Journal of Advanced Computer Science & Technology, 13 (1) (2026) 27-37

Journal of Advanced Computer Science & Technology

Website: www.sciencepubco.com/index.php/JACST
https://doi.org/10.14419/vq3n9107
Research paper

Individual Classifiers Accuracy Based Online Boosting Algorithm

Nagaraj Honnikoll *, Ishwar Baidari

Department of Computer Science, Karnatak University, Dharwad 580001, Karnataka, India
**Corresponding author E-mail: ishwarbaidari@kud.ac.in*

Received: December 29, 2025, Accepted: March 6, 2026, Published: May 28, 2026

Abstract

Boosting is a method for transforming weak learners into a powerful ensemble. To model this technique, classifiers are trained on several datasets. This article proposes a boosting-like online learning algorithm, a modified form of Oza and Russell's online boosting, aimed at maintaining high and consistent accuracy in environments where concepts change continuously. We use each classifier's accuracy to transform weak learners into strong learners. Our algorithm was tested against other state-of-the-art methods using several real and artificial datasets. Accuracy improved drastically in most tested situations.

Keywords: *Concept Drift; Hellinger Distance; Data Stream; Online Learning; Hoeffding Bound.*

1. Introduction

Data streams in an online environment are continuous flows of data that arrive in large quantities and very high speed. Efficient algorithms are needed to handle such data because it is difficult to store and process it as it arrives.
When the flow of data is continuous and rapid, the data distribution is usually highly dynamic. In other words, changes may occur over time in the data distribution; this condition is known as concept drift [1] [2]. Learning from a continuous flow of data in the presence of drift is known as online learning [3] [4]. Many online learning applications exist, such as wireless sensor data monitoring, industrial process monitoring systems, and more. Thus, the learning mechanism adopted to handle drift must be updated to adjust to new instances.
Boosting [5] [6] is a popular and generalized approach for enhancing the accuracy of the base model (``weak learners''). The main strategy in the boosting technique is to train base models using several different distributions on the training data and join all base models in an ensemble.
There are a number of approaches in the literature for identifying concept drift. Some approaches adapt the internal configuration of a base learner to adjust it to new instances. Other approaches detect changes when concept drift is noticed, discard the existing classifier, and train a new classifier. Some approaches adopt waiting policies and ensemble techniques. There are also methods that try to enhance accuracy by identifying abrupt changes in the distribution.
In settings where concept changes occur frequently, base learners must continuously adjust to the current (new) distribution. However, if the transition period from the old concept to the new concept is long, wrong predictions are more likely.
The proposed method is based on Oza and Russell's online bagging and boosting [7]. We introduce a weighting method based on accuracy which uses the individual classifier's accuracy to compute the Poisson distribution (λ) value. Cumulative accuracy of the correctly classified and incorrectly classified instances of all classifiers will be used to calculate the weight for the current classifier. When a classifier misclassifies a training instance, the Poisson distribution (λ) parameter related to that instance will be increased and passed to the next classifier. The formulated equations to calculate λ value is shown in the equations 1 and 2. On the other hand, for correctly classified instance, the λ value will be decreased using the same equations 1 and 2. We modified the traditional voting technique for final ensemble prediction. The newly defined weighting strategy, which is based on classifiers accuracy, along with other modifications improved overall accuracy of the ensemble.
The rest of the paper is organized as follows, the related work is presented in section 2. The proposed method is explained in section 3. Section 4 describes the experimental set up and data sets used. The experimental results analysis is presented in section 4.3 and conclusion is presented in section 5.

2. Related Work

Several techniques have been defined by various authors to learn data streams that have drifts. For example, some methodologies proposed initially include OzaBag [7], OzaBoost [7] and OSBoost [8]. Recently, many techniques have been introduced based on ensemble classifiers.

Bagging and boosting [6] are two original methods, in both the techniques, a set of classifiers are trained on training data, combining the output of each classifier to achieve better accuracy. Online bagging and boosting algorithms are designed to be suitable for data stream environments. Both the algorithms make use of Poisson distribution parameter to imitate the character of their off-line algorithms.
Two bagging algorithms ADWINBAG [9] and Leveraging [10], use Poisson distribution parameter originally proposed by Oza and Russell [7]. To detect the drifts, adaptive window technique has been introduced in ADWIN [11]. Further, in Leverage Bagging, the authors modified the original idea and made two changes: 1) Increasing the λ value, so classifier will be trained on the same instance more times, and 2) By increasing diversity, by changing the usual way classifiers predict instances in order to enhance the diversity and reduce the correlation.
The DWM [12] is a further improvement of weighted majority algorithm to detect the drifts by implementing a weighted ensemble classifier. Classifiers are added and deleted based on ensemble accuracy. For example, if the ensemble misclassifies, a classifier is added. A classifier weight will be decreases if it misclassifies. If any classifier continues the misclassify and carries a low weight, it is deleted from the ensemble.
The idea accomplished in DDD [13] is to find best weighted majority of ensembles with high and low diversity before and after drift is identified by the EDDM [14].
The AUE2 [15] introduces the idea of adding and removing, performing and underperforming classifiers respectively for every n instances. The weight is decided based on individual classifier accuracy. The strategy is to make the classifier more effective by updating it, so the method will be more productive to changes in the concepts.
ADOB [16], BOLE [] [17]is based on OZABoost [7]. The main aim is to speed up the recovery of the classifier after the drifts. ADOB sorts the classifiers based on accuracy. This strategy impacts the way diversity is distributed among the classifiers, strengthening the accuracy of the ensemble of the drifts. BOLE further, improved the accuracy of ADOB by allowing more number of classifiers to vote for the final prediction. By this modification BOLE improved ensemble accuracy.
Boosting Ensembles OABM1 and OABM2 [18] were continues the features of the traditional batch version. The main aim of the OABM1 is to determine the number of times, the same instance is used to train the classifier which indicates the amount of training, Poisson distribution (Oza and Russel) used to stimulate the behaviour of ADABoost.M1. OABM2 is based on ADABoost.M2 and was modified to streaming environments. Accordingly, it is made ready to handle problems that have more than two classes.
Having presented the algorithms with their approaches and drawbacks, in the next section we present our approach in detail.

Table 1: Used Mathematical Symbols and Their Descriptions

Symbols	Descriptions
d	An instance of data.
M	Ensemble Size.
m	Current classifier in the ensemble M.
ε_m	Error rate for classifier m.
β_m	Weight given to the current classifier m.
λ	The parameter of the Poisson distribution.
λ_m^{sc}	Sum of instances perfectly classified by the classifier m.
λ_m^{sw}	Sum of instances wrongly classified by the classifier m.
N	The number of instances seen in the big data stream so far.
c	The number of classes for the datasets given.
s_c^m	The probabilities of each class for the classifier m.
p_m	Classifier error rate of m.

