International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 / https://doi.org/10.14445/22315381/IJETT-V70I9P212

Volume 70 Issue 9, 119-130, September 2022
© 2022 Seventh Sense Research Group®

Original Article

A Systematic Ensemble Approach for Concept Drift Detector Selection in Data Stream Classifiers

Rucha Chetan Samant[1], Suhas H. Patil[2], Rahul Nand Sinha[3], Amol K. Kadam[4]

[1,2,3,4]*Department of Computer Engineering, Bharati Vidyapeeth (Deemed to be University), College of Engineering, Pune, India.*

[1]*Corresponding Author :* ruchasamant25@gmail.com

Received: 04 June 2022 Revised: 17 August 2022 Accepted: 11 September 2022 Published: 17 September 2022

Abstract - *Most applications generate data in a stream format in the Big Data world. Mining this data stream is considered necessary to extract meaningful information from such a large amount of data. To be successful in this well-known field of analytics, traditional classification, clustering, and aggregation techniques must be improved. Ensemble-based classifiers developed using bagging, boosting, or hybrid methods outperformed traditional single classifiers. The ensemble concept has been shown to improve classifier accuracy and diversity in design. At the same time, using a drift detector to address the concept drift issue of a data stream has yielded fantastic results. The primary goal of this proposed system is to provide a suitable methodology for selecting an appropriate drift detector for an effective ensemble classifier by combining a cutting-edge base ensemble classifier with standard drift detectors. Similarly, this paper also examined a proposed boosting ensemble strategy using several drift detectors to determine the most effective combination to address all types of concept drift. The results and analysis discussed in this paper are expected to be relevant and useful for selecting the proper parameters of drift detectors and designing strong ensemble classifiers.*

Keywords - *Concept Drift, Data Stream mining, Drift Detector, Ensemble-based learning, Real-time data analysis.*

1. Introduction

A key concern with data stream analysis is the lack of a proper strategy for dealing with the continuous flow of data. It is a difficult area for data mining because the technique must recognise changes efficiently to extract meaningful data from it. It must keep reliable stats on this revolving data. Likewise, the predictive model should be revised as input changes.

When multiple models of base techniques are applied to the same input to improve overall performance, the term "ensemble" is used. This same concept is used in data stream mining, where ensembles are designed using Bagging, Boosting [1], and stacking[2]approaches. The change in data concept must be handled with caution because it affects the accuracy of prediction models. Drift detection methods detect such changes, allowing the prediction model to update under the drift rate.

Oza and Russell designed Online Boosting (OZABOOST)[1] to deal with big data stream problems for the first time in the literature. They used a Poisson distribution to predict behavioral drift patterns. Researchers have developed a variety of ensemble algorithms that manage data in stream format by overcoming speed, time, and volume issues [3]–[6]. This paper's major goal is to look into ways to select a drift handler that can effectively manage all types of drifted data

The significant contributions of this work include the following:

- To assess the accuracy of the prediction, an ensemble of 10 base classifiers is utilized rather than a single classifier.
- Three separate ensemble classifiers were used to test three different drift detection algorithms.
- Two majors are used: kappa and accuracy.
- To make more precise judgments, experimenters use a variety of data streams, including real and synthetic data streams that contain rapid, complex, and progressive drift.
- Comparisons are made based on performance.

Many academics have employed the ensemble to address large stream difficulties while performing data stream analysis. Different strategies were used to produce these ensembles. This research will focus on a systematic approach for emphasizing the use of drift detectors in ensemble creation. Similarly, the behavior of our suggested boosting ensemble classifier is examined on drifted data streams using a variety of drift detector algorithms. The following is how the paper is organized. Section 2's literature review examines ensembles of various types and covers the most popular and effective methods for drift detection. Section 3 contains information on the experimental setup and data sets. Section 4 contains results and comparative analysis. Section 5 deals with the conclusion.

2. Literature Review

2.1. Ensemble Method

To deal with different drifts in data streams, this research emphasises the combination of ensemble classifier and drift detection method. This study considered various ensemble-based classification methods as well as drift detectors. Some of the most well-known ensemble-based classifiers were developed to address the concept drift problem. In this part, the ensemble classifier literature is covered in more detail.

Accuracy Weighted Ensembles AWE[7], developed by Haixun Wang et al., employs a weighting strategy for ensemble selection. A weighted ensemble classifier design strategy is suggested by this method. It used naive Bayesian-like base classifiers, C4.5, RIPPER, and input in the form of sequential data chunks. If the expert model predicts that instance Ai (A) of the input data belongs to label l, this is probable. In the case of ensemble development, the output of several base experts(1..n) is pooled, and the final output is just the average of all as follows:

$$f(A) = i\backslash k * \sum_{k=1}^{k} f(\mathrm{A}) \qquad (1)$$

Here f_{l}^{i} (A) is an output of ith base expert in the system. The approximate probability, the bias, and the model's variance are all considered when calculating the prediction probability for a particular case. To lessen training inconsistencies overall, the AWE technique considers variance-related errors. As a result, AWE created the weighted technique as opposed to the general ensemble strategy, which averages the outputs of the basic experts. A weight Wi has been given to each base expert. This weight is inversely related to the base expert's estimated mistake.

By combining a window-based ADWIN[8] drift detection approach with online bagging, Albert Bifet et al. created the Leveraging Bagging method[9]. The method's creators have improved bagging performance by altering the data input and output identification processes. Variation can be reduced by changing data resampling by utilizing various lambda values. The result is rebuilt by transforming a multiclass label into a binary class label and utilizing error-correcting coding. The authors utilized bagging without input data stream replacement, half-bagging, and sub-bagging, increasing performance time but decreasing model accuracy.
To address the shortcomings of AWE[9], Dariusz Brzezinski and Jerzy Stefanowski presented Accuracy Updated Ensemble (AUE)[12]. AUE focuses on online stream processing rather than working with data blocks. Therefore it not only modifies expert weights but also updates them based on current performance. Similar changes have been made to the AUE error computation, which now takes into account mean square error for both random prediction and each instance as follows:

$$MSR_{\mathrm{e}} = 1/|Ai| \sum_{A,l\in Ai} \left(1 - f\ (A)\right)^2 \qquad (2)$$

$$MSRr = \sum_{be} p(be)\left(1 - p(be)\right)^2 \qquad (3)$$

Here equation 2 depicts the mean square error calculation for each instance, whereas eq. 3 gives random prediction errors used in the AUE method. Final, the weight of each expert is calculated as W_{be} given in eq. 4.

