Semiconductor Optoelectronics, *Vol. 42 No. 1 (2023), 01-15*
https://bdtgd.cn/

Semiconductor Optoelectronics
ISSN:
1001-5868

ONLINE BOOSTING METHOD BASED ON CLASSIFIER ERROR RATE

Nagaraj Honnikoll*, [a], Ishwar Baidari [a]
[a] Department of Computer Science, Karnatak University, Dharwad, Karnataka, 580003, India

Abstract:
Online learning approaches often have to operate in the presence of data distribution (concept drift), it is a challenge that is steadily attracting attention to a great extent. This paper introduces Online Boosting Method based on Classifier Error Rate (OBMCER) based on heuristic adjustment to Oza and Russell's Online Boosting. In exact terms, we empirically examine the facts of a) use classifier error rate for the calculation of diversity of distribution for training, b) computing weight distribution for the classifier using classifier error rate. OBMCER was try-out against the original and other modified versions of both boosting methods as well as other ensembles using popular artificial and real-world data sets. Results suggest that the proposed method achieves high accuracies, outperforming other state-of-art methods.

Keyword: Data Streams, Online Boosting, Online Learning, Ensemble Learning.

1. Introduction:

At the present time, numerous applications need the use of procedures that enable the extraction of information in real-time. Examples of such applications include industrial process control, monitoring the purchase history of customers, TCP/IP traffic. Thus, the methods used for this reason must be constantly updated to make it suitable for new instances, considering the computational constraints as well.

Boosting (Freund, 1995; Freund & Schapire, 1996) is a very popular and general method in order to improve the accuracy of other base learners. The aim is to train several base learners using different distributions over the training data merging them into an ensemble. Notice that several boosting methods come with some theoretical assurance about their results.

The accomplishment of boosting based on AdaBoost.M1 (Freund, 2001) only allows a base learner to vote if its error is below 50%, the value connected to random guessing. However, when the problem is not binary, many times the necessity of 50% is too strong (Freund & Schapire, 1996).

Furthermore, AdaBoost.M1 also stops executing a current instance of data quickly when it finds a base learner error greater than 50%. It is not a good idea to rejecting an instance because, one of the classifiers shows low accuracy because they access each data only once.

Oza and Russell's proposed a new online boosting (OzaBoost) (Oza, 2005) which makes use of Poisson distribution for the data distribution in an online setting. Online Boosting has been motivated by AdaBoost.M1 of the offline boosting method. Motivated by Oza and Russell's Online Boosting (OzaBoost), ADOB (de Carvalho Santos et al., 2014) is proposed, in which diversity of distribution for training data set has been modified to speedy recovery when drift

occurs.
After closely observing the aforementioned methods, we improved the Oza-Boost by adopting two new strategies with the intention to improve the accuracy of the ensemble. One of the strategies is to use classifier error rate for the calculation of diversity of distribution for training, to speed up recovery. Another strategy is to use the classifier error rate for the calculation of the weight distribution for the classifier. Based on the results of the experiments, Online Boosting Method based on Classifier Error Rate (OBMCER) maintains good accuracy in different situations, surpassing the OzaBoost in all the cases.
The rest of the paper is organized as follows: Section 2 presents the related work; Section 3 presents the detailed explanation of the proposed boosting method. Section 4 presents the experimental settings and datasets used. Section 5 presents the experimental results and analysis. Section 6 presents conclusion.

2. Related Work

Bagging and Boosting (Freund et al., 1996) is based on a strategy to train base learners on training data and by combining the output of individual base learners to give better predictions using various strategies.
Oza and Russell's (Oza, 2005) Online Bagging and Boosting is based on AdaBoost.M1 (Freund, 2001). The quantity of training that individual base learners will accept on distinct instance showed by its weight will be accomplish using a Poisson distribution. When the weights are higher, there is a greater possibility of the classifiers are going to get more training. This function is alike to the both methods.
OSBoost (Chen et al., 2012) is based on Smooth Boost (Servedio, 2003) uses smoother distribution concept to set new need for base learners, in order to revise the weights in a more traditional way, showing theoretical proof of performance. Given the base learners satisfy the constraint of showing prediction error better than the random guess in a distribution without difficulties, the last hypothesis will have a less error rate.
ADOB (de Carvalho Santos et al., 2014) is a boosting ensemble based on OzaBoost(Oza, 2005). It has a distinct approach to increase in speed among the expert's recovery after drifts. ADOB arrange the experts systematically according to accuracy before executing each instance before influencing the way dissimilarity is distributed to the classifiers and slightly enhancing the accuracy of the ensemble just after the drifts, mainly when these concept drifts are abrupt. Note ADOB also uses a configurable auxiliary drift detector.
BOLE (de Barros et al., 2016) is based on plain heuristic alteration to ADOB (de Carvalho Santos et al., 2014). More explicitly, BOLE debilitate the need to permit the experts to vote, enhancing the ensemble accuracy in most conditions, mainly when the drifts are frequent and/or abrupt, where the accuracy procures can be too high. Furthermore, BOLE gives very good performance in maximum datasets, regardless of the auxiliary drift identification approach used.
Dynamic Weighted Majority (DWM) (Kolter & Maloof, 2007) ensemble is based on weights that expands the Weighted Majority Algorithm (WMA) (Blum, 1997) to detect drifts. DWM joins and take away base learners as per its overall performance: a base learner is joined when the ensemble miss-classify; the weight of individual base learner is decreased when it miss-classifies; and a base learner is removed when its weight is very less, pointing out it presented

low accuracy on many examples.

Diversity for Dealing with Drifts (DDD) (Minku & Yao, 2012) make use of four ensemblewith high and low diversity, before and after a drift is identified. It makes an attempt to choose the ensemble which are greater in number before and after drifts identified by the EDDM (Baena-Garc\ia et al., 2006).