3. Proposed Work

The proposed method modifies the online boosting technique originally proposed by Oza and Russell. The proposed algorithm distributes data instances systematically among classifiers, making the classifiers more appropriate for situations in which concept drift occurs frequently. We also modify the way λ (Poission parameter) is computed. This strategy influences classifier performance substantially. Table 1 presents the notations and their descriptions used in the rest of the paper.
As the classifiers have a high level of similarity in the beginning due to the reduced number of instances, classifier accuracy is used to control classifier behaviour. As more instances arrive from the same distribution, the classifiers become more diverse.
In the case of a new distribution, the algorithm uses the accuracy of each classifier to reduce the initial errors. The modified Poisson distribution parameter (λ) is computed using Eqs. 1 and 2.

$$\beta_m = \frac{p_m}{1-p_m} \tag{1}$$

$$\lambda_d = \lambda_d \times \beta_m \times \frac{\lambda_m^{sc}+\lambda_m^{sw}}{2\times\lambda_m^{sw}} \tag{2}$$

Eqs. (1) presents the normalized error for the classifier m, if the classifier m correctly classifies an instance than this β_m is utilized by the eqs. (2) which decrements the value of λ_d. If the classifier m incorrectly classifies an instance than this β_m is not utilized, which increases the value of λ_d.

Algorithm 1 CEROB (h_m, OnlineBase, d)	
•	Set the example's ``weight" $\lambda_d \leftarrow 1, \alpha \leftarrow 1$
•	For each base model h_m, $(m \in 1,2,3,\dots,M)$ in the ensemble,
1.	Set k according to $\text{Poisson}(\lambda_d)$.
2.	Do k times, $h_m \leftarrow \text{OnlineBase}(h_m, d)$
3.	If $h_m(d)$ is the correct label, then $\text{pred} \leftarrow 0$ else $\text{pred} \leftarrow 1$
4.	Calculate the classifier $h_m(d)$ error rate, $p_m \leftarrow p_m + \left(\frac{\text{pred}-p_m}{N}\right)$
5.	Accuracy of $h_m(d)$, $acc_m \leftarrow (1-p_m)$
6.	If $h_m(d)$ is the correct label,
○	then

1.	$\lambda_m^{sc} \leftarrow \lambda_m^{sc} + acc_m$
2.	$\alpha \leftarrow 1$
○	else
1.	$\lambda_m^{sw} \leftarrow \lambda_m^{sw} + acc_m$
2.	$\alpha \leftarrow 0$
7.	$\beta_m = \frac{p_m}{1-p_m}$
8.	$\lambda_d = \lambda_d \times \beta_m^{\alpha} \times \frac{\lambda_m^{sc}+\lambda_m^{sw}}{2\times\lambda_m^{sw}}$

Looking at lines 6--8 of Algorithm 1, λ_m^{sw} is the sum of the accuracy values for instances that were classified incorrectly by the experts at stage m, and λ_m^{sc} is the sum for correctly classified instances. In the next stage, the Poisson distribution parameter (λ) for the current instance is increased or decreased and passed to the next classifier.
Instead of using all M classifiers to vote for the final ensemble prediction, we modify the approach based on accuracy, as shown in Eqs. (3) and (4). Lines 1 and 2 in Algorithm 2 show the modified method.

$$w_m = \log\left(\frac{1-p_m}{p_m}\right) \tag{3}$$

$$IN_c = \frac{2^{s_c^m \times w_m}}{1+2^{s_c^m \times w_m}} \tag{4}$$

Algorithm 2 Computation for classifying new instance	
•	For each base model $h_m, (m \in 1,2,3,\ldots,M)$ in the ensemble,
1.	Calculate Weight of the $h_m, w_m = \log\left(\frac{1-p_m}{p_m}\right)$
2.	For each class $c \in C$ $ for the Instance d,
○	$IN_c = \frac{2^{s_c^m \times w_m}}{1+2^{s_c^m \times w_m}}$
•	Return $h(d) = \text{argmax}_{c \in C} \sum_{m:h_m(d)} = y^{IN_c}$

Therefore, introducing a modified weighting strategy that computes λ using accuracy, and implementing Eqs. (3) and (4) for the final ensemble prediction, enhances the overall accuracy of the ensemble.

4. Experimental Configuration and Results

This section provides detailed information about the experiments used to assess our algorithm against other state-of-the-art methods, namely OzaBag, OzaBoost, OSBoost, ADOB, BOLE, OABM1, OABM2, KUE, and AUE (known as AUE2). The selected methods are:

4.1. Drift configurations in the artificial datasets

While generating the artificial datasets, we introduced abrupt drifts and, in some cases, noise in order to observe the behaviour of the methods under these conditions.
We used four variants of the Mixed dataset, each containing abrupt drifts. In the first variant, drifts occurred at instances 10,000, 20,000, 30,000, and 40,000. In the second variant, abrupt drifts were inserted at instances 20,000, 40,000, 60,000, and 80,000. In the remaining two variants, abrupt drifts were inserted at instances 7,000, 14,000, 21,000, and 28,000, and at 100,000, 200,000, 300,000, and 400,000. In all four variants, 10% noise was inserted each time a drift occurred. We also examined abrupt drifts in the RBF, Agrawal, Sine, and LED datasets, using the same drift positions as for the Mixed dataset. The characteristics of the generated datasets are presented in Table 2. In addition, the real-world datasets used for testing are listed in Table 3.

Table 2: Characteristics of Generated Artificial Datasets

Dataset	#Instances	#Attributes	#Class	#Drifts	Drift Type
MIXED	50K, 100K, 300K, 500K	4	2	4	Abrupt
SINE	50K, 100K, 300K, 500K	4	2	4	Abrupt
LED	50K, 100K, 300K, 500K	4	2	4	Abrupt
RBF	50K, 100K, 300K, 500K	4	2	4	Abrupt
AGRAWAL	50K, 100K, 300K, 500K	10	8	4	Abrupt

Table 3: Real-World Dataset Characteristics

Dataset	#Instances	#Attributes	#Class	#Drifts	Drift Type
Pokerhand [9]	829,201	10	10	Unknown	Unknown
Covertype [19]	581,012	53	7	Unknown	Unknown
Electricity [3]	45,312	9	2	Unknown	Unknown
Airlines [20]	539,383	24	4	Unknown	Unknown
Connect-4 [21]	67,557	42	3	Unknown	Unknown
Eeg-Eye-State [21]	2,310	19	7	Unknown	Unknown
Shuttle [22]	58,000	9	7	Unknown	Unknown
Gas Sensor Array Drift (Gas Sensor) [23]	13,910	129	6	Unknown	Unknown

4.2. Ensemble methods configuration

To carry out a proper comparison, the usual parameters among the methods were all set likewise. The classifier used was Hoeffding Tree, ten classifiers were involved in the ensemble.
To observe each method's performance on default values, all methods were executed with ten different configurations for each dataset to consider individual parameters. In exceptional cases, we set a different parameter.