$$W_{be} = MSRr - MSR_{\mathrm{e}} \qquad (4)$$

Even while classifier performance has increased, using comparable examples for training reduces classifier variety.
By combining the advantages of block and incremental processing, Dariusz Brzezinski and Jerzy Stefanowski created the next technique, also known as Online Accuracy Updated Ensemble OAUE[10]. An incremental online ensemble is created by changing three major components of a block-based ensemble. The authors used online component evaluation, an incremental learner, and a drift detector to evaluate input. This Accuracy Updated Ensemble weights expert members according to their errors in constant time and memory. The weights of the base expert in the OAUE method are calculated on a window of the last n errors, which is not similar to the sliding window algorithms used in data stream classification. Similarly, this method reconstructs the ensemble regularly by replacing base experts[10]. The error calculation procedure utilized here is the same as in equation 4; however, for newly recruited experts, the error is always treated as zero for the first n occasions.

The ensemble basis classifier is thoroughly examined to determine the most effective methods for fending off each drift. A comparative examination of ensemble-based classifier design at the cutting edge is conducted for various kinds of datasets. This study emphasizes the suitability and restrictions of each ensemble designing technique[11].
In a recent study, L. Durga and R. Deepu used the convergence of image analysis and learning algorithms to extend graph logical notions to suit the big five personality classifications. Correlations between the graph's logical properties and the big five personality observations are also made using clustering-based analysis. It has also developed an ensemble training classifier model for predicting the Big Five Personality Traits using Graph logical Features[12].

Making decisions based on several criteria is a crucial challenge in big data study. It often uses the most recent ML techniques to provide insights into big data, including decision-making algorithms and deep learning methods based on many criteria. To raise the dual of runtime and increase the potentiality and efficacy of the overall system, the derivations are developed to go with the approximations. Deep learning and multicriteria decision-based decision-making challenges are used in many industries, including agriculture. By offering new research with the merging approaches of data-driven techniques, S. L.V. Papineni et al. have presented various applications that involve the principles of deep learning techniques and utilizing the multicriteria methods for problems faced in big data analytics[13].

The speed of data generation has recently made big data analytics more important. Many creative solutions have been developed to address the big data challenge, including distributed systems, network systems, and huge frameworks. After completing tests at Huawei Noah's Ark Lab, the writers Bifet et al. developed the idea of StreamDM[14]. The authors improved the Spark API by utilizing Spark streaming, a scalable data stream processing tool. Using a machine learning library, a new piece of free software for data mining is produced. StreamDM is simple for users, programmers, and researchers to use and extend. The article claims this is the first Spark Streaming library that includes sophisticated stream mining techniques[14].

2.2. Drift detection methods

Drift detectors play an important role in improving the accuracy of the classification model. The primary characteristics of a drift detector are:

1. To determine when a change has occurred.
2. To preserve design consistency by retaining more relevant instances.
3. If a change takes place, update the model.

This section focuses on three basic drift detection techniques as follows. Albert Bifet and Ricard Gavalda have created a drift detection technique that employs a sliding window. This method, rather than using a stationary window, use a window which can shrink or expand based on data attitude. All of these tasks are carried out using this windowing strategy in this Adaptive Windowing (ADWIN)[15]drift detection method. In this method, a fixed-size window of size W is maintained for longer until there is no change in the incoming data stream labels. The drift d is detected by calculating the difference between the actual estimation (e) and the expected estimation ($\bar{e}$). As a result, Window is used to track the failure rate of each model. A threshold value is maintained to keep an upper limit on the error; if the error rate increases, the model is updated to account for the changes and, in some cases, rebuilt if required.

Gama, Medas, and colleagues presented the Drift Detection Method (DDM)[16], a novel technique for identifying drift. The model's error rate is calculated using the binomial distribution, and it has shown promise in identifying abrupt drift. This error rate is calculated by comparing the probability distribution ($\acute{P}$) of the incoming observations ($\acute{O}$) to the error rate of the prediction model ($\dot{\varepsilon}$). The DDM has two phases. First, if the results differ, the model activates and saves samples of the incoming instances for future reference. It is regarded as a warning, and an alarming level is set. A drift level is detected for upcoming instances if the change in incoming instances exceeds the threshold. It is an alarm to delete the rehabilitation of the existing model and bring in a new model with previously saved instances from the warning bell. It is important to create a new model to adapt to the new notion because the present examples are connected to new ideas.

Manuel Baena-Garca1 et al. [17] created an advanced version of DDM known as the Early Drift Detection Model (EDDM), which can handle gradual drift in a data stream. As DDM employs a simple error calculation formula, this model could not handle gradual drift in data streams because the error rate of gradual drift increases slowly. To address this issue, EDDM was created using an advanced error calculation technique. EDDM considered the distance between two errors rather than just the number of errors. It has been observed that this method detects both gradual and sudden drift with greater accuracy.

3. Datasets and Experiments

Data stream (*DS)* can be described as a continuous flow with multiple instances as below:

$DS \leftarrow \Sigma\ (N_i, Y_i)$

Ni ← (n1, n2,n3⋯...ni) attributes of stream mapped to class outcome Y_i. In the case of data stream classification accuracy of the prediction model decreases if the new arrival of the stream has different distribution than that of the learning samples. This concept drift problem hampers the performance of the prediction model.

3.1. Datasets

Data stream experiments make use of a standard data set that is divided into two categories. Many synthetic datasets are used in the literature. This research uses eight standard data sets, each belonging to a different drift category. This study uses five synthetic data streams and three real datasets as a benchmark. The stream generator can create a stream by adding noise, specifying the number of classes and attributes, or using a sigmoid function. For comparisons, the majority of data stream classification algorithms used Random RBF and Hyperplane, SEA[18], Sine and waveform synthetic streams, and Electricity, Cover type and Poker hand real data sets[18]–[20]. These streams are produced using the Massive Online Analysis [21] java framework.