OABM1 and OABM2 (Santos & de Barros, 2020) are boosting ensemble based on AdaBoost.M1 and AdaBoost.M2 (Freund & Schapire, 1996), respectively. The OABM1 modifies the activity account- able for revising the instances weights of OzaBoost, preserving the features of the traditional batch version. The OABM2 is focusing at multiclass problems, targeting not only on complicated instances but also on complicated classes. The main distinctive feature between OABM1 and OzaBoost is the way the weights are revised. The constraints considered in OzaBoost i.e., $\lambda_{msc}> N/2$ and $\lambda_{msc}< N/2$ and does not promise the weights will be properly revised in all conditions when$\epsilon m \leq 1/2$. On the other hand, the function opted in OABM1 for revising the weights does promise the weights will be properly revised in all scenarios. Similarly, OABM2 made ready to handle difficulties that have more than two classes and perform the job by increasing the weights of the weak learners in both the instances and the classes which are difficult to sort.

2.1 Online Boosting

Algorithm 1: Online Boosting ($h_M, OnlineBase, d$)

```
• Set the example's "weight" $\lambda_d \leftarrow 1$
• For each base model $h_m, (m \in \{1,2,\ldots,M\})$ in the ensemble,
    • 1. Set $k$ according to $Pois(\lambda_d)$.
    • 2. Do $k$ times
            $h_m \leftarrow OnlineBase(h_m, d)$
    • 3. If $h_m(d)$ is the correct label,
        ▪ then
            • $\lambda^{sc}_m \leftarrow \lambda^{sc}_m + \lambda_d$
            • $\lambda_d \leftarrow \lambda_d \times \frac{N}{(2\times\lambda^{sc}_m)}$
        ▪ else
            • $\lambda^{sw}_m \leftarrow \lambda^{sw}_m + \lambda_d$
            • $\lambda_d \leftarrow \lambda_d \times \frac{N}{(2\times\lambda^{sw}_m)}$
To Classify new examples:
• For each $m \in \{1,2,\ldots,M\}$
  Calculate $\epsilon_m = \frac{\lambda^{sw}_m}{(\lambda^{sc}_m+\lambda^{sw}_m)}$ and $\beta_m = \frac{\epsilon_m}{1-\epsilon_m}$
• Return $h(x) = argmax_{c\in C} \sum_{m:h_m(x)=y} log \frac{1}{\beta_m}$
```

Online Boosting is based on AdaBoost.M1 (Freund et al., 1996), the main strategy is to train the base learners after the drifts are identified. Algorithm 1 indicates the Online Boosting (OzaBoost) pseudo-code as published in (Oza, 2005).
Notice that OzaBoost uses the Poisson distribution parameter (λ_d) for the diversity of distribution for the training classifier. If the classifier miss-classifies an instance, the parameter (λ_d) will be decreased, otherwise it is increased. The increase or decrease of the parameter is done as shown below:

$$\lambda_d \leftarrow \lambda_d \underset{m}{\times} \frac{N}{(2 \times \lambda^{sc})}$$

$$\lambda_d \leftarrow \lambda_d \underset{m}{\times} \frac{N}{(2 \times \lambda^{sw})}$$

For the weight calculations, the error (ϵ_m) is calculated which mainly depends on the parameter(λ_d). For the error calculated, (β_m) is calculated which will be taken with a log for the assignment of weight for the current classifier (m).

Table 1 presents the notations used in the rest of the paper with their meanings.

Table 1: Used notations and their meanings.

Notations	Meaning
d	A data point or instance.
M	Size of the ensemble.
m	Current expert in the ensemble M.
ϵ_m	Error rate for expert m.
β_m	Weight assigned to the current expert m.
λ	Poisson distribution parameter.
$\lambda^{sc}\ m$	Sum of instances correctly classified by the expert m.
$\lambda^{sw}\ m$	Sum of instances wrongly classified by the expert m.
N	Number of instances seen so far in the data stream.
p_m	Classifier error rate of m.

3. Proposed Method

This section introduces the proposed strategies, one is distribute instances among classifiers in such a manner as to achieve a desired result. Another one is computing the weight of the current classifier.
Online Boosting Method based on Classifier Error Rate (OBMCER) proposes to continue the properties of traditional OzaBoost method, which proposes to distribute instances effectively among classifiers, aimed at train the classifier quickly under current environment of the data distribution. The distribution is controlled by the diversity, which is a Poisson distribution parameter (λ_d).

Definition 1. *Classifier Error Rate:*

OBMCER uses classifier error rate or mean error rate to implement both the strategies effectively in order to improve the accuracies.

It is defined as:

$$p_m \leftarrow p_m + \frac{(pred - p_m)}{N}$$

Where, pred is the prediction of the current classifier, which is either 0 or 1. Initial value of is one.

Definition 2. *Mean Accuracy:*

Based on the classifier error rate (p_m), the computed accuracy mean is used for calculating thesum of correctly classified (λ^{sc}) and sum of wrongly classified (λ^{sw}) weights of the classifier. The mean accuracy is defined as:

$$M_{acc} \leftarrow (1 - p_m)$$

Definition 3. *Normalized Error:*

The classifier error rate (p_m) of the classifier is weighted by the current data distribution, so we compute the normalized error as:

$$\beta_m \leftarrow \frac{p_m}{(1 - p_m)}$$

Definition 4. *Diversity distribution:*

For the diversity distribution of the data, the Poisson distribution parameter (λ_d) has been modified with respect to the classifier error rate, which is defined as:

$$\lambda_d \leftarrow \lambda_d \times \beta_m^{\alpha} \times \frac{\lambda_m^{sc} + \lambda_m^{sw}}{2 \times \lambda_m^{sw}}$$

Based on these definitions, Algorithm 2 provides the protocols to overcome the deficiencies of OzaBoost. To perform distribution correctly, Oza and Russell (Oza, 2005) adapted the rational, involving previous classifier Poisson distribution parameter (λ_d) for sum of the correctly classified (λ^{sc}) and the sum of the wrongly classified (λ^{sw}) for deciding the amount of training that each classifier will receive. However, this strategy in line 3 of algorithm 1, does not promisethe weights will be updated properly in all the scenarios, due to involvement of previous classifier's for updating weights of current classifier, not only on the error ϵm giving space for exception.

