2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Experimenting with Supervised Drift Detectors in Semi-supervised Learning

José Luis Martínez Pérez*, Roberto Souto Maior de Barros*, Silas Garrido Teixeira de Carvalho Santos*†
**Centro de Informática, Universidade Federal de Pernambuco*, 50740-560, Recife-PE, Brazil
{jlmp,roberto,sgtcs}@cin.ufpe.br
†*SiDi Institute*, 51110-160, Recife-PE, Brazil
s.garrido@sidi.org.br

***Abstract*—Machine learning algorithms to aid decision-making processes are increasingly common in several areas. When fully-trained, algorithms tend to perform better, but the availability of data labels shortly after testing without human intervention is a challenging task in many areas, especially in data stream learning with concept drifts, where data is generated very fast, in real-time, with the possibility of changes in the data distribution. Concept drifts have been addressed in different ways, but using drift detectors with base classifiers in semi-supervised learning is *not* so common. This article shows how to use state-of-the-art *supervised* detectors in *semi-supervised* learning problems, and it also includes an extension to the MOA framework. The Experiments designed to test our proposal used Hoeffding Tree (HT) as base classifier, combined with eight drift detectors and a total of 62 artificial and five real-world datasets, configured with 15% and 30% labeled instances. The results indicate that drift detectors designed for supervised learning can also be effectively used in semi-supervised environments. This finding could lead to a change of paradigm for future research, since supervised drift detectors have never been considered as a viable alternative due to the absence of labels shortly after testing in many real-world data streams.**

***Index Terms*—Concept drift, data stream, semi-supervised learning, drift detection, online learning**

I. Introduction

Nowadays, online learning algorithms are increasingly used to analyze large data streams that arrive at great speed [1]. Such algorithms must be constantly updated, adapting quickly to new data instances. Besides, the distributions of the data can change over time (concept drift), which could degrade the learning models. Concept drift is often present in data mining and machine learning problems and applications, making the extraction of knowledge a challenging task that needs controlled computational resources.

Several classification algorithms use supervised inductive learning and rely on representative training samples; thus, labelled examples are provided by experts for training. Also, if a concept drift happens, the classifier must be retrained, requiring another training set. This process is complicated, expensive, and an ideal training set is not always available.

Also, some instances of the sample may be unlabeled, making its learning semi-supervised and creating extra difficulty in finding a good generalization. Moreover, concept drift detectors used with base classifiers in semi-supervised learning are difficult to find, and the ideal learner combines robustness to noise with sensitivity to concept drifts.

Semi-supervised classification can be seen as an extension of supervised classification where the training set (X) contains data labeled in the form $X_L = \langle \vec{x}_i, y_i \rangle_{i=1}^{l}$ and also unlabeled data represented as $X_U = \langle \vec{x}_i \rangle_{i=l+1}^{l+u}$. The variables $\vec{x}$ and y store the set of attributes and the label, respectively.

Most semi-supervised algorithms are extensions of supervised and/or unsupervised learners, thus inheriting some of their problems, such as high memory consumption and/or runtime. Semi-supervised classification sometimes has problems associated with labeling and the insertion of instances in training: a mistake in the process can negatively influence the model performance. For instance, even under most accommodating circumstances, learning under the cluster assumption can be hazardous and lead to a prediction performance worse than simply ignoring the unlabeled data and doing supervised learning [2]; also, the semi-supervised part of the methods should be used only when there is *not* enough labeled data [3].

This paper shows that concept drift detectors proposed for supervised learning with a base classifier can also be used in semi-supervised scenarios with high performance. The rationale is: in partially labeled data streams, the auxiliary detector is executed only when the processed instance of data is labeled.

To make the experiments and also their correct evaluation possible, the Massive Online Analysis (MOA) [4] framework was extended with new functionalities which allow for the preparation and execution of experiments with semi-supervised methods based on the simulation of unlabeled instances. This arrangement makes it possible to control how many labels are hidden from the methods and, at the same time, to measure the accuracy of the methods precisely.