1. *Electricity*: The Electricity dataset contains 45,312 instances and eight attributes from the Electricity Market of Australia. These prices get updated every five minutes, and the class label indicates how the price has changed with a 24-hour moving average. Here the challenge is to forecast whether the price will rise or fall.
2. *Random RBF Generator (Radial Basis Function):* It generates complex concept drifts by randomly generating x number of centroids. Each center is assigned randomly, with a single standard deviation, a class label, and a weight. It has a lot of classes in it, but there isn't any noise in it. It has a gradual drift pattern.
3. *Hyper Plane:* Continuous addition changes the position and orientation of a hyperplane in d-dimensional space. It is divided into two classes and contains 5% noise. It is an incremental concept drift type with noise added.
4. SEA: This dataset is an example of significant concept drift. Three attributes are used to generate it, of which only the first two are relevant. Here values of attributes range from 0 to 10.
5. Sine: The dataset contains two related attributes, each with values uniformly distributed between zero and one. As a sine wave, all values are positive for the first interval and change for the next cycle, becoming reversed.
6. Waveform: The waveform comprises a stream with three decision classes, and 40 different attributes describe the instances. It is an example of gradual concept drift.
7. Cover Type: This data set consists of information of 30 x 30-meter cells obtained from the US Forest Service, Region 2 Resource Information System. It is made up of 581, 012 instances and 54 attributes. It has appeared in several papers on data stream classification.
8. Poker Hand: Every instance of this dataset represents a card game hand made up of five playing cards drawn from a 52-card deck. In this case, a card is identified by two parts: its suit and its rank. It is to forecast a total of ten attributes.

3.2. Experiment Setup

In the above experiment, all three data streams are generated with 1 lack, 2 lacks and 5 lacks. It has a sampling frequency of 1000 and a window size of 1000.

Hoeffding tree, naïve based, and our newly proposed boosting-based ensemble classifier methods were each considered for the experiment in 3 separate ensembles with 10 base classifiers [22]. Our recent paper covers the construction concept of the new Boosting-based ensemble classification approach [23]. For ADWIN drift detector window's delta value is set to 0.02 by default. The minimum number of instances is set to 30 with a warning level of 2 and an output control level of 3. A basic classification performance evaluator with a sampling frequency of 1k and a window width of 1k is used to assess classifier accuracy. The evaluate prequential method is used, which tests and trains the classifier with each example in order. The behavior of all these drift detection methods is tested using different data streams of different sizes. Some data sets have noise added to them. The experiments are run on core i5processor with 4GB Ram, Windows OS. All these methods are run in MOA[21] framework; similarly, our proposed method is also implemented in java and plugged in with MOA code.

4. Comparative Analysis

Three different ensemble classifiers are used in the experiments: naive Bayes, Hoeffding trees, and our proposed enhanced boosting-based ensemble classifier[22]. These methods are subjected to a test to compare the effectiveness of the three drift detector techniques. All three drift detection methods, i.e. ADWIN, DDM and EDDM, were run on the same data sets, and their accuracy and kappa measures were calculated. These three drift detector algorithms are available on MOA[21]framework. Following are the results of the experiments. This section concentrates on the efficacy of the drift detector approach. Therefore, only the Hoeffding tree ensemble classifier is used to compare the performance of the three drift detectors.

4.1. Accuracy Analysis

The calculation of accuracy is as:

$$Accuracy = \left(\frac{correctly\ classified\ instances}{N}\right) * 100 \qquad (5)$$

Where, N ← Total Instances

Table 1 shows the accuracies observed for each drift detector technique on artificial and real datasets using the Hoeffding tree as the base classifier. Bold values highlight the best outcomes. The average of multiple runs on real and artificial datasets with 1 lakh, 2 lakh, and 5 lakh instances is calculated and placed for reference.

A sudden drift occurs when a new concept entirely substitutes an old one. SEA and Sine datasets are examples of this type of drift. As shown in table 1, ADWIN and DDM obtained approximately the same 88.00+ percent and 98.00+ percent accuracy for the SEA and Sine datasets, respectively. Similarly, EDDM's accuracy for these two datasets is near-perfect. As mentioned in EDDM [17], both the DDM and EDDM are designed to detect slow progressive changes because longer references are required for drift detection.
In contrast, the Window-based approach (ADWIN) is good at identifying rapid changes. While windowing approach, ADWIN has completely failed to recover from identifying drift in a Cover type data set where values are projected based on 54 attributes. In contrast, the DDM method recovers quickly and with greater accuracy.

The occurrence of drift in the Electricity dataset is unknown in advance because it is a real-world dataset. The price variances on a one-day average are only shown at the class level, removing the impact of longer-term price patterns. As a result, this is a data input for a short-term forecast. Electricity is a binary dataset, and as shown in table 1, DDM is more responsive to this dataset and outperforms other approaches.

When there is noisy data, as in Poker hand and RBF Generator datasets, EDDM seems more accurate than the other two methods, allowing it to identify changes and improve performance.

Each instance of the poker hand dataset's 10 attributes is created using a hand-drawn at random from a deck. The change in prediction is unknown; therefore, memorizing the results for longer periods is not recommended. As a result, EDDM has demonstrated its high accuracy in identifying changes. As shown in Table 1, the EDDM method accuracy for the poker hand dataset is 4 percent more than the DDM method.

When analyzing the RBF generator dataset with incremental concept drifts, EDDM again has the highest accuracy, followed by DDM and ADWIN. The progressive modifications in this dataset take twice as long to complete. As a result, all drift detectors took longer to notice a change and took time to recover.

ADWIN performed admirably on the hyperplane dataset, which has complex drift. In real-world datasets, however, ADWIN has performed poorly. According to the findings, abrupt drift identification accuracy of ADWIN and DDM is the same in the SEA and Sine. To summarise, the ADWIN window method and error prediction between consecutive instances of DDM is ideal for detecting rapid drift.

DDM and EDDM work on error calculation mechanisms to detect drift, although their criteria differ. In the case of realistic data sets, both DMM and EDDM produce excellent results, such as DMM for cover type and EDDM for poker hand. In the case of RBF generator drift, EDDM has shown good accuracy.

The kappa measure: It is a second performance matrix that is used to examine the difference between the system's real accuracy and its random accuracy. Throughout the process, it is used to predict model performance.

Table 2 shows the kappa measures for three drift detection techniques on all eight datasets with the Hoeffding tree as the base classifier. For complex types of drift, ADWIN performance remains steady since it is high for hyperplane; however, for real data sets, once the drift has occurred, it is impossible to recover from it; hence ADWIN throughout performance remains low for these datasets.