On the other hand, the rational adopted in OBMCER for updating the weights for Poisson distribution (λ_d)) and classifier as well, detailed in definitions above, does indeed guarantee the weights increases after wrong prediction and decrease after the correct ones in all scenarios, leads to tremendous impact on accuracies.

Algorithm 2: OBMCER ($h_M, OnlineBase, d$)

```
Set the example's "weight" $\lambda d \leftarrow 1$, a $\leftarrow$ 1
For each base model $hm$, ($m \in \{1, 2, . . . , M\}$) in the ensemble,
        1. Set $k$ according to $Pois(\lambda d)$.
        2. Do $k$ times
                $hm \leftarrow OnlineBase(hm, d)$
        3. If $h(d)$ is the correct label, then $pred \leftarrow 0$ else $pred \leftarrow 1$

        4. Calculate classifier error rate, $pm \leftarrow pm + \frac{(pred-pm)}{N}$
        5. Mean Accuracy, $Macc \leftarrow (1 - pm)$
        6. If $h(d)$ is the correct label,
                ▪ then
                        • $\lambda^{sc}_m \leftarrow \lambda^{sc}_m + M \quad acc$
                        • a$\leftarrow$ 1
                • $\beta m \leftarrow \frac{pm}{(1-pm)}$
                ▪ else
                  $\lambda^{sw}_m \leftarrow \lambda^{sw}_m + M acc$
                  a$\leftarrow$ 0
                  $\lambda + \lambda\ sc\ sw$
        7. $\lambda d \leftarrow \lambda d \times \beta^{\alpha}_m \times \frac{m}{2\times\lambda sw}$
To Classify new examples:
For each $m \in \{1, 2, \dots , M\}$

Calculate $\beta m \leftarrow \frac{pm}{(1-pm)}$

Return $h(x) = argmax\sum_{m:hm(x)=y} log\ 1c\in C\ \beta m$
```

4. Experimental settings

This section expresses the test planned to experiment and estimate our ideas. Specifically, the meaningful medication, proposed to modify the way diversity is distributed during training to speed up the expert's recovery in all scenarios. We tested against other ensemble aimed at learning from data streams with concept drifts.

To evaluate accuracy, we use prequential test methodology (Bifet et al., 2010)(Gama et al., 2013) with a sliding window as it forgetting mechanism (default in MOA), each input of the instance is tested and then it is used for training. This procedure promises that each and every instance make use of testing and training and avoids the conflict of training before testing on any given instance. We generated three artificial datasets of distinct complexities, and constructed abrupt and gradual drift versions of four distinct sizes (50K, 100K, 500K and 1M instances respectively) of whole 24 artificial datasets.

Furthermore, we selected eight real world datasets to supplement the estimation of the proposed method, which were collected from UCI machine learning repository. The artificial datasets generated and real-world datasets, along with their characteristics has been given in Table 2 and Table 3.

Table 2: Characteristic of Artificial Datasets Generated.

Dataset	#Instances	#Attributes	#Classes	#Drifts	Drift Type
MIXED	50K, 100K, 500K, 1M	4	2	4	Abrupt, Gradual
SINE	50K, 100K, 500K, 1M	4	2	4	Abrupt, Gradual
RANDOM TREE	50K, 100K, 500K, 1M	10	8	4	Abrupt, Gradual

Table 3: Characteristic of Real World Datasets.

Dataset	#Instances	#Attributes	#Classes	#Drifts	Drift Type
Pokerhand (Bifet et al., 2009)	829,201	10	10	Unknown	Unknown
Covertype (Fr\'\ias-Blanco et al., 2015)	581,012	53	7	Unknown	Unknown
Electricity (Gama et al., 2004)	45,312	9	2	Unknown	Unknown
Wall-Following Robot Navigation (WFRN) (Freireet al., 2009)	5,456	24	4	Unknown	Unknown
Connect-4 (Zhong et al., 2005)	67,557	42	3	Unknown	Unknown
Segment (Zhong et al., 2005)	2,310	19	7	Unknown	Unknown
Shuttle (King et al., 1995)	58000	9	7	Unknown	Unknown
Gas Sensor Array Drift(Gas Sensor) (Vergara et al., 2012)	13,910	129	6	Unknown	Unknown

5. Experimental results and analysis

This part introduces the outcomes of the performed experiments, including analysis of accuracy of the methods over the selected datasets using Hoeffding Tree (HT) as base learner.

5.1 Accuracy results and analysis

Table 4 shows the accuracy outcomes of the tested approaches in all chosen datasets as well as their ranks using Hoeffding Tree. In each and every dataset and in the ranks, the best outcomes indicated in bold. In absolute terms, the proposed approach improved the predictive accuracies of OzaBoost in all most all experimented configurations, across all three experimented dataset generators, and in the real-world datasets.

Furthermore, in the artificial datasets with both abrupt and gradual drifts, OBMCER presented outstanding performance in all the versions using Hoeffding Tree as base learner, whereas OzaBoost and other approaches were normally the best. As a result, OBMCER was the best ranked approach in the experiments with Hoeffding Tree as base learner. It is also noticeable that OBMCER was comparatively more efficient in the real-world datasets. Note this fact does not mean OBMCER was superior to the other tested approaches. When compared to OzaBoost and OABM2, OBMCER consistently provide higher accuracies in all the 100K, 500K and 1M instances maintaining the same results in the real-world datasets as well.