4.3. Accuracy analysis

Table 4 shows the accuracies secured for each technique for both artificial and real-world datasets. Comparative results are presented; best results are highlighted in bold. We computed the average rank for each method with different datasets.
In the Mixed dataset with four drifts, our method performed the best, followed by ADOB, BOLE, OABM1, OABM2, OzaBoost, AUE and OSBoost, KUE and OzaBag, with OzaBag having the worst results.
In the Sine dataset with four drifts, our method again had the best accuracy, closely followed by OABM, BOLE, ADOB, AUE, KUE, OSBoost and OzaBag, with OzaBag having the worst results.
In the Agrawal dataset with four drifts, our method achieved the second-best accuracy, closely following AUE OABM1, OzaBoost, and remaining methods had the worst results compared to AUE and our method.
After detailed examination of RBF dataset with four drifts, our method achieved the third best accuracy close to the top two performing methods AUE and KUE. AUE performed the best on most datasets, closely followed by KUE. OABM1 had the worst performance.
Similarly, for the LED dataset with four drifts, our method performance was second best, nearest to AUE followed by BOLE and ADOB. Following them are OzaBoost, OABM1, OABM2, KUE, OSBoost and OzaBag. OzaBag and OSBoost secured the worst results.
On the real datasets, CEROB was the method with outstanding performance followed by BOLE, AUE, ADOB, OABM1, OABM2, Oza-Boost, KUE, OSBoost and OzaBag respectively.
The results are shown in the figures 2-5.
Complementing the accuracy analysis, a statistical test based on the non-parametric Friedman test [24] was used. In this test, the null hypothesis states that all methods are statistically equal when the hypothesis is rejected, the test indicates that there is a statistical difference in any of the methods, but it does not indicate which methods differ. For this task, the use of a Posthoc-test is required. In our case, we used the Bonfornni-Dunn to compare CEROB with the other methods.
Initially, we compared the accuracies of the methods using 95% confidence interval using Hoffeding Tree as the classifier. Therefore, with F_F, the null hypothesis is rejected. Then, proceeding to the Posthoc-test, the critical difference (CD) was found to be 2.41. So, we can say that CEROB is statistically superior in accuracy to OzaBoost, ADOB, BOLE, OABM1, OABM2, KUE, AUE, OSBoost and OzaBag. Figure 1 shows the test conducted result.

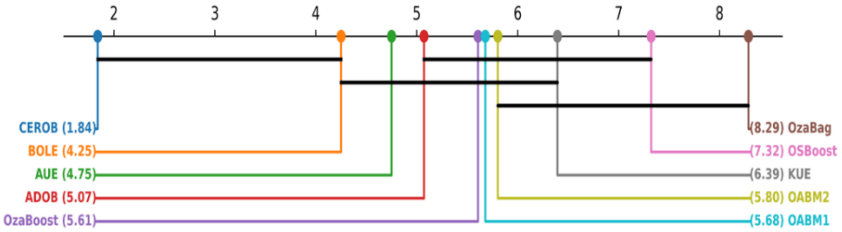

Fig. 1: Accuracies Compared for the Methods Using 95% Confidence Interval Using Posthoc-Test.

The memory usage in megabytes and the time usage in seconds is presented in Tables 5 and 6. The best results are shown in bold, and the worst results are shown in italic. In most cases, OzaBag shows the lowest memory and time usage due to its evaluation strategy, which aligns with its poorer accuracy results. The worst result is shown by OABM2 because it treats the weights differently and therefore demands more time and memory for computation. Our method shows memory and time usage similar to AUE in most cases. This indicates that our method's memory and time usage are competitive and well suited for streaming environments.

Table 4: Average Accuracies Using A 95% Confidence Interval Using Hoffeding Tree

TYPE-SIZE	DATASET	OzaBag	OzaBoost	OSBoost	ADOB	BOLE
	Agrawal	75.03±2.60	78.23±2.46	77.42±2.45	75.71±2.68	75.74±2.67
	SINE	89.65±1.85	93.72±2.75	90.57±1.97	94.19±2.82	94.19±2.84
Abrupt-50K	LED	67.44±2.48	67.34±2.35	68.14±2.35	68.28±1.72	68.36±1.73
	MIXED	89.62±5.14	94.04±1.38	89.95±4.68	94.50±1.46	94.50±1.46
	RBF	48.64±3.58	47.57±4.77	49.99±3.05	50.65±3.87	52.65±2.01
	Agrawal	74.95±3.12	79.06±2.15	77.79±2.09	77.22±3.24	77.23±3.23
	SINE	90.28±1.58	95.35±2.55	90.98±1.69	95.79±2.55	95.79±2.56
Abrupt-100K	LED	68.16±2.47	68.23±2.39	68.47±2.46	69.09±1.79	69.14±1.79
	MIXED	91.48±3.31	95.89±1.58	91.03±4.00	96.07±1.56	96.07±1.56
	RBF	54.12±5.01	54.95±3.67	54.03±4.36	57.69±4.32	57.98±4.26
	Agrawal	82.00±3.35	83.18±2.94	82.07±2.88	80.40±4.02	80.41±4.01
	SINE	93.32±1.88	97.35±2.02	93.59±1.83	97.62±1.94	97.62±1.94
Abrupt-300K	LED	69.96±2.02	69.84±1.93	70.88±1.60	71.46±1.12	71.48±1.11
	MIXED	88.16±3.94	97.28±1.04	88.34±3.65	97.37±1.23	97.37±1.23
	RBF	64.16±5.51	66.10±6.60	64.66±5.34	68.53±6.90	68.99±6.99
	Agrawal	82.39±4.00	83.10±2.49	83.30±3.19	79.65±3.26	79.65±3.25
	SINE	94.33±1.75	97.91±1.70	93.98±1.56	98.17±1.63	98.17±1.64
Abrupt-500K	LED	69.78±2.07	71.25±1.25	70.73±1.70	71.93±1.04	71.95±1.03
	MIXED	95.07±1.77	98.34±1.32	94.87±2.00	98.44±1.25	98.44±1.25
	RBF	66.71±5.08	72.20±6.81	69.41±5.42	72.99±6.92	74.67±7.33
	COVERTYPE	83.62±4.26	88.59±4.80	83.89±4.15	88.88±4.60	88.88±4.59
	EEG-EYE-STATE	77.88±5.44	97.19±0.36	79.95±3.94	96.70±0.72	96.70±0.72
	SHUTTLE	96.31±1.60	99.10±0.70	95.72±1.70	97.66±1.12	97.66±1.12
Real World	CONNECT-4	76.77±2.35	78.79±1.96	77.59±2.16	78.74±2.44	78.76±2.42
	ELECTRICITY	84.96±2.08	89.26±1.99	85.99±1.74	89.23±1.92	89.28±1.97
	AIRLINES	66.10±1.23	63.42±0.94	66.98±1.23	63.55±0.81	63.55±0.81
	POKERHAND	79.88±2.38	83.06±3.37	78.40±2.01	82.75±2.09	82.77±2.06
	GAS SENSOR	59.68±3.22	60.08±3.66	60.46±3.63	59.13±5.00	62.48±3.58
RANK	-	8.29	5.61	7.32	5.07	4.25

-- table continued

Table 5: Average Accuracies Using A 95% Confidence Interval Using Hoffeding Tree (Table Continued from Previous).