DDM and ADWIN have recovered quickly from abrupt types of drift, such as SEA and Waveform datasets, and their performance metrics have never dropped. However, EDDM has outperformed gradual types of drift (RBF generator).

Table 1. Accuracy comparison of ADWIN, DDM and EDDM Methods with Hoeffding classifier

Datasets	**Drift Detector Method**		
	ADWIN	**DDM**	**EDDM**
Hyper plane	54.30	89.14	86.56
SEA	88.02	88.10	87.71
Sine	98.46	98.72	98.14
Waveform	81.11	81.76	80.92
Cover Type	22.41	85.38	83.905
Electricity	43.44	86.70	85.72
Poker Hand	53.49	72.76	78.9
RBF Generator	84.07	90.68	91.58

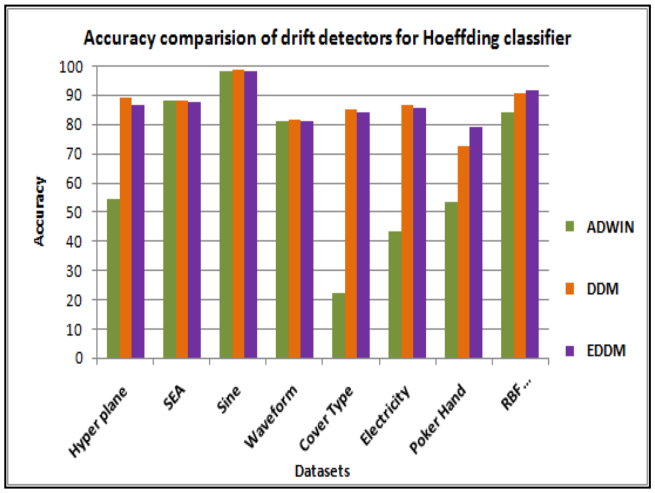

Fig. 1 Accuracy comparison of Drift Detection methods on real and synthetic datasets.

Table 2. Kappa measure for ADWIN, DDM and EDDM Methods using Hoeffding Tree classifier

Datasets	**Drift Detector Method**		
	ADWIN	**DDM**	**EDDM**
Hyper plane	11.70	78.29	77.52
SEA	74	74.50	72.77
sine	97.43	97.53	96.48
Waveform	74.13	74.75	72.06
Cover type	1	76.58	73.77
Electricity	1.31	63.84	70.82
Poker Hand	9.8	56.32	62.49
RBF Generator	72.77	81.33	83.14

The efficiency of drift detectors on drifted datasets is depicted graphically in Fig. 1. The model performance on various datasets is compared using the kappa metric in Fig. 2. As shown in figs. 1 and 2, the window-based approach (ADWIN) of drift identification performed well for sine datasets. In contrast, the performance of ADWIN for an RBF generator, cover type, and poker hand is lower than the other two methods, as shown in fig. 2. This is related to the data stream's nature and the drift detector's needs for operation.

In most datasets, especially real datasets, error-based drift detection algorithms have demonstrated good results. The probabilistic error has failed in the case of gradual drift.

The primary goal of this paper is to evaluate the effectiveness of coupling different classifiers with different drift detectors to offer a robust solution to the data stream concept drift problem. The most popular base classifiers, Hoeffding trees and Naive bayes, and a proposed enhanced boosting-like classifier, are the ones that are focused on here.

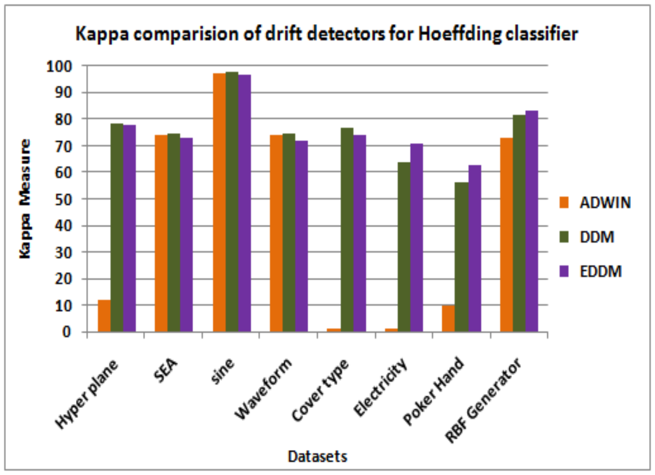

Fig. 2 Kappa comparison of Drift Detection methods on the real and synthetic dataset using Hoeffding tree classifier.

Table 3 shows the detailed accuracy calculation of all three drift detectors using the Hoeffding tree and the naive bayes base learner. In the gradual drift of the RBF generator dataset, the Hoeffding tree showed a higher recovery rate than Naive Bayes for all three drift detector techniques.

In the case of complex drift like in the Hyperplane dataset, a naive Bayes works consistently and achieves excellent results in all three drift detector methods. In this case, the independence of the naive predictor rule had a significant effect on classification.

Both classifications provided nearly identical accuracy in the case of all three real-world data sets. There is no discernible difference in the performance of these drift detectors. Only in the case of the poker hand and the electricity data set did DDM and EDDM improve their accuracy. It could be because of the size of the data sets.
In the SEA and Waveform datasets, which are examples of abrupt and gradual drift, respectively, the prediction accuracy of both classification methods is the same. None of the three drift detectors for these datasets ever fall below the standard level.

Both the SEA and Sine data sets exhibit abrupt drift behavior, but according to the observations, the Hoeffding tree captures and recovers from drift more accurately than the naive Bayes.

A statistic based on the nonparametric Bonferroni correction is applied to the results of two methods simultaneously to supplement the accuracy analysis. The results are shown in Table 4. For this test, a null hypothesis was created that stated, "All methods are statistically equal." If the results fail to reject the null hypothesis, there is no difference between the two methods. If the null hypothesis is rejected, there is a significant difference between the two methods.

The parameter alpha is initially set to 0.05 for comparison, but because there are three methods to compare, the value of benoferrial alpha is changed to 0.016666667. When comparing ADWIN and DDM, the holms value is =0.05. When comparing ADWIN and EDDM, the holms value is =0.025, much greater than the t-test value, indicating a significant difference between these two methods.