Table 4: Average accuracies in percentage (%) using Hoeffding Tree, with 95% confidence intervals in the artificial datas

TYPE-SIZE	DATASET	OzaBag	OzaBoost	OSBoost	ADOB	BOLE	OABM1	OABM2
Abrupt - 50K	MIXED	89.62±5.14	94.04±1.38	89.95±4.68	94.50±1.46	94.50±1.46	94.21±1.31	93.69±1.2
	SINE	89.65±1.85	93.72±2.75	90.57±1.97	94.19±2.82	94.19±2.84	**94.48±2.28**	93.14±3.0
	RANDOM TREE	77.90±6.82	77.10±8.10	78.23±6.57	74.87±8.38	76.15±6.30	76.54±7.86	76.94±7.0
Abrupt - 100K	MIXED	91.49±3.30	95.89±1.58	91.02±4.00	96.07±1.56	96.07±1.56	95.92±1.63	95.64±1.3
	SINE	90.28±1.58	95.35±2.55	90.98±1.69	95.79±2.55	95.79±2.56	95.93±2.17	95.03±2.8
	RANDOM TREE	80.10±5.58	80.03±6.73	80.68±5.50	77.62±6.76	78.35±5.24	79.59±6.57	79.63±5.9
Abrupt - 500K	MIXED	95.08±1.75	98.34±1.32	94.87±2.00	98.44±1.25	98.44±1.25	98.31±1.33	98.31±1.2
	SINE	94.33±1.75	97.93±1.71	93.98±1.56	**98.17±1.63**	**98.17±1.64**	**98.17±1.46**	98.04±1.8
	RANDOM TREE	85.07±4.14	86.62±5.03	85.17±3.88	85.06±5.26	85.25±4.68	86.29±5.00	84.02±5.7
Abrupt - 1M	MIXED	96.42±1.41	98.92±1.08	96.33±1.51	98.96±1.01	98.96±1.01	98.87±1.08	98.90±1.0
	SINE	95.68±1.58	98.58±1.35	95.59±1.47	**98.74±1.27**	**98.74±1.27**	98.73±1.16	98.71±1.4
	RANDOM TREE	87.38±3.98	89.53±4.78	87.28±3.68	88.63±5.22	88.73±4.88	89.17±4.73	89.48±4.5
Gradual - 50K	MIXED	89.36±5.44	93.83±1.52	89.57±5.08	94.00±1.59	94.00±1.59	93.94±1.22	92.92±1.9
	SINE	89.64±1.85	93.70±2.75	90.58±1.97	94.14±2.79	94.13±2.81	94.60±2.33	93.10±3.0
	RANDOM TREE	77.90±6.82	77.10±8.11	78.23±6.57	74.86±8.37	76.14±6.29	76.55±7.87	76.94±7.0
Gradual - 100K	MIXED	90.95±4.09	95.64±1.54	90.55±4.65	95.94±1.55	95.94±1.55	95.83±1.59	95.40±1.3
	SINE	90.38±1.62	95.40±2.57	90.98±1.69	95.75±2.53	95.74±2.54	95.94±2.17	95.03±2.8
	RANDOM TREE	80.10±5.58	80.03±6.72	80.68±5.50	77.58±6.73	78.31±5.21	79.60±6.58	79.63±5.9
Gradual - 500K	MIXED	95.10±1.73	98.31±1.31	94.83±2.03	98.39±1.23	98.39±1.23	98.29±1.32	98.28±1.2
	SINE	94.42±1.78	97.92±1.70	93.97±1.56	98.16±1.63	98.16±1.63	**98.17±1.61**	98.03±1.8
	RANDOM TREE	85.07±4.14	86.63±5.03	85.17±3.88	85.06±5.26	85.25±4.68	86.29±5.00	86.90±5.0
Gradual - 1M	MIXED	96.40±1.45	98.90±1.07	96.28±1.56	98.94±1.00	98.94±1.00	98.86±1.08	98.88±1.0
	SINE	95.67±1.58	98.58±1.35	95.59±1.47	**98.74±1.27**	**98.74±1.27**	98.73±1.15	98.71±1.4
	RANDOM TREE	87.39±3.99	89.53±4.77	87.28±3.68	88.63±5.22	88.73±4.88	89.17±4.73	89.48±4.5
Real World	COVERTYPE	83.62	88.59	83.89	88.88	88.88	88.69	88.63
	POKERHAND	79.88	83.08	78.4	82.78	82.8	82.16	69.72
	ELECTRICITY	84.96	89.26	85.99	89.23	89.28	89.53	52.65
	CONNECT-4	76.77	78.79	77.59	78.74	78.76	78.47	78.24
	SEGMENT	77.43	81.1	79.22	80.83	82.17	80.25	81.38

	SHUTTLE	83.29	98.67	80.59	98.63	98.63	98.88	98.65
	GAS SENSOR	59.68	60.08	60.46	59.13	62.48	60.16	61.07
	WFRN	59.15	62.85	59.35	59.39	65.41	65.73	80.5
Rank	-	6.7878	4.303	6.4848	4.3787	3.7727	3.8333	4.9999

Complementing the analysis of the obtained results, we make use of the statistic (Demšar, 2006)(Barros et al., 2017), based on the non-parametric Friedman tests, to compare the outcomes of the tests. The null hypothesis agreed that all approaches are statistically equal, and because it was discarded, it indicates that there is a statistical difference in some of the approaches, but the test doesn't indicate which. Therefore, to find this, a Post-Hoc test is essential. We selected the Nemenyi-test (de Barros & de Carvalho Santos, 2019), which differentiate each approach against all others and make use of a critical difference as a reference. The outcomes of the experiments referring to the data in Table 4 are outlined in Fig. 1. Fig. 2-5 presents the accuracy results in case of Artificial and Real-world datasets.

