concept drift," *Knowledge Based Systems*, vol. 215, 2021.

[18] G. Yule, "On the association of attributes in statistics: with illustrations from the material of the childhood society," *Philosophical Transactions of the Royal Society of London, Series A*, vol. 194, no. 252–261, pp. 257–319, 1900.

[19] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 5, pp. 730–742, 2010.

[20] E. K. Tang, P. N. Suganthan, and X. Yao, "An analysis of diversity measures," *Machine Learning*, vol. 65, no. 1, pp. 247–271, 2006.

[21] M. Baena-Garcia, J. Del Campo-Avila, R. Fidalgo, and A. Bifet, "Early drift detection method," in *Proceedings of the Fourth ECML PKDD International Workshop Knowledge Discovery from Data Streams (IWKDDS'06)*, pp. 77–86, New York, NY, USA, 2006.

[22] R. Elwell and R. Polikar, "Incremental learning of concept drift in nonstationary environments," *IEEE Transactions on Neural Networks*, vol. 22, no. 10, pp. 1517–1531, 2011.

[23] A. Asuncion and D. Newman, *UCI Machine Learning Repository*, https://www.ics.uci.edu/mlearn/MLRepository.html, 2007.

[24] P. Domingos and G. Hulten, "Mining high speed data streams," in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining ACM*, pp. 71–80, San Diego, CA, USA, August 2000.

[25] 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 (KDD'01)*, pp. 317–354, San Francisco, CA, USA, August 2001.

[26] Massive Online Analysis (MOA) datasets, 2010, https://moa.cms.waikato.ac.nz/datasets.

[27] The UCI KDD Archive, *Information and Computer Science*, University of California, Irvine, CA, USA, 2009, https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[28] Massive Online Analysis (MOA) datasets, 2010, https://moa.cms.waikato.ac.nz/datsets/Covertype.

[29] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavalada, "New Ensemble methods for evolving data streams," in *Proceedings of the Fifteenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining ACM*, pp. 139–148, Paris, France, June 2009.

[30] Intel Lab data, https://db.csail.mit.edu/labdata//labdata.html, 2004.

[31] N. Settouti, M. E. A. Bechar, and M. A. Chikh, "Statistical comparisons of the top 10 algorithms in data mining for classi cation task," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 1, pp. 46–51, 2016.

[32] J. Demsar, "Statistical comparison of classifiers over multiple datasets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.