The DMM and EDDM holms value are 0.016666667, which is very low compared to the t-test value, which rejects the null hypothesis and proves that these two methods are the same. Thus, even if the results of classifiers are similar in some methods, all drift detection methods are distinct.

Table 3. Accuracy comparison for ADWIN, DDM and EDDM Methods using Naïve Bayes and Hoeffding Tree classifiers

	Naïve Bayes classifier			**Hoeffding Tree Classifier**		
Datasets	**ADWIN**	**DDM**	**EDDM**	**ADWIN**	**DDM**	**EDDM**
Random RBF Generator (1L)	71.85	71.85	71.71	81.26	81.26	81.11
Random RBF Generator (2L)	72.06	72.06	71.71	83.97	83.97	81.50

Random RBF Generator (5L)	68.96	68.96	71.71	86.98	86.98	82.60
Hyper plane (1L)	92.92	92.92	92.92	53.34	89.14	88.76
Hyper plane (2L)	93.34	93.34	93.34	55.79	89.03	83.93
Hyper plane (5L)	93.70	93.70	93.70	55.79	89.26	87.00
SEA (1L)	87.94	87.94	87.53	87.37	87.62	87.49
SEA (2L)	88.05	88.05	88.22	88.01	88.01	88.22
SEA (5L)	88.14	88.14	87.42	88.67	88.67	87.42
Sine (1L)	92.55	92.55	92.8	97.93	97.9	97.90
Sine (2L)	92.83	92.83	98.65	98.46	98.45	97.86
Sine (5L)	93.14	93.14	98.72	98.98	98.99	98.65
Waveform(1L)	80.42	80.48	80.48	80.52	80.24	80.08
Waveform (2L)	80.51	80.51	80.78	80.52	81.23	80.78
Waveform (5L)	80.50	80.50	80.90	82.28	82.08	81.90
Cover Type (1L)	21.51	80.88	81.40	21.51	83.52	82.20
Cover Type (2L)	22.41	84.45	83.75	22.41	86.29	83.75
Cover Type (5L)	22.24	86.43	84.89	23.1	87.24	84.89
Poker Hand (1L)	54.18	75.42	79.18	54.18	75.42	79.18
Poker Hand (2L)	54.11	74.45	79.08	53.18	74.45	79.08
Poker Hand (5L)	53.18	68.42	78.41	53.1	68.42	78.41
Electricity (1L)	43.44	82.48	85.72	43.44	86.70	86.19
Electricity (2L)	43.44	82.48	85.72	43.44	86.70	86.19
Electricity (5L)	43.44	82.48	85.20	43.44	86.70	86.19

Table 4. Comparison results for ADWIN, DDM and EDDM Methods using Bonferroni Correlation Method

Compared Methods		Base classifier - Hoeffding Tree			
		T-test	Bonferroni	P-Rank	Holm
ADWIN	**DDM**	0.00044755	0.016666667	3	0.05
ADWIN	**EDDM**	0.000455139	0.016666667	2	0.025
DDM	**EDDM**	0.877068374	0.016666667	1	0.016667

4.1.1. Data Drift Analysis

The behavior of all three drift detectors is depicted graphically for three types of drift: abrupt, complex, and gradual. It shows how to change detectors react when they encounter a change in concepts.

As shown in Figure 3, the ADWIN technique cannot recover after encountering the first drift after 5000 instances of examination. As can be seen from the graph, ADWIN's performance has steadily declined.DDM and EDDM had a lower performance at 5000, 10000, and 15000 instances, but after continuous training, the model adapted to changes and the accuracy of both methods improved further.

Figure 4 compares drift detector reactions for a real-world data set Poker hand. In this dataset, all three methods react differently because the drift is unknown in advance, and each method's recovery rate differs.

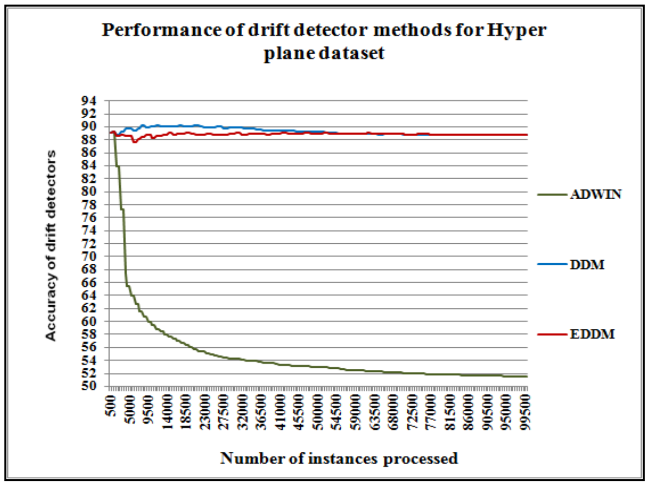

Fig. 3 Performance of Drift Detection methods for Hyper Plane dataset.

The probability-based error calculation method, EDDM, is the winner here. The graph clearly shows that the drift occurs regularly; after training with around 15000 instances, the accuracy of DDM and EDDM never falls below a certain level. In comparison, the ADWIN method is incapable of dealing with frequent changes.

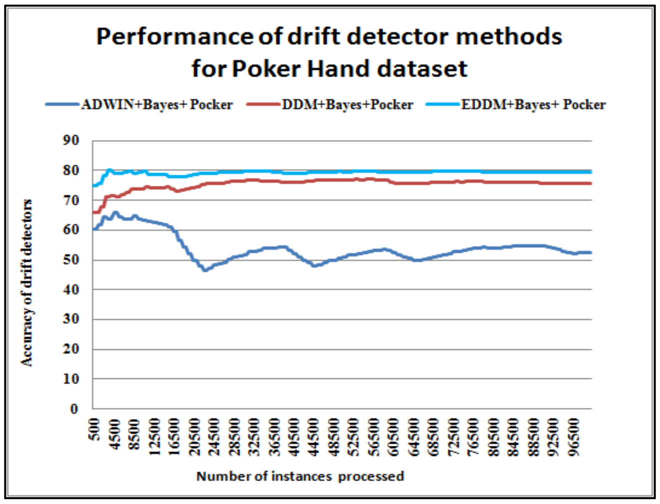

Fig. 4 Performance of Drift Detection methods with Nave bayes classifier for Poker Hand dataset