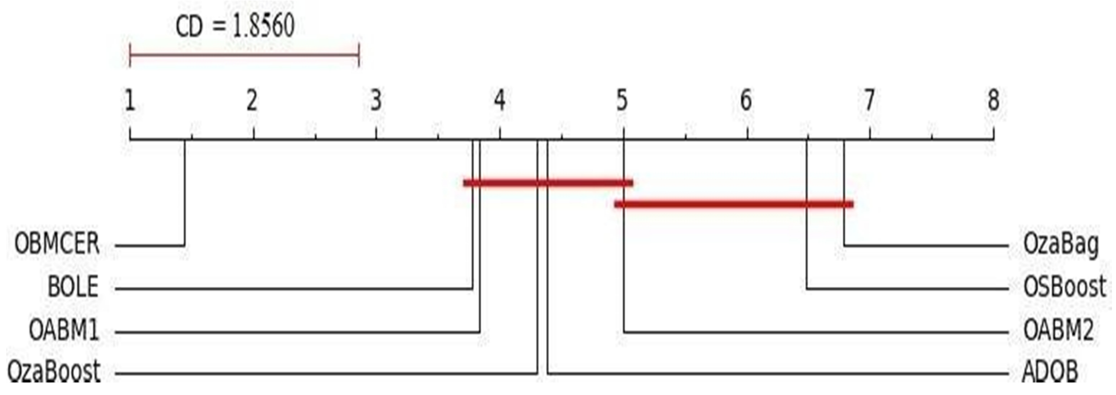
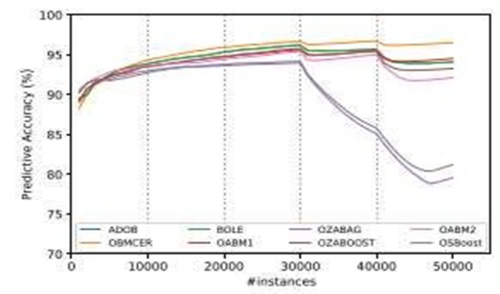


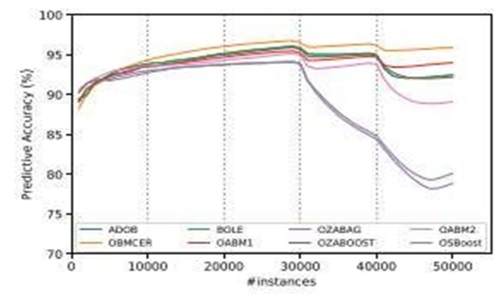
Figure 1: Comparison results of the m95% confidence on the artificial data

6. Conclusion

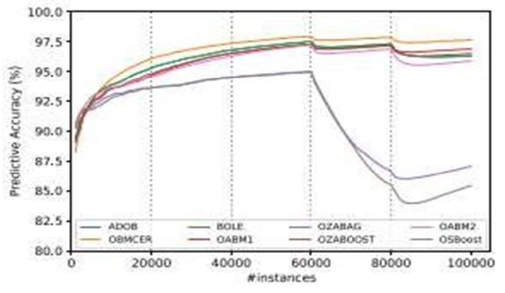
This article proposed OBMCER, which is new online boosting method based on OzaBoost. The OBMCER modified the basic task responsible for updating the instance weights of OzaBoost and the data diversity for training. The important properties of OBMCER were described, make sure appropriate behavior. Furthermore, it performs diversity of data distribution more efficiently which is based on classifier error rate to the instances compared to OzaBoost, aiming at regaining quickly from the conditions where changes occur frequently. We executed experiments to compare OBMCER to seven different online boosting methods. For comparative accuracy analysis, we used two different versions of three chosen artificial datasets and eight real world datasets as well. The tested OBMCER configuration shows good accuracy in several conditions. It is worth to emphasizing that OBMCER shows the best overall accuracy on all experimented datasets.