TYPE-SIZE	DATASET	OABM1	OABM2	KUE	AUE	CEROB
	Agrawal	78.91±2.86	73.30±1.96	74.12±6.96	81.21±6.17	79.13±2.62
	SINE	94.48±2.28	93.14±3.04	89.00±6.92	91.26±4.62	94.99±2.67
Abrupt-50K	LED	66.85±2.65	68.79±1.68	65.25±9.07	68.60±4.26	69.25±1.49
	MIXED	94.21±1.31	93.69±1.22	89.42±6.39	91.21±3.47	94.96±2.12
	RBF	26.52±3.08	52.90±2.71	60.79±11.44	63.36±9.12	52.95±3.32
	Agrawal	81.12±3.03	75.20±2.17	80.74±5.93	84.60±5.79	81.41±2.71
	SINE	95.93±2.17	95.03±2.84	92.46±5.55	93.42±3.97	96.37±2.35
Abrupt-100K	LED	67.89±2.54	69.51±1.74	69.02±7.05	71.13±3.45	70.69±1.31
	MIXED	95.92±1.63	95.64±1.35	91.02±4.83	93.59±3.01	96.44±2.00
	RBF	26.42±2.48	57.81±6.37	68.29±10.58	71.24±9.80	58.63±5.37
	Agrawal	85.17±3.18	76.85±4.00	84.56±4.95	88.66±4.82	86.62±3.22
	SINE	97.66±1.71	97.39±2.23	96.02±3.90	96.40±3.06	97.87±1.73
Abrupt-300K	LED	70.86±1.51	71.77±1.00	71.73±4.36	72.93±2.21	72.28±0.88
	MIXED	97.51±1.13	96.95±1.03	93.33±3.22	95.17±2.00	97.64±1.42
	RBF	47.60±11.70	68.69±7.90	75.53±13.37	80.32±8.43	75.25±8.75
	Agrawal	85.72±2.77	79.89±2.90	87.83±3.64	90.39±4.38	87.44±2.80
	SINE	98.17±1.46	98.04±1.88	96.69±3.28	97.26±2.57	98.31±1.44
Abrupt-500K	LED	71.07±1.37	72.27±0.91	72.57±3.51	73.38±1.77	72.22±1.29
	MIXED	98.31±1.33	98.31±1.29	96.35±3.16	97.34±2.04	98.58±1.34
	RBF	60.02±13.69	74.28±8.10	79.95±7.57	83.24±7.40	80.31±8.58
	COVERTYPE	88.18±5.47	88.63±4.27	83.50±9.40	85.44±6.80	90.88±4.05
	EEG-EYE-STATE	96.58±1.32	97.53±0.91	52.22±6.31	48.73±5.49	97.76±0.68
	SHUTTLE	99.11±0.76	97.07±1.26	97.39±3.03	96.52±2.53	99.11±1.00
Real World	CONNECT-4	76.69±2.51	78.24±2.18	60.28±16.36	69.97±4.27	79.85±2.07
	ELECTRICITY	89.53±1.95	52.67±11.31	75.64±4.87	76.41±3.08	90.59±1.90
	AIRLINES	63.23±0.92	61.21±3.81	66.32±1.23	66.96±1.11	63.44±1.07
	POKERHAND	82.16±2.42	69.72±7.93	84.21±7.93	66.74±1.12	85.92±3.27
	GAS SENSOR	55.97±4.97	61.07±3.77	33.15±14.04	49.15±9.22	65.16±4.00
RANK	-	5.68	5.80	6.39	4.75	1.84

Table 6: Memory Usage in Terms of MEGABYTES (MB).

TYPE-SIZE	DATASET	OzaBag	OzaBoost	OSBoost	ADOB	BOLE
	Agrawal	0.0140	0.0127	0.0137	0.0250	0.0142
	Sine	0.0088	0.0086	0.0100	0.0088	0.0090
Abupt-50K	LED	0.0317	0.0364	0.0501	0.0537	0.0447
	Mixed	0.0064	0.0072	0.0083	0.0091	0.0070
	RBF	0.0359	0.0390	0.0529	0.0422	0.0439
	Agrawal	0.0257	0.0275	0.0299	0.0406	0.0276
	Sine	0.0153	0.0178	0.0190	0.0185	0.0179
Abupt-100K	LED	0.0621	0.0733	0.1054	0.0994	0.0860
	Mixed	0.0111	0.0141	0.0151	0.0158	0.0157
	RBF	0.0634	0.0727	0.1014	0.0812	0.0823
	Agrawal	0.0915	0.0949	0.1063	0.1189	0.1022
	Sine	0.0457	0.0601	0.0619	0.0600	0.0594
Abupt-300K	LED	0.1894	0.2199	0.3072	0.2812	0.2677
	Mixed	0.0391	0.0530	0.0494	0.0497	0.0511
	RBF	0.1959	0.2620	0.2879	0.2668	0.2751
	Agrawal	0.1561	0.1609	0.1872	0.1933	0.1803
	Sine	0.0790	0.1088	0.1048	0.1093	0.1091
Abupt-500K	LED	0.2918	0.3701	0.4631	0.4269	0.4119
	Mixed	0.0577	0.0944	0.0760	0.0888	0.0879
	RBF	0.3373	0.4361	0.4867	0.4348	0.4569
	COVERTYPE	0.5723	0.7923	0.8594	0.6918	0.7914
	EEG-EYE-STATE	0.0059	0.0065	0.0078	0.0064	0.0064
	SHUTTLE	0.0160	0.0164	0.0247	0.0191	0.0205
Real World	CONNECT-4	0.0363	0.0442	0.0504	0.0434	0.0452
	ELECTRICITY	0.0097	0.0113	0.0133	0.0123	0.0116
	AIRLINES	0.9822	1.1111	0.8836	1.0849	1.1712
	POKERHAND	0.2681	0.3692	0.4209	0.3437	0.3805
	GAS SENSOR	0.0902	0.0945	0.1474	0.1108	0.1222

-- table continued

Table 7: Memory Usage in Terms of MEGABYTES (MB) (Table Continued from Previous)

TYPE-SIZE	DATASET	OABM1	OABM2	KUE	AUE	CEROB
	Agrawal	0.0131	0.0572	0.0230	0.0223	0.0190
	Sine	0.0093	0.0368	0.0131	0.0136	0.0139
Abupt-50K	LED	0.0288	0.5848	0.0454	0.0626	0.0641
	Mixed	0.0070	0.0304	0.0111	0.0074	0.0125
	RBF	0.0260	0.5691	0.0509	0.0540	0.0741
	Agrawal	0.0293	0.1090	0.0449	0.0474	0.0380
	Sine	0.0172	0.0688	0.0290	0.0265	0.0285
Abupt-100K	LED	0.0596	1.1710	0.0947	0.1299	0.1382
	Mixed	0.0135	0.0532	0.0210	0.0181	0.0255
	RBF	0.0534	1.0038	0.1042	0.1127	0.1250
	Agrawal	0.1044	0.3100	0.1638	0.1481	0.1407
Abupt-300K	Sine	0.0557	0.1861	0.0755	0.0807	0.0955
	LED	0.2095	3.4767	0.2960	0.4034	0.4126