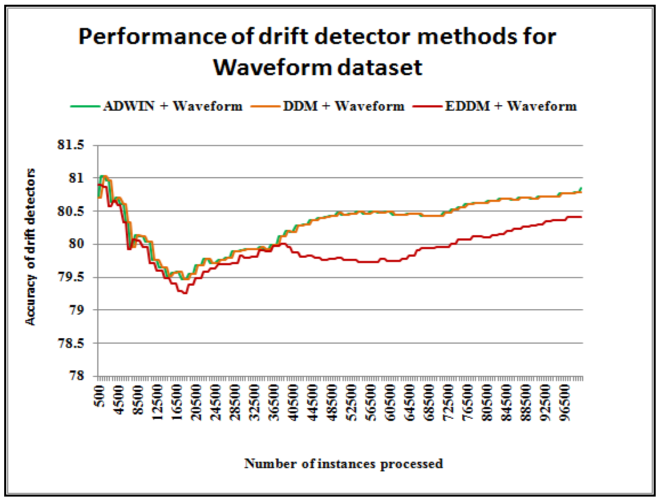

Fig. 5 Performance of Drift Detection methods with Hoeffding tree classifier for Waveform dataset

Finally, Figure 5 depicts the reaction to a waveform dataset with gradual drift. As shown in the graph, changes in concepts began at the beginning and have continued to change, resulting in the performance of all three methods gradually degrading. DDM and ADWIN have gradually recovered their performance, but the EDDM method has consistently degraded throughout the experiment.

4.2. Comparative Analysis

A Boosting-based ensemble method is proposed[22] by changing the instance selection strategy in the exiting Boosting-based online ensemble method (BOLE)[24]. This proposed method is tested in this research using various drift detectors to select a suitable drift detector for all types of drifted datasets. The comparison of this proposed boosting-based technique with Hoeffding trees and Naive Bayes classifiers is shown in table 4 below. Five distinct datasets are chosen for comparison, including the real-world datasets of electricity and the hyperplane with gradual drift and sine with rapid drift. The experiment uses one lakh, two lakh, and five lakh instances of synthetic datasets, and the averaged findings are shown in table 4.

It has been found that ADWIN's drift detector's accuracy for the Hyperplane data set is lower than that of other drift detection methods. Unlike the stagger dataset, this provided an accuracy of 99.95% to 99.96% for all three classifiers and all three drift detectors. Considering SEA datasets, the proposed method outperformed all others with a DDM drift detector accuracy of 88.41%, whereas the Hoeffding methods with ADWIN had the lowest accuracy.

For the analysis of the performance of drift detectors, classifier accuracy is considered. The proposed boosting ensemble approach for the ADWIN drift detector has demonstrated good performance for the electricity and sine datasets with 74.97% and 98.80%, respectively, as shown in figure 6.

The average performance on all five data sets for ADWIN using the proposed boosting ensemble approach, naive bayes, and Hoeffding trees is 89.42%, 83.52%, and 76.83%, respectively, as shown in figure 6. Except for Hyper plane data, all datasets with the DDM drift detector method with the proposed boosting ensemble approach have performed well. As depicted in figure 7, for the proposed boosting ensemble technique, naïve bayes, and Hoeffding trees, DDM recorded the highest average accuracy with 93.40%, 91.33%, and 92.51%, respectively.

Table 5 shows that our proposed boosting-based ensemble method has the highest average accuracy of 89.42%, 93.40%, and 93.02% in ADWIN, DDM, and EDDM drift detectors, respectively when the average results of all three classifier techniques are considered.

As in figure 8, the EDDM drift detector with the proposed boosting ensemble approach produced accuracy nearly on par with that of the DDM method but required more time. While EDDM has achieved outcomes of 91.62%, 92.69%, and 93.02% for Hoeffding trees, naïve bayes, and the proposed boosting ensemble approach, respectively.

The DDM drift detector performs better for all types of drifted data with less time than the other two approaches, as shown in figure 9. It might be related to the drift detector's error calculation formulation.

Table 5. Accuracy for ADWIN, DDM, and EDDM drift detectors for Hoeffding tree, Naïve Bayes and Proposed Boosting Ensemble method

	ADWIN			DDM			EDDM		
DataSets	Hoeffding Tree	Naïve Bayes	Proposed Method	Hoeffding Tree	Naïve Bayes	Proposed Method	Hoeffding Tree	Naïve Bayes	Proposed Method
Hyper plane	54.3	93.32	85.4	89.14	93.32	88.47	86.56	93.32	87.71
Sine	98.46	92.84	98.8	98.72	92.84	98.8	98.14	96.72	98.8
Stagger	99.95	99.95	99.95	99.88	99.96	99.97	99.96	99.95	99.97
Electricity	43.44	43.44	74.97	86.7	82.48	91.34	85.72	85.72	91.00
SEA	88.02	88.04	88	88.1	88.04	88.41	87.71	87.72	87.63
Average Performance	**76.83**	**83.52**	**89.42**	**92.51**	**91.33**	**93.40**	**91.62**	**92.69**	**93.02**

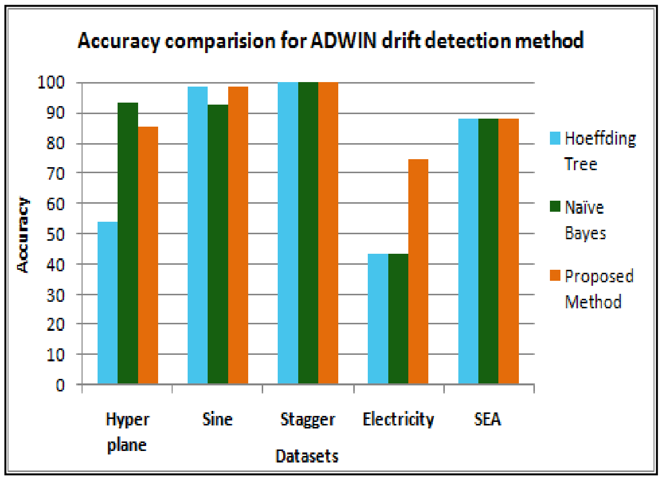

Fig. 6 Accuracy comparison of ADWIN Drift Detection methods for three different classifiers.