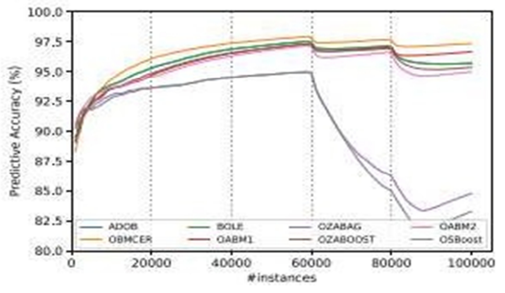
(a) 50K Abrupt



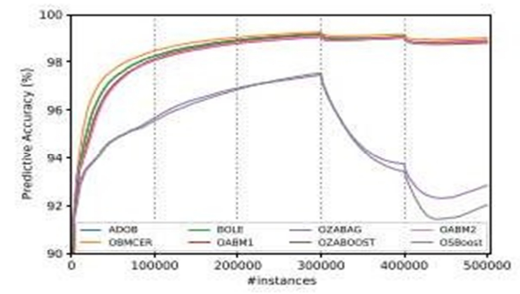
(b) 50K Gradual



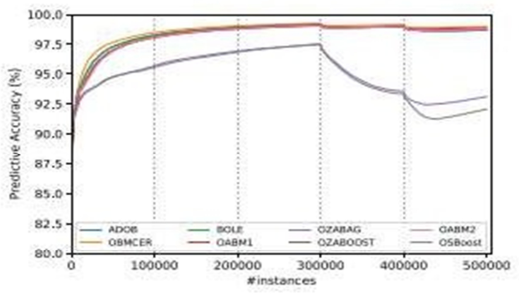
(c) 100K Abrupt



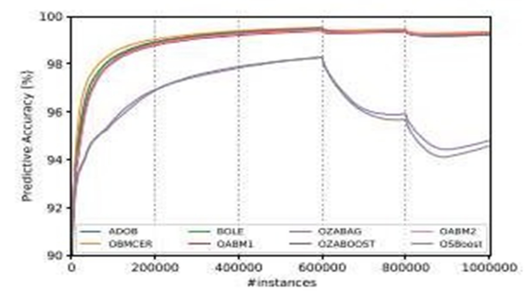
(d) 100K Gradual



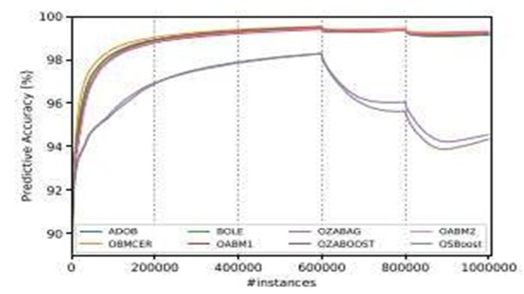
(e) 500K Abrupt



(f) 500K Gradual

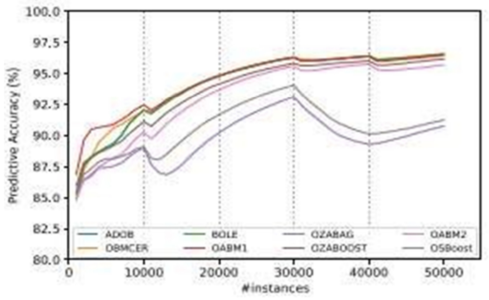


(g) 1M Abrupt

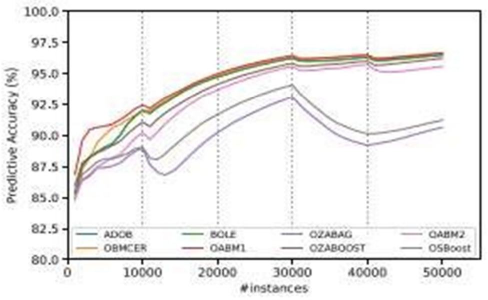


(h) 1M Gradual

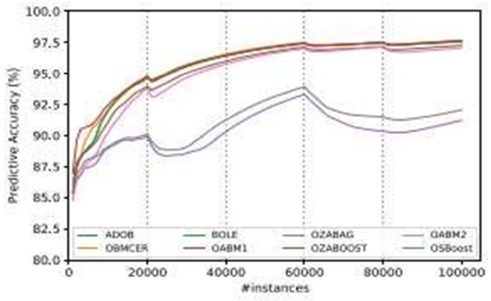
Figure 2: Accuracy results in case of Mixed Dataset



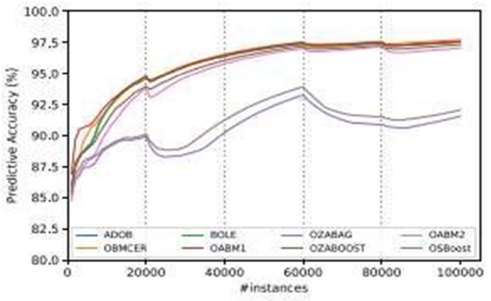
(a) 50K Abrupt



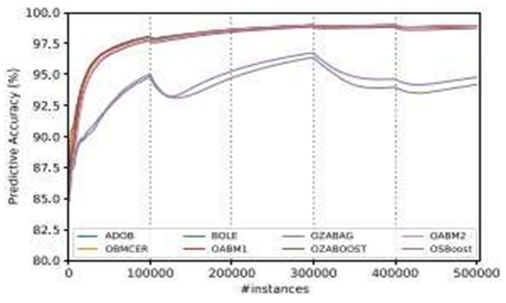
(b) 50K Gradual



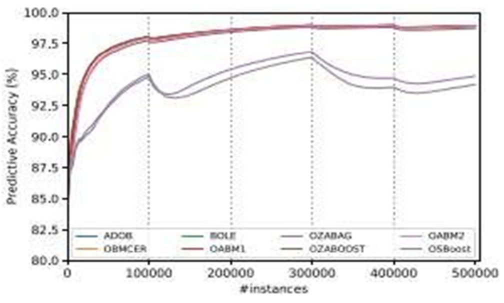
(c) 100K Abrupt



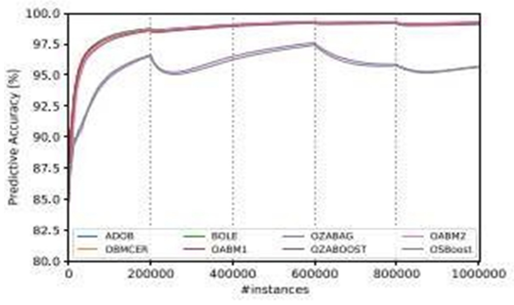
(d) 100K Gradual



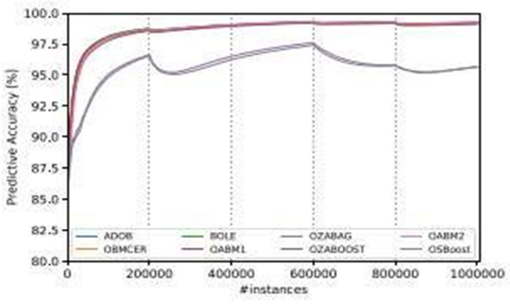
(e) 500K Abrupt



(f) 500K Gradual

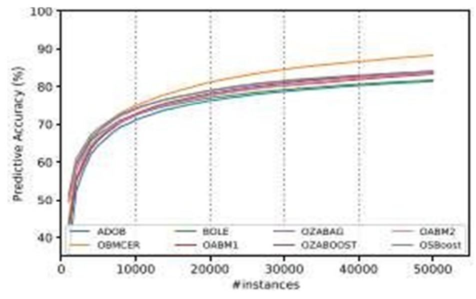


(g) 1M Abrupt

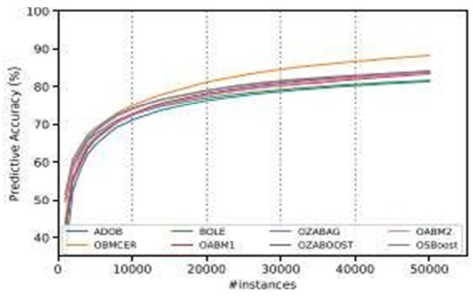


(h) 1M Gradual

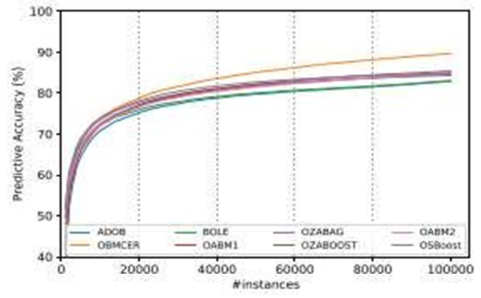
Figure 3: Accuracy results in case of Sine Dataset