	Mixed	0.0534	0.1690	0.0622	0.0582	0.1002
	RBF	0.1995	2.9927	0.3105	0.3640	0.4057
	Agrawal	0.1722	0.5563	0.2730	0.2465	0.2634
	Sine	0.0958	0.3198	0.1264	0.1352	0.1737
Abupt-500K	LED	0.3133	5.6578	0.4525	0.6962	0.6787
	Mixed	0.0810	0.2493	0.1024	0.1010	0.1455
	RBF	0.3555	4.8518	0.5336	0.6172	0.6718
	COVERTYPE	0.7776	3.1687	1.0108	1.2203	1.0318
	EEG-EYE-STATE	0.0136	0.0195	0.0084	0.0066	0.0127
	SHUTTLE	0.0177	0.2086	0.0218	0.0277	0.0268
Real World	CONNECT-4	0.0369	0.1741	0.0562	0.0635	0.0741
	ELECTRICITY	0.0132	0.0469	0.0176	0.0195	0.0189
	AIRLINES	1.1307	0.4774	1.8598	2.0522	1.0594
	POKERHAND	0.3613	5.2346	0.4202	0.3979	0.4678
	GAS SENSOR	0.0831	0.2563	0.1188	0.1264	0.1595

Table 8: Time Usage in Terms of SECONDS (s)

TYPE-SIZE	DATASET	OzaBag	OzaBoost	OSBoost	ADOB	BOLE
	Agrawal	1.0740	0.8908	1.0314	1.4017	0.9426
	Sine	0.6405	0.6673	0.7300	0.6649	0.6520
Abupt-50K	LED	2.0738	2.4224	3.2580	3.2524	2.8960
	Mixed	0.5160	0.5578	0.6003	0.6262	0.5283
	RBF	2.6102	2.7506	3.7150	2.8345	3.0207
	Agrawal	2.0471	2.2125	2.2111	2.7817	2.1873
	Sine	1.1715	1.3976	1.4348	1.4078	1.3277
Abupt-100K	LED	4.1443	4.9604	6.7939	6.1944	5.5093
	Mixed	0.8523	1.1645	1.1499	1.2249	1.2283
	RBF	4.7321	5.5185	7.4838	6.0823	6.0677
	Agrawal	7.1940	7.4736	8.2639	8.9225	7.9932
	Sine	3.5129	4.9223	4.7478	4.9451	4.9234
Abupt-300K	LED	12.6184	14.0185	20.0024	18.7564	16.9969
	Mixed	3.1549	4.4269	4.0143	4.1836	4.1958
	RBF	14.1932	19.4612	20.3318	19.4448	19.7057
	Agrawal	12.5005	12.8283	14.5522	14.4225	14.2815
	Sine	6.1992	9.1610	8.1777	8.9943	8.9237
Abupt-500K	LED	18.8360	23.6500	29.0352	26.2985	26.4118
	Mixed	4.6565	8.3841	6.0532	7.7984	7.6606
	RBF	24.6383	32.3364	35.1370	31.0381	32.7926
	COVERTYPE	36.3547	50.0353	54.6661	44.6106	49.1773
	EEG-EYE-STATE	0.4260	0.4573	0.5211	0.4320	0.4354
	SHUTTLE	1.1055	1.0732	1.7142	1.2354	1.3173
Real World	CONNECT-4	2.1564	2.8132	3.0834	2.7765	2.8321
	ELECTRICITY	0.7198	0.8307	0.9888	0.9439	0.8437
	AIRLINES	75.2503	82.5471	74.8600	82.3911	85.6947
	POKERHAND	20.2293	27.1698	31.0183	25.0912	28.0751
	GAS SENSOR	4.8397	5.2241	7.8874	6.0763	6.3412

-- table continued

Table 9: Time Usage in Terms of SECONDS (S) (Table Continued from Previous).

TYPE-SIZE	DATASET	OABM1	OABM2	KUE	AUE	CEROB
	Agrawal	0.9370	2.3767	1.6121	1.7517	1.4237
	Sine	0.7023	1.5363	0.9434	1.0743	1.0526
Abupt-50K	LED	1.8649	14.4354	3.0435	4.2808	4.3029
	Mixed	0.5672	1.3333	0.7917	0.6354	0.9914
	RBF	1.6849	14.1162	3.6118	4.0070	5.1124
	Agrawal	2.3803	4.8992	3.4070	3.5805	2.9936
	Sine	1.2230	2.9044	2.0339	2.0854	2.1760
Abupt-100K	LED	4.1075	29.0459	6.1519	8.8134	8.9975
	Mixed	1.1747	2.4733	1.5453	1.4135	2.0826
	RBF	3.7164	25.6148	7.4494	7.8576	9.1806
	Agrawal	8.1589	13.5732	12.5461	10.6345	11.7827
	Sine	4.5201	8.1557	5.7096	5.9277	7.9620
Abupt-300K	LED	14.2145	85.4279	18.7729	26.7732	27.3787
	Mixed	4.7926	8.1747	4.6055	4.2865	9.0981
	RBF	16.2596	75.7760	22.2196	25.8835	29.3457
	Agrawal	13.8581	25.4475	20.3537	17.8071	22.3856
	Sine	7.8629	14.1428	9.1776	9.9938	14.8822
Abupt-500K	LED	19.3391	141.3838	29.0777	45.7816	45.2071
	Mixed	7.4705	12.3170	7.5940	7.5139	12.7808
	RBF	28.6687	124.3403	36.0892	40.9915	45.2702
	COVERTYPE	49.5299	91.6175	62.6195	76.4871	59.1569
	EEG-EYE-STATE	0.9193	0.8276	0.5315	0.5564	0.8409
	SHUTTLE	1.1190	5.7074	1.4539	1.9343	1.7594
Real World	CONNECT-4	2.5479	6.3847	3.3711	3.9477	4.4359
	ELECTRICITY	0.9911	2.0543	1.3143	1.4909	1.3893
	AIRLINES	83.8543	20.6562	134.3802	138.4213	70.9038
	POKERHAND	26.7677	134.5122	31.4598	29.4070	31.8971
	GAS SENSOR	4.0865	6.9235	5.9960	8.3249	9.0523