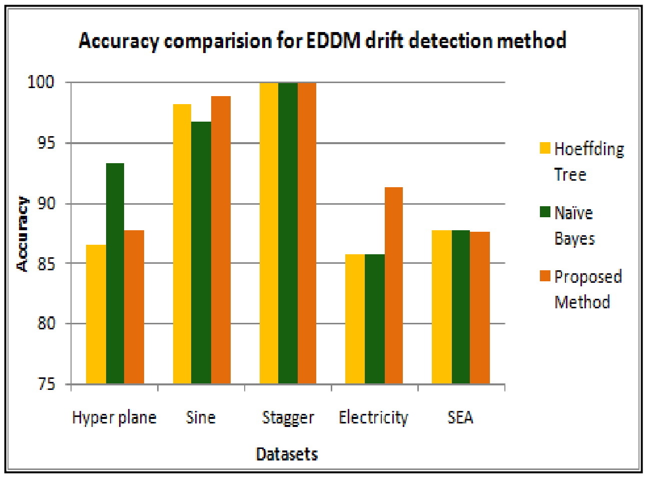

Fig. 8 Accuracy comparison of EDDM Drift Detection methods for three different classifiers

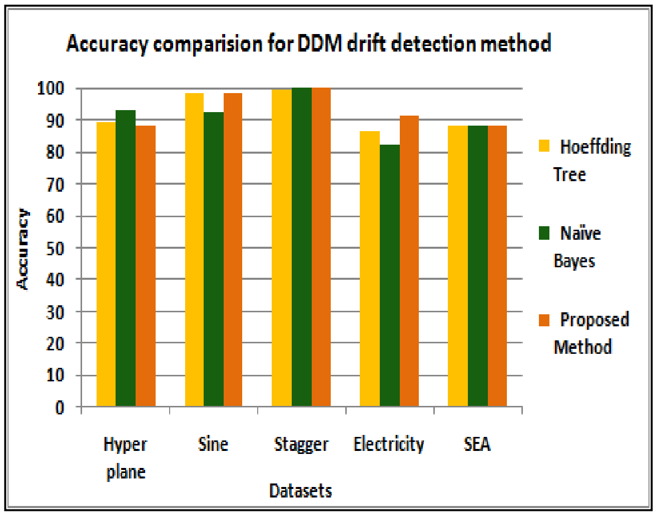

Fig. 7 Accuracy comparison of DDM Drift Detection methods for three different classifiers.

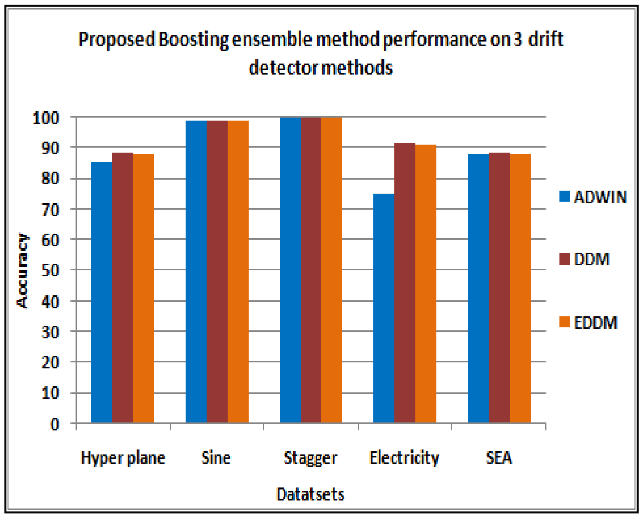

Fig. 9 Accuracy comparison of proposed Boosting ensemble method with three drift detectors.

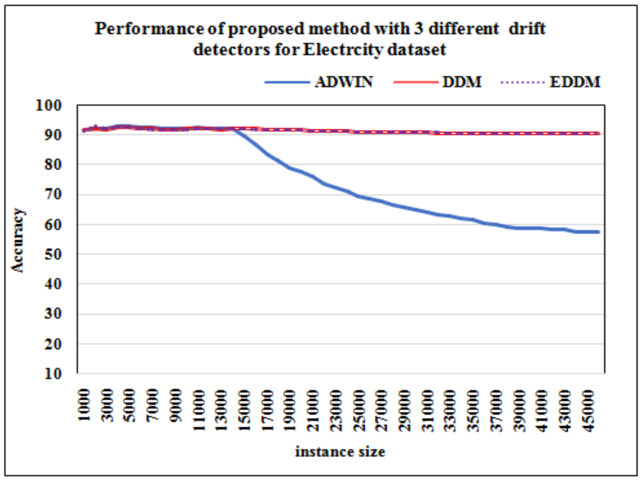

Fig. 10 Performance of proposed method with 3 different drift detectors for Electricity dataset

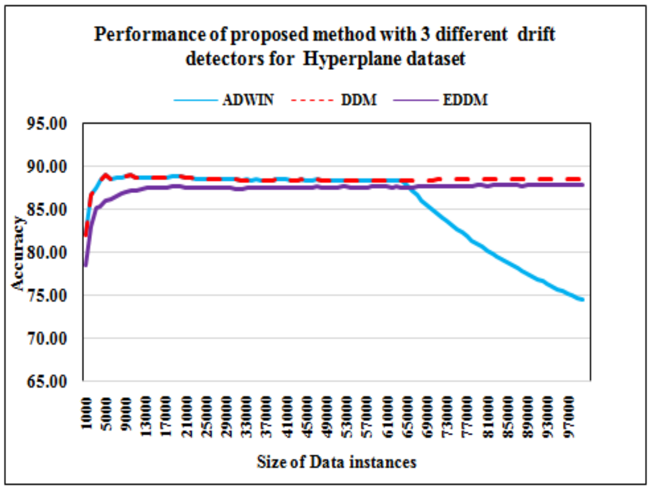

Fig. 11 Performance of proposed method with 3 different drift detectors for Hyperplane dataset

Fig. 10 and 11 demonstrate how to drift detectors help detect drift in data and how a strong ensemble construction policy aids in recovering from drift and improving classifier performance. The window-based approach cannot recover its performance in both cases of drift occurrence, whereas the error-based approach never degrades its performance during the process.

5. Conclusion

This paper presents a systematic analysis of drift detection methods for improving real-world data analysis performance. The most recent ensemble-based approach, which is most powerful in dealing with massive amounts of the data stream, is also discussed, implemented, and studied. The primary goal of this study is to demonstrate the utility of drift detectors in stream classification using real-world data sets. A large amount of data generated by IoT sensors, medical devices, and social media must be accurately analyzed in light of the large volume, change in concept, and velocity.