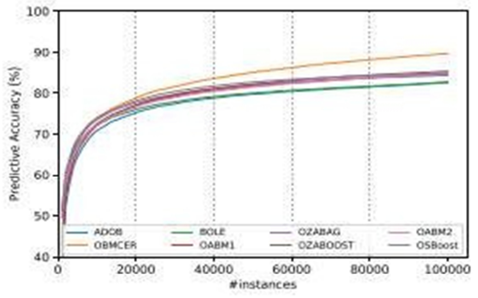
(a) 50K Abrupt



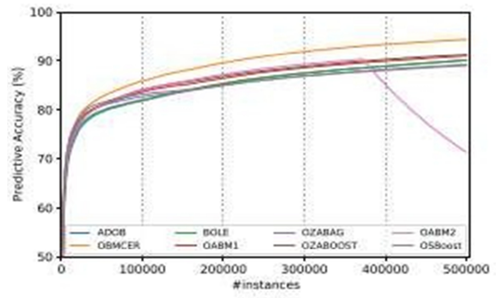
(b) 50K Gradual



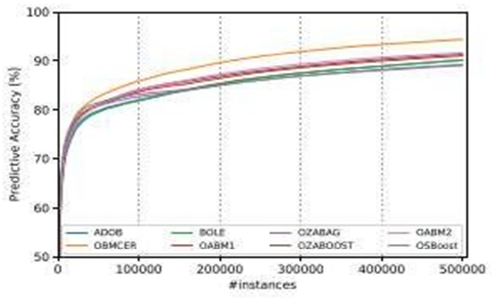
(c) 100K Abrupt



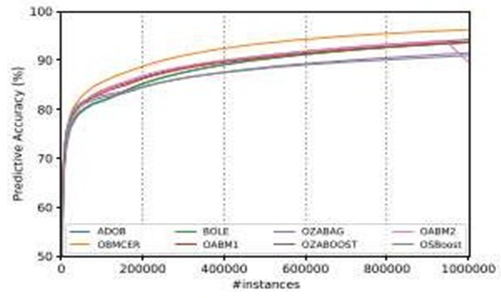
(d) 100K Gradual



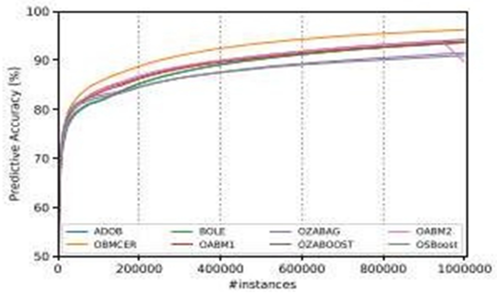
(e) 500K Abrupt



(f) 500K Gradual

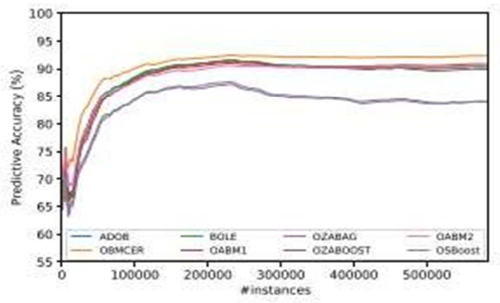


(g) 1M Abrupt

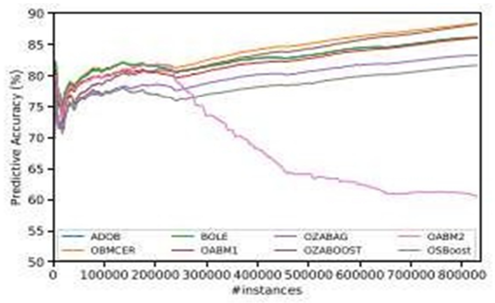


(h) 1M Gradual

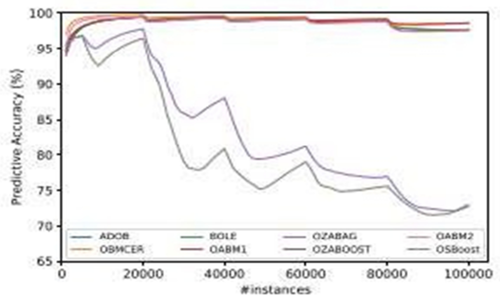
Figure 4: Accuracy results in case of Random Tree Dataset