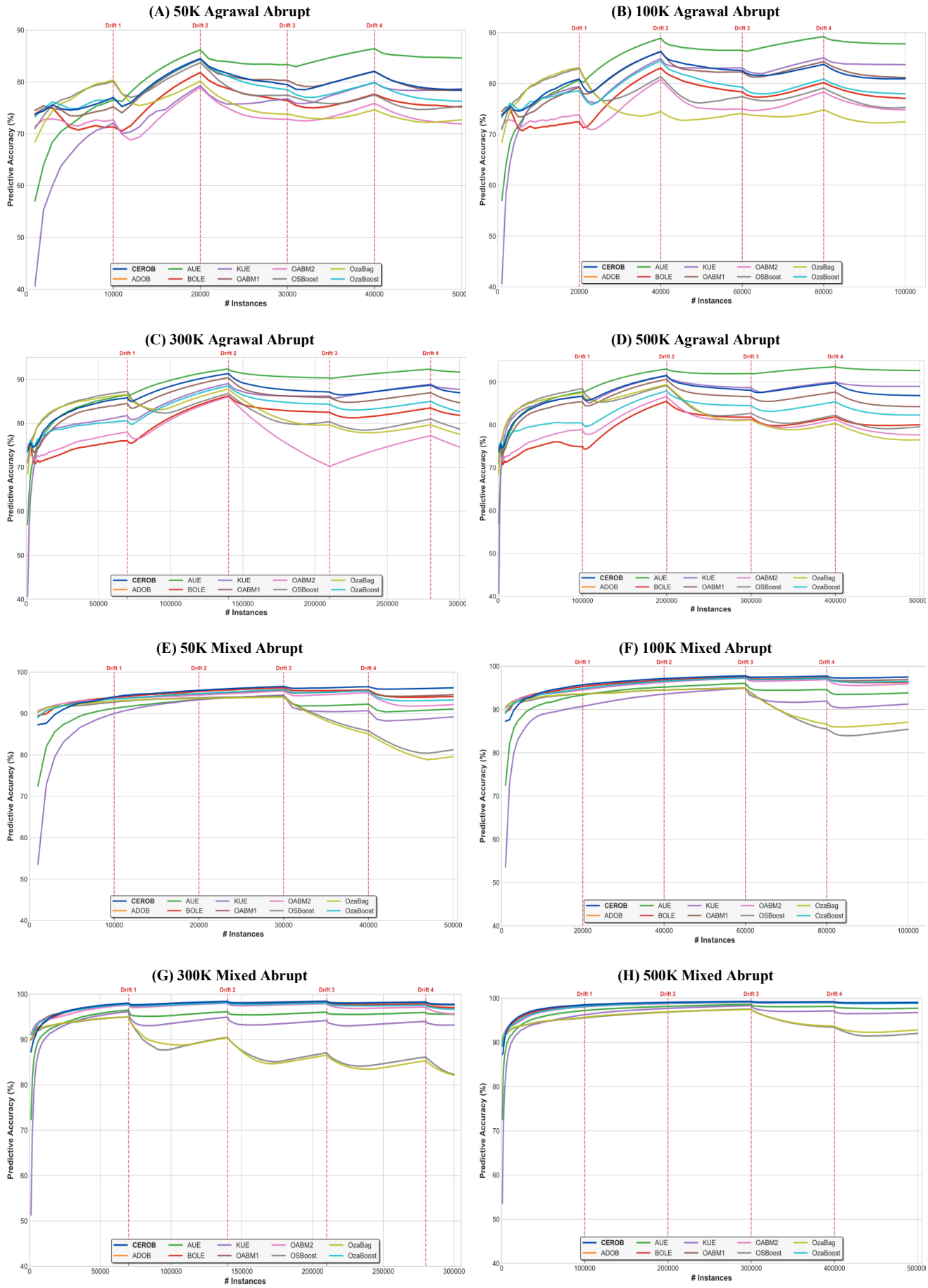

Fig. 2: Accuracy Results in Case of Agrawal and Mixed Dataset.

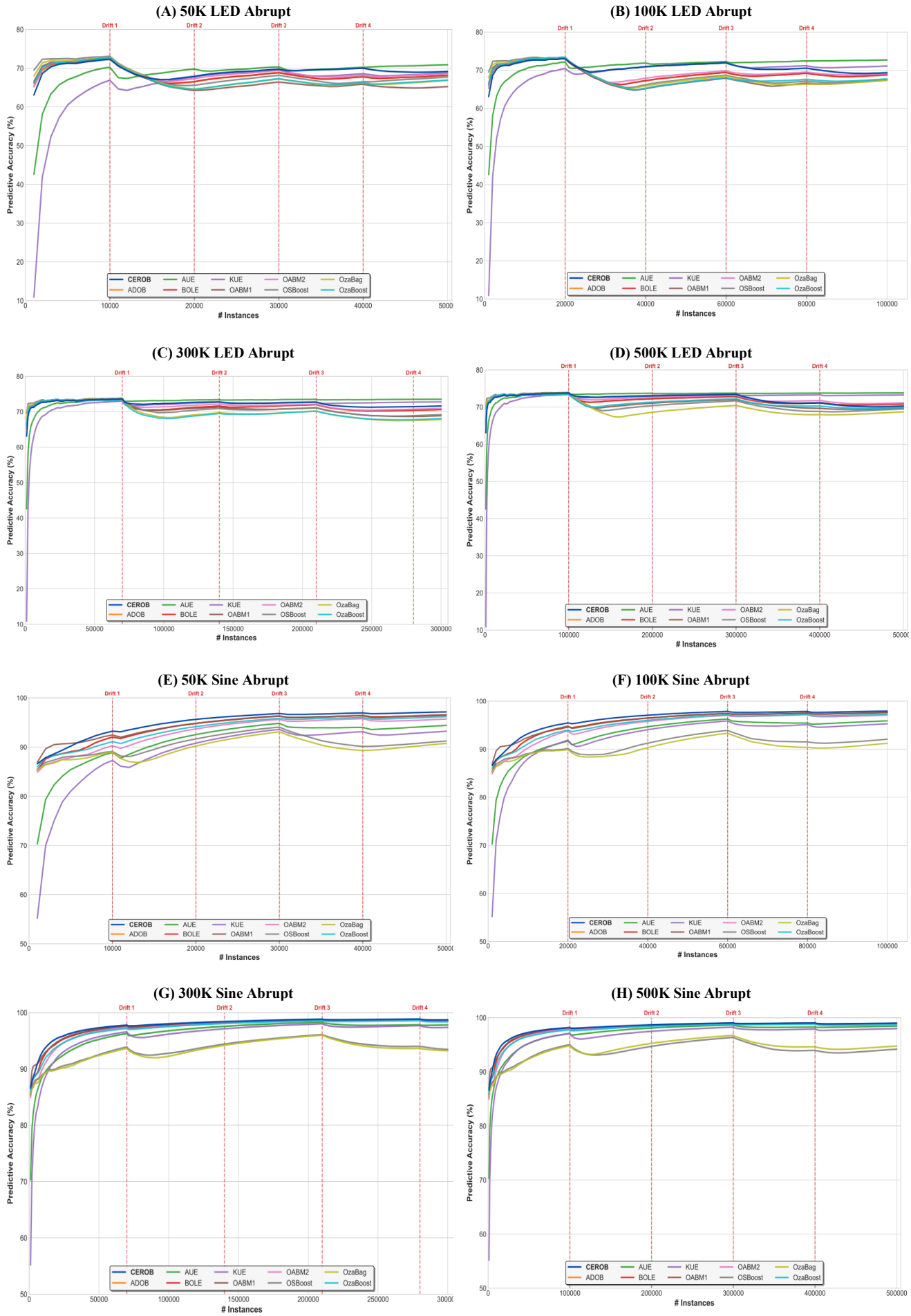

Fig. 3: Accuracy Results in Case of LED and Sine Dataset.