The results of this experiment reveal that error-based drift detection systems are extremely effective at identifying all types of drift. Whereas in the case of the window-based approach, the most important task is to determine the window size; a window that is too small will not detect progressive drift, while a window that is too large will miss out on abrupt drift. As a result, arranging a correct size or changeable size window is a solution.

This research will aid in the creation of a new ensemble using the drift detection technique. A suitable method can be employed based on the application's data drift type. This study shows that DDM is always good for all types of drift and that a small change in the error calculation span can increase its performance.

Based on a thorough examination of various data sets, it is clear that the difference between the two results - Error based and window-based methods - has demonstrated its applicability in different types of drift. As a result, these methods will be more useful in online data stream analysis. For future work, a combination of EDDM and DDM by modifying its parameters with an ensemble classifier should be tested to see if better results can be achieved.

Acknowledgements

The authors thank Bharati Vidyapeeth Deemed to be University, Pune, for its strong support.

References

[1] S. R. Nikunj oza, "Online Bagging and Boosting," in *8th Int. Workshop on Artificial Intelligence and Statistics*, pp. 105–112, 2001.

[2] I. Frías-Blanco, A. Verdecia-Cabrera, A. Ortiz-Díaz, and A. Carvalho, "Fast adaptive stacking of ensembles," *Proc. ACM Symp. Appl. Comput.*, vol. 04-08, pp. 929–934, 2016, doi: 10.1145/2851613.2851655.

[3] M. Kholghi, H. Hassanzadeh, and M. R. Keyvanpour, "Classification and evaluation of data mining techniques for data stream requirements," *3CA 2010 - 2010 Int. Symp. Comput. Commun. Control Autom.*, vol. 1, pp. 474–478, 2010, doi: 10.1109/3CA.2010.5533759.

[4] S. Muthukrishnan, "Data Streams: Algorithms and Applications," *Data Streams Algorithms Appl.*, pp. 1–39, 2005, doi: 10.1561/9781933019604.

[5] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "On demand classification of data streams," *KDD-2004 - Proc. Tenth ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 503–508, 2004, doi: 10.1145/1014052.1014110.

[6] H. M. Gomes, J. P. Barddal, A. F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Comput. Surv.*, vol. 50, no. 2, 2017, doi: 10.1145/3054925.

[7] H. Wang, W. Fan, P. S. Yu, and J. Han, "Mining concept-drifting data streams using ensemble classifiers," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 226–235, 2003, doi: 10.1145/956750.956778.

[8] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, "New ensemble methods for evolving data streams," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 139–147, 2009, doi: 10.1145/1557019.1557041.

[9] A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6321 LNAI, no. PART 1, pp. 135–150, 2010, doi: 10.1007/978-3-642-15880-3_15.

[10] D. Brzezinski and J. Stefanowski, "Combining block-based and online methods in learning ensembles from concept drifting data streams," *Inf. Sci. (Ny)*, vol. 265, pp. 50–67, 2014, doi: 10.1016/j.ins.2013.12.011.

[11] R.Samant and S. Patil, "Comparative Analysis of drift detection techniques used in ensemble classification approach," in *International Conference on Recent Challenges in Engineering Science and Technology (ICRCEST 2K21)*, pp. 201–204, 2021.

[12] L. Durga and R. Deepu, "Ensemble Learning Based Analysis Correlating Graphology to Big Five Personality Model," *International Journal of Engineering Trends Technology*, vol. 70, no. 1, pp. 240–251, 2022, doi: 10.14445/22315381/IJETT-V70I1P229.

[13] S. L. V. Papineni, S. Yarlagadda, H. Akkineni, and A. M. Reddy, "Big data analytics applying the fusion approach of multicriteria decision making with deep learning algorithms," *International Journal of Engineering Trends Technology*, vol. 69, no. 1, pp. 24–28, 2021, doi: 10.14445/22315381/IJETT-V69I1P204.

[14] A. Bifet, S. Maniu, J. Qian, G. Tian, C. He, and W. Fan, "StreamDM: Advanced Data Mining in Spark Streaming," in *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*, Nov. 2015, pp. 1608–1611. doi: 10.1109/ICDMW.2015.140.

[15] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," *Proc. 7th SIAM Int. Conf. Data Min.*, no. April, pp. 443–448, 2007, doi: 10.1137/1.9781611972771.42.

[16] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 3171, no. September, pp. 286–295, 2004, doi: 10.1007/978-3-540-28645-5_29.

[17] A. Bifet *et al.*, "Early Drift Detection Method," *4th ECML PKDD Int. Work. Knowl. Discov. from Data Streams*, vol. 6, no. August 2014, pp. 77–86, 2006.

[18] W. Nick Street and Y. S. Kim, "A streaming ensemble algorithm (SEA) for large-scale classification," *Proc. Seventh ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vol. 4, pp. 377–382, 2001, doi: 10.1145/502512.502568.

[19] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository," 2007. http://www.ics.uci.edu/~mlearn/ MLRepository.html

[20] "Datasets." https://github.com/vlosing/driftDatasets

[21] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive Online Analysis," *J. Mach. Learn. Res.*, vol. 11, pp. 1601–1604, 2010.

[22] R. C. Samant and S. H. Patil, "An Enhanced Online Boosting Ensemble Classification Technique to Deal with Data Drift,", Department of Computer Engineering, Bharati Vidyapeeth Deemed University, Pune, 2022.

[23] R. C. Samant and S. H. Patil, "A Systematic and Novel Ensemble Construction Method for Handling Data Stream Challenges," *Chen, J.IZ., Tavares, J.M.R.S., Shi, F. Third Int. Conf. Image Process. Capsul. Networks. ICIPCN 2022. Lect. Notes Networks Syst.*, vol. 514, 2022, doi: https://doi.org/10.1007/978-3-031-12413-6_20.

[24] R. S. M. De Barros, S. G. T. De Carvalho Santos, and P. M. G. Junior, "A Boosting-like Online Learning Ensemble," *Proc. Int. Jt. Conf. Neural Networks*, vol. 2016, no. 2018, pp. 1871–1878, 2016, doi: 10.1109/IJCNN.2016.7727427.