We claim that this work might encourage a change of paradigm in the direction of future research, as many researchers have previously dismissed supervised learning in general and/or concept drift detectors in particular as unfeasible in real-world data streams by merely arguing that obtaining all labels shortly after testing is unrealistic [5], [6].

The rest of this paper is organized as follows: next section examines the literature about classification in semi-supervised learning (including tool support) and concept drift detectors; then, we describe the MOA extension implemented and used in the experiments; after, all the configuration details of the experiments are provided, followed by the experimental results and, finally, conclusions, including proposals for future work.

II. Related work

Currently, most semi-supervised online learning algorithms are variations of traditional machine learning ones. Methods such as COP-*k* [7], SEEDED-*k-means*, CONSTRAINED-*k-means* [8], and REDLLA [9] are all based on the *k-means* non-supervised algorithm [10]. Another widely used proposal is the CO-Training algorithm [11] from which other versions were also proposed [12], [13].

Semi-supervised ensembles that consider the presence of drifts include Semi-supervised Ensemble Learning of Data Streams in the Presence of Concept Drift (SSEL) [5], Weight Estimation Algorithm (WEA) [14], and Co-op Training [12].

Concept drift has been addressed from different points of view, bringing solutions with varying degrees of mathematical guarantees. Strategies for detecting the drifts are numerous and are meant to adapt the classifiers when concept drift occurs.

Concept drift detectors are used to improve the performance of classifiers. Several methods have been proposed over the years. Drift Detection Method (DDM) [15] controls the error rate (p) and its standard deviation (s) to detect concept drifts.

The Drift Detection Method based on the Hoeffding's inequality (HDDM) [16] has two versions: HDDM$_A$ and HDDM$_W$. HDDM$_A$ compares the moving averages to detect concept drifts and HDDM$_W$ uses the Exponentially Weighted Moving Average (EWMA) forgetting scheme [17] to weight the moving averages which are then compared to detect concept drifts. Both versions use Hoeffding's inequality [18] to set an upper bound to the level of difference between averages.

Fast Hoeffding Drift Detection Method for Evolving Data Stream (FHDDM) [19] detects concept drifts using a sliding window of size n and Hoeffding's inequality. A drift is detected when a significant difference between the maximum and the most recent probabilities of correct predictions is observed.

Reactive Drift Detection Method (RDDM) [20] modified DDM to tackle a well-known performance loss in long concepts. It signals a new drift type when the number of instances of the current concept reaches the predefined maximum number of instances and updates the DDM statistics using only the most recent instances without modifying the base learner. In addition, RDDM signals concept drifts when the number of instances under the warning level reaches a threshold limit.

The state-of-the-art of semi-supervised methods that use explicit change detection techniques is very limited. We found only five methods that can be included in this definition and all of them can be considered expensive in terms of execution, both run-time and memory consumption.

The Semi-supervised Adaptive Novel Class Detection and Classification over Data Stream (SAND) framework [6] is an ensemble and detects concept drifts by detecting changes in classifier confidences when testing and by detecting outliers having strong cohesion among themselves.

Efficent Concept Drift and Concept Evolution Handling over Stream Data (ECHO) [21] uses dynamic programming to reduce SAND's concept drift detection module time complexity, which is cubic. Its experimental results show that ECHO is significantly faster than SAND maintaining similar accuracy.

The Diversity Measure Drift Detection Method for Semi-supervised Environment (DMDDM-S) [22] uses two base classifiers at the same time, Hoeffding Tree (HT) and Naive Bayes (NB), to detect sudden drift by monitoring diversity. In DMDDM-S, the class labels of incoming data are not always necessary, but the authors use k-prototype clustering to label the unlabelled data, which are later used to train the model.