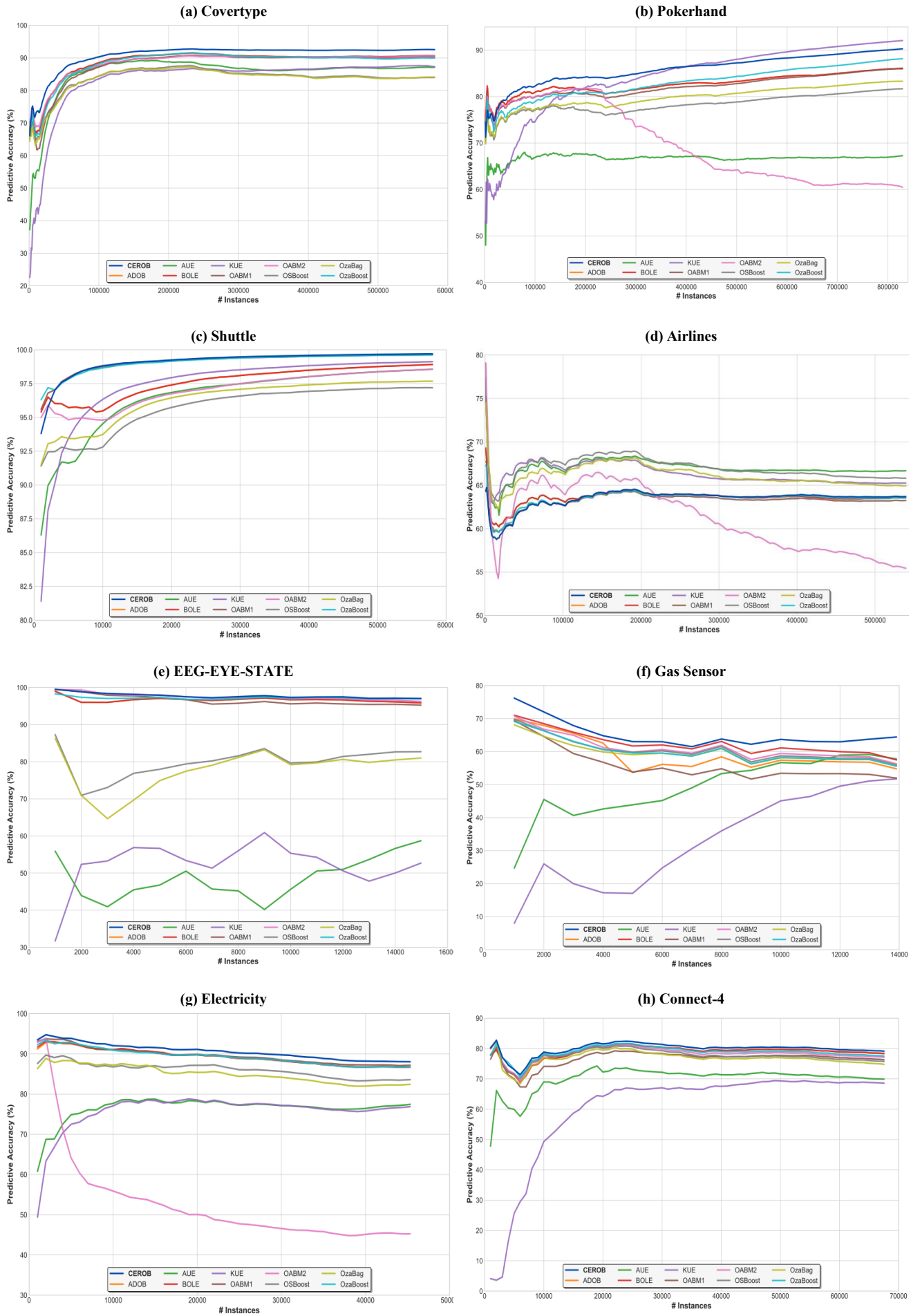

Fig. 4: Accuracy Results in Case Real-World Dataset.

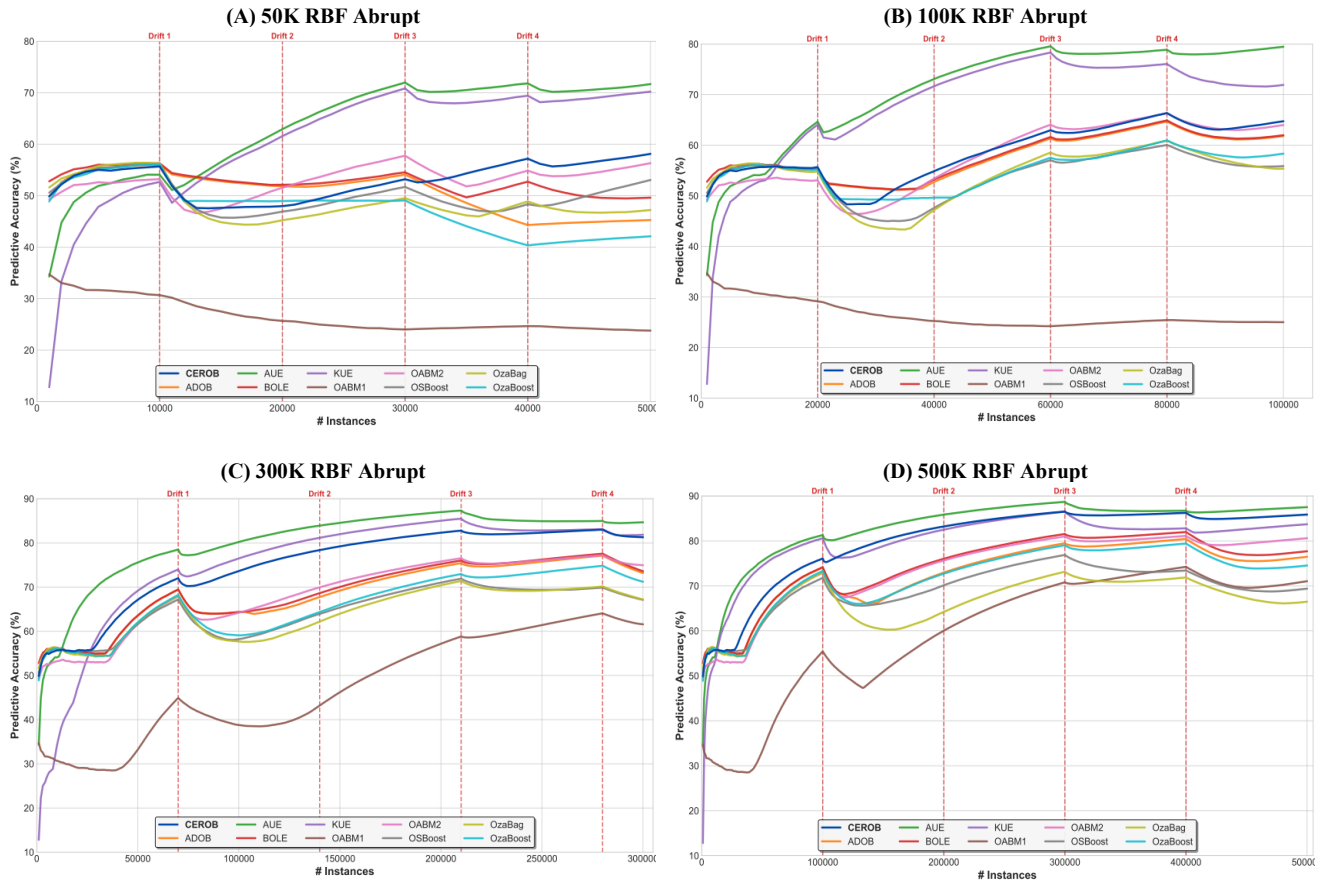

Fig. 5: Accuracy Results in Case of RBF Dataset.