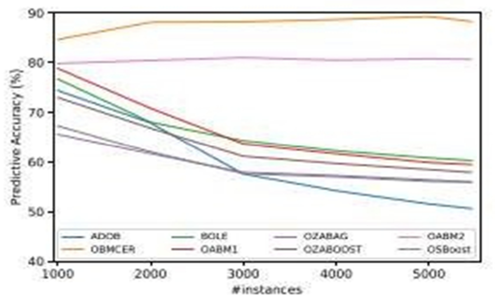
(a) Covertype



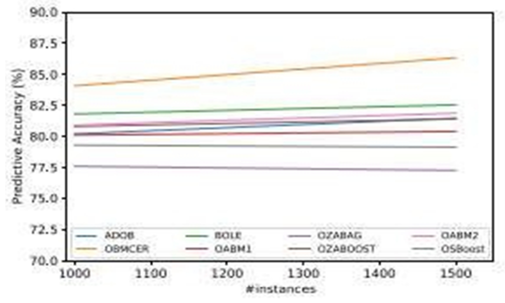
(b) Pokerhand



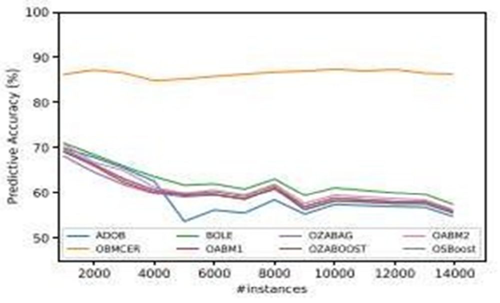
(c) Shuttle



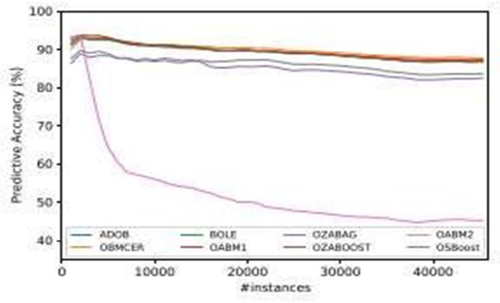
(d) WFRN



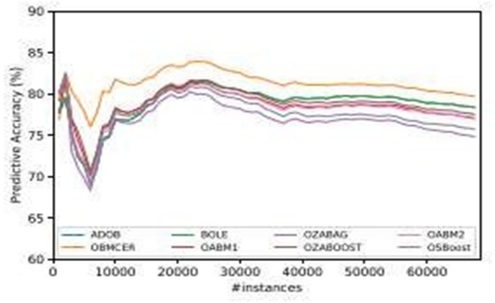
(e) Segment



(f) Gas Sensor



(g) Electricity



(h) Connect-4

Figure 5: Accuracy results in case of Real World Dataset

References:

Baena-Garc\ia, M., del Campo-Ávila, J., Fidalgo, R., Bifet, A., Gavalda, R., & Morales-Bueno, R. (2006). Early drift detection method. *Fourth International Workshop on Knowledge Discovery from Data Streams*, *6*, 77–86.

Barros, R. S. M., Cabral, D. R. L., Gonçalves, P. M., & Santos, S. G. T. C. (2017). RDDM: Reactive drift detection method. *Expert Systems with Applications*. https://doi.org/10.1016/j.eswa.2017.08.023

Bifet, A., Holmes, G., Kirkby, R., & Pfahringer, B. (2010). MOA: Massive Online Analysis. *Journal of Machine Learning Research*, *11*, 1601–1604.

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

Blum, A. (1997). Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning*, *26*(1), 5–23. https://doi.org/10.1023/A:1007335615132

Chen, S. T., Lin, H. T., & Lu, C. J. (2012). An online boosting algorithm with theoretical justifications. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, *2*, 1007–1014.

de Barros, R. S. M., & de Carvalho Santos, S. G. T. (2019). An overview and comprehensive comparison of ensembles for concept drift. *Information Fusion*, *52*, 213–244.

de Barros, R. S. M., de Carvalho Santos, S. G. T., & Junior, P. M. G. (2016). A Boosting-like Online Learning Ensemble. *Proceedings of the International Joint Conference on Neural Networks*, *2016-Octob*, 1871–1878. https://doi.org/10.1109/IJCNN.2016.7727427

de Carvalho Santos, S. G. T., Júnior, P. M. G., dos Santos Silva, G. D., & de Barros, R. S. M. (2014). Speeding up recovery from concept drifts. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 179–194.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*.

Freire, A. L., Barreto, G. A., Veloso, M., & Varela, A. T. (2009). Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. *2009 6th Latin American Robotics Symposium, LARS 2009*. https://doi.org/10.1109/LARS.2009.5418323

Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, *121*(2), 256–285. https://doi.org/10.1006/inco.1995.1136

Freund, Y. (2001). An adaptive version of the boost by majority algorithm. *Machine Learning*, *43*(3), 293–318. https://doi.org/10.1023/A:1010852229904

Freund, Y., icml, R. S.-, & 1996, undefined. (1996). Experiments with a new boosting algorithm. *Citeseer*. https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=d186abec952c4348870a73640bf849af9727f5a4

Freund, Y., & Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. *Proceedings of the 13th International Conference on Machine Learning*, 148–156. https://doi.org/10.1.1.133.1040

Fr\'\ias-Blanco, I., del Campo-Ávila, J., Ramos-Jiménez, G., Morales-Bueno, R., Ortiz-D\'\iaz, A., & Caballero-Mota, Y. (2015). Online and non-parametric drift detection methods based on Hoeffding's bounds. *IEEE Transactions on Knowledge and Data Engineering*, *27*(3), 810–823.

Gama, J., Medas, P., Castillo, G., & Rodrigues, P. (2004). Learning with drift detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. https://doi.org/10.1007/978-3-540-28645-5_29

Gama, J., Sebastião, R., & Rodrigues, P. P. (2013). On evaluating stream learning algorithms. *Machine Learning*, *90*(3), 317–346. https://doi.org/10.1007/s10994-012-5320-9

King, R. D., Feng, C., & Sutherland, A. (1995). Statlog: Comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence*. https://doi.org/10.1080/08839519508945477

Kolter, J. Z., & Maloof, M. A. (2007). Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, *8*, 2755–2790.

Minku, L. L., & Yao, X. (2012). DDD: A new ensemble approach for dealing with concept drift. *IEEE Transactions on Knowledge and Data Engineering*, *24*(4), 619–633. https://doi.org/10.1109/TKDE.2011.58

Oza, N. C. (2005). Online bagging and boosting. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, *3*, 2340–2345. https://doi.org/10.1109/icsmc.2005.1571498

Santos, S. G. T. de C., & de Barros, R. S. M. (2020). Online AdaBoost-based methods for multiclass problems. *Artificial Intelligence Review*, *53*(2), 1293–1322. https://doi.org/10.1007/s10462-019-09696-6

Servedio, R. A. (2003). Smooth boosting and learning with malicious noise. *Journal of Machine Learning Research*, *4*(Sep), 633–648.

Vergara, A., Vembu, S., Ayhan, T., Ryan, M. A., Homer, M. L., & Huerta, R. (2012). Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, *166–167*(3), 320–329. https://doi.org/10.1016/j.snb.2012.01.074

Zhong, S., Tang, W., & Khoshgoftaar, T. M. (2005). Boosted noise filters for identifying mislabeled data. *Department of Computer Science and Engineering, Florida Atlantic University*.