The Batch Training using Distance-based Score and Fixed Threshold Method (BDF) [1] is based on *Self-training* and trains on all labeled instances plus the most confident predictions of the unlabeled ones, based on a distance-based confidence score and a fixed confidence threshold.

Dynamic Selection Drift Detector (DSDD) [23] is based on online bagging and uses self-training online learning [24]. Its algorithm has three modules: ensemble creation, dynamic classifier selection, and drift detection. In the third module, for each member classifier, a detector is applied – DDM or Early Drift Detection Method (EDDM) [25].

Regarding tool support for semi-supervised data stream experiments, we are only aware of MOA-SS [1], a MOA extension that permits experiments using semi-supervised data streams. Although our proposal bears some similarities with MOA-SS, as both were implemented as extensions to MOA, we claim our proposal provides more functionality and better control over the generation of the semi-supervised data streams by using configurable parameters.

III. Implementation of Semi-supervised Support

To carry out this research, there were two main problems: (1) limited support to semi-supervised data streams learning; and (2) limited availability of semi-supervised methods with explicit concept drift detection. The first problem was solved by implementing new resources in the MOA framework and constitutes one of the contributions of this paper.

More specifically, the MOA framework classification tab was expanded with several new functionalities to support semi-supervised learning experiments.

1) A new learning type selector was created with values *Supervised* and *Semi-supervised*. If supervised learning is selected, MOA will work as the original version. However, when semi-supervised learning is selected, the algorithms will train only with the instances with a label, though still testing in all processed instances.
2) A new field was introduced to control the percentage of instances that will be considered unlabeled.
3) Another new field controls the number of instances with labels needed at the beginning of the training before starting to use data without labels.

An important adaptation was made to expand the repetition field, available in the MOAManager tool [26], which permits changing the seed used to generate the actual data in artificial dataset generators and repeating the experiments to obtain results with confidence intervals. The extended version uses this field also with real-world datasets, varying the places where the labels are hidden from the classifiers.

The aforementioned extensions allow for the generation of data streams for semi-supervised learning, both from artificial and real-world datasets. In the artificial datasets, it is also possible to configure concept drifts in the data streams. Notice that the provided implementation of the algorithm also avoids the removal of labels in an unequal way in different concepts.

Another contribution was to modify the *Other Tasks* tab to permit generating real-world semi-supervised datasets in the *arrf* format: they can be generated both from labeled real-world datasets and from scripts of artificial dataset generators.

It is also worth emphasizing that all the modifications mentioned above are compatible with the MOAManager tool and respect the principles of the original MOA implementation. In fact, because our implementation is strongly compatible with MOAManager, we decided to name it MOAManagerSS.

IV. Experiment Settings

This section presents all the configuration details of the experiments designed to evaluate the use of *supervised* concept drift detectors in *semi-supervised* stream problems.

Based on their popularity and efficiency [27], we selected DDM, HDDM$_A$, FHDDM and RDDM as supervised detectors. Since we could not find MOA codes for any of the semi-supervised methods, we implemented DMDDM-S, BDF and DSDD in MOA, mainly because we found their codes for other platforms. In addition, the DSDD version used in this work uses DDM as its drift detector, the one with the best performance in the results presented by the authors [23].

The hyper-parameters of all the tested methods were set to the values proposed by their respective authors or their default in MOA. However, an extra configuration of BDF (BDF_RDDM) was also tested, setting RDDM as its drift detector. Note that all detectors used HT as base classifier.

The tests used the MOAManager tool [26], integrated with MOA-SS [1] for the experiments using BDF, and with our MOAManagerSS for the remaining methods.

The accuracy evaluation used the Prequential methodology with a sliding window of size 1,000 as its forgetting mechanism [28], default in MOA: each data instance is firstly used for testing and then for training, and the final accuracy is based on the cumulative sum of the sequential errors over time.