5. Conclusion

Handling concept drifts in online environment is quite challenging because concepts change frequently. In this article we presented CEROB an ensemble algorithm based on Oza and Russell online boosting algorithm technique designed to handle abrupt drift more effectively. Our algorithm computes the diversity (λ) based on the accuracy of the individual learner, aiming to assign the current classifier's weight to the current instance in a data stream. Furthermore, we modified the traditional voting technique implemented in Oza and Russell for final ensemble prediction. Our main aim in this algorithm is to evaluate performance of Algorithm 1 and 2. For performance comparison, we used two different versions of four selected artificial datasets, involving both abrupt and gradual drifts, along with eight real world datasets. The results indicated that the performance of the proposed algorithm is outstanding in all context. Furthermore, we used a statistic based non-parametric Friedman test to compare different state-of-art methods, which indicates that our algorithm is statistically outstanding in terms of accuracy when compared to other methods. Future work will be to prune and add new classifiers to the ensemble based on accuracy and to handle imbalanced data streams.

References

[1] I. Baidari and N. Honnikoll, "Bhattacharyya distance based concept drift detection method for evolving data stream," *Expert Syst. Appl.*, vol. 183, p. 115303, 2021, https://doi.org/10.1016/j.eswa.2021.115303.

[2] R. S. M. Barros, D. R. L. Cabral, P. M. Gonçalves, and S. G. T. C. Santos, "RDDM: Reactive drift detection method," *Expert Syst. Appl.*, 2017, https://doi.org/10.1016/j.eswa.2017.08.023.

[3] J. Gama, P. Medas, G. Castillo, and P. P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence – SBIA 2004*, vol. 3171, Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 286–295. doi: 10.1007/978-3-540-28645-5_29.

[4] L. Hu, Y. Lu, and Y. Feng, "Concept Drift Detection Based on Deep Neural Networks and Autoencoders," *Applied Sciences*, vol. 15, no. 6, 2025, https://doi.org/10.3390/app15063056.

[5] R. S. M. de Barros and S. G. T. de C. Santos, "An overview and comprehensive comparison of ensembles for concept drift," *Information Fusion*, 2019, https://doi.org/10.1016/j.inffus.2019.03.006.

[6] Y. Freund and R. E. Schapire, "Experiments with a New Boosting Algorithm," *Proceedings of the 13th International Conference on Machine Learning*, pp. 148–156, 1996.

[7] N. C. Oza and S. J. Russell, "Online bagging and boosting," in *International workshop on artificial intelligence and statistics*, 2001, pp. 229–236.

[8] S. T. Chen, H. T. Lin, and C. J. Lu, "An online boosting algorithm with theoretical justifications," in *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, 2012, pp. 1007–1014.

[9] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà, "New ensemble methods for evolving data streams," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009. https://doi.org/10.1145/1557019.1557041.

[10] A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," in *Machine Learning and Knowledge Discovery in Databases*, vol. 6321, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 135–150. https://doi.org/10.1007/978-3-642-15880-3_15.

[11] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognit. Lett.*, vol. 33, no. 2, pp. 191–198, Jan. 2012, https://doi.org/10.1016/j.patrec.2011.08.019.

[12] J. Z. Kolter and M. A. Maloof, "Dynamic weighted majority: An ensemble method for drifting concepts," *Journal of Machine Learning Research*, vol. 8, pp. 2755–2790, 2007.

[13] L. L. Minku and X. Yao, "DDD: A new ensemble approach for dealing with concept drift," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 4, pp. 619–633, 2012, https://doi.org/10.1109/TKDE.2011.58.
[14] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Fourth international workshop on knowledge discovery from data streams*, 2006, pp. 77–86.
[15] D. Brzezinski and J. Stefanowski, "Reacting to different types of concept drift: The accuracy updated ensemble algorithm," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 1, pp. 81–94, 2014, https://doi.org/10.1109/TNNLS.2013.2251352.
[16] S. G. T. D. C. Santos, P. M. Gonçalves, G. D. D. S. Silva, and R. S. M. De Barros, "Speeding up recovery from concept drifts," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, pp. 179–194. https://doi.org/10.1007/978-3-662-44845-8_12.
[17] R. S. M. De Barros, S. G. T. De Carvalho Santos, and P. M. G. Junior, "A Boosting-like Online Learning Ensemble," in *Proceedings of the International Joint Conference on Neural Networks*, 2016, pp. 1871–1878. https://doi.org/10.1109/IJCNN.2016.7727427.
[18] S. G. T. de Carvalho Santos and R. S. M. de Barros, "Online adaboost-based methods for multiclass problems," *Artif. Intell. Rev.*, vol. 53, no. 2, pp. 1293–1322, 2020. https://doi.org/10.1007/s10462-019-09696-6.
[19] I. Fr\'\ias-Blanco, J. del Campo-Ávila, G. Ramos-Jiménez, R. Morales-Bueno, A. Ortiz-D\'\iaz, and Y. Caballero-Mota, "Online and non-parametric drift detection methods based on Hoeffding's bounds," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 3, pp. 810–823, 2015. https://doi.org/10.1109/TKDE.2014.2345382.
[20] A. L. Freire, G. A. Barreto, M. Veloso, and A. T. Varela, "Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study," in *2009 6th Latin American Robotics Symposium, LARS 2009*, 2009. https://doi.org/10.1109/LARS.2009.5418323
[21] S. Zhong, W. Tang, and T. M. Khoshgoftaar, "Boosted noise filters for identifying mislabeled data," *Department of Computer Science and engineering, Florida Atlantic University*, 2005.
[22] R. D. King, C. Feng, and A. Sutherland, "Statlog: Comparison of classification algorithms on large real-world problems," *Applied Artificial Intelligence*, 1995, https://doi.org/10.1080/08839519508945477.
[23] A. Vergara, S. Vembu, T. Ayhan, M. A. Ryan, M. L. Homer, and R. Huerta, "Chemical gas sensor drift compensation using classifier ensembles," *Sens. Actuators B Chem.*, vol. 166–167, no. 3, pp. 320–329, May 2012, https://doi.org/10.1016/j.snb.2012.01.074.
[24] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, 2006.