Six dataset generators were used to configure 20k, 50k and 100k instances datasets with abrupt and gradual drift versions, and five real-world datasets were selected. For all of them, versions with 15%, 30% and 100% of the labels were created. The fully-labeled versions served as upper-bounds to provide better insight into how good the other results might be. The artificial datasets have four concept drifts regularly distributed.

The artificial dataset generators used in the experiments have all been extensively used in the area and their descriptions can be found elsewhere [27]. These are: Agrawal, LED, Mixed, Sine, Waveform, and RandomRBF. The real-world datasets are also well-known: Airlines, Connect4, Spam, and Weather. In addition, we used a version of Covertype proposed by Ienco Et al. [29], which is ordered by the elevation attribute, inducing gradual drifts.

V. Experimental results and analysis

This section presents and analyses the results of the tests prepared to compare the selected supervised drift detectors with the semi-supervised methods, also including a BDF version configured with RDDM (BDF_RDDM), using HT as base classifier in a semi-supervised environment.

The accuracies of the methods are presented in Tables I and II, referring to the artificial datasets containing abrupt and gradual drifts, respectively, and in Table, III regarding the real-world datasets. The best results are shown in **bold** in all these tables.

Analyzing the results in Tables I and II, in general, the supervised detectors were effective, even with fewer (15%) known labels. This can be confirmed by comparing the accuracy of the supervised detectors with the semi-supervised methods: in most cases, all the supervised methods delivered better results than those of the semi-supervised ones. With more labels (30%), the accuracy also increased, as expected. Two reasons explain this: (1) more training, since the classifier receives more labeled instances; and (2) concept drift detections happen earlier and, thus, are likely to be more accurate.

A natural conclusion is that fewer labels delay detections, as it takes more errors and instances to trigger them. Still, drifts are eventually detected and help classifier recovery, positively influencing the final accuracy. The relevance of the detectors becomes clearer by analyzing the two versions of BDF. In BDF_RDDM, the inclusion of RDDM improved the results in almost all scenarios and even delivered a few top results in Agrawal(F1-F5) and RandomRBF datasets with 15% of labels.

Regarding the detectors, RDDM stands out: its accuracies were the best or close to the best in most datasets. A likely beneficial characteristic of RDDM is its greater sensitivity in detecting concept drifts. Although this aspect potentially causes more false positives, with few labels available, RDDM will likely trigger drifts more quickly than the other detectors.

To complete this analysis, we look at the accuracies in the real-world datasets, shown in Table III. Despite presenting the same patterns, the performance gain of BDF_RDDM draws our attention: it suggests that aligning semi-supervised methods with supervised drift detectors can yield good results.

The results using 100% of the labels were omitted due to limited space. They were collected to serve as an upper bound and provide more insight on how much drop in performance the methods suffer from fewer labels: it turned out that such drops were smaller than expected. In most methods, with 15% of the labels, the drop in accuracy was up to 10% in most datasets, being smaller in the ones with larger concepts and in four of the real-world datasets. Moreover, in several datasets, there were drops of 2% or smaller. The most notable exception was DSDD, with much larger drops, mainly because it is an ensemble with high accuracies in the fully-labeled datasets.

In order to statistically evaluate the experiment results, we applied F_F [30], a variation of the non-parametric Friedman test: the ranks included in all the tables are the results of F_F. In addition, the methods were compared using the *Nemenyi* posthoc test [30] to point out the statistical differences.

TABLE I
MEAN ACCURACIES IN PERCENTAGE (%) WITH 95% CONFIDENCE AND HT BASE CLASSIFIER IN ARTIFICIAL DATASETS WITH ABRUPT DRIFTS

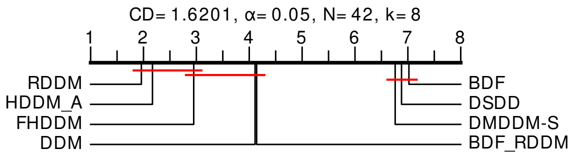
%Labels	SIZE	DATASET	DDM	HDDM$_A$	FHDDM	RDDM	DMDDM-S	BDF	BDF_RDDM	DSDD
15%	20K	Agrawal (F1-F5)	59.84±0.46	60.73±0.34	61.02±0.36	61.07±0.39	53.64±0.22	57.97±0.43	**62.32±0.39**	51.03±1.10
		Agrawal (F6-F10)	71.44±1.85	78.40±0.84	**78.89±1.05**	77.59±1.93	70.52±0.16	61.34±0.50	68.50±1.28	66.57±2.61
		LED	64.55±0.50	64.34±0.47	63.75±0.50	**64.95±0.33**	38.27±0.73	46.85±0.36	61.71±0.41	35.39±3.56
		Mixed	87.44±0.21	**88.20±0.22**	87.45±0.21	87.88±0.23	60.99±0.21	63.45±0.37	87.69±0.26	57.00±0.74
		Sine	84.55±0.59	85.43±0.34	85.00±0.27	**85.48±0.32**	62.94±0.23	61.79±0.55	84.81±0.40	60.20 ±0.86
		Waveform	77.59±0.45	78.03±0.39	78.34±0.38	**78.66±0.29**	58.01±0.94	76.28±0.39	77.72±0.42	58.37 ±2.46
		RandomRBF	31.42±0.49	31.40±0.52	30.38±0.52	31.07±0.46	21.82±0.26	28.90±1.20	**33.92±0.52**	26.52±1.38
	50K	Agrawal (F1-F5)	61.50±0.73	63.32±0.40	63.58±0.34	63.67±0.37	53.19±0.12	59.85±0.63	**64.07±0.34**	54.24±1.05
		Agrawal (F6-F10)	77.56±1.96	82.39±0.59	**83.63±0.52**	82.77±0.97	71.37±0.15	64.36±0.37	71.51±0.58	70.37±0.80
		LED	69.63±0.18	69.68±0.18	69.43±0.28	**69.81±0.16**	38.81±0.68	50.01±0.91	67.98±0.36	36.81±5.22
		Mixed	89.95±0.59	**90.74±0.22**	90.62±0.19	90.69±0.20	61.23±0.16	65.00±0.54	89.76±0.21	63.50±0.91
		Sine	86.92±0.55	**88.33±0.15**	88.23±0.14	87.92±0.17	62.97±0.12	62.65±0.43	86.50±0.29	66.91±0.97
		Waveform	78.99±0.25	79.17±0.21	79.25±0.26	**79.58±0.22**	57.84±0.62	77.02±0.27	79.09±0.25	62.43±1.57
		RandomRBF	31.11±0.51	31.59±0.39	30.79±0.48	31.38±0.38	21.67±0.14	29.85±1.85	**35.88±0.50**	25.00±1.36
	100K	Agrawal (F1-F5)	65.25±0.86	66.66±0.45	66.09±0.66	**66.89±0.40**	53.52±0.17	61.60±0.44	66.36±0.31	55.55±0.66
		Agrawal (F6-F10)	82.62±0.51	84.62±0.30	**84.98±0.24**	84.51±0.37	71.62±0.09	65.79±0.32	73.09±0.30	72.78±1.20
		LED	71.09±0.20	71.18±0.15	71.06±0.30	**71.57±0.14**	38.37±0.36	50.62±0.83	70.59±0.20	27.41±4.39
		Mixed	90.47±0.21	91.06±0.10	**91.06±0.09**	90.96±0.11	61.20±0.11	65.98±0.65	90.76±0.17	71.13±1.08
		Sine	88.71±0.19	**89.50±0.13**	89.49±0.11	89.04±0.16	63.04±0.11	63.77±0.54	88.16±0.21	75.33±0.64
		Waveform	79.07±0.18	79.66±0.18	79.66±0.15	**79.71±0.12**	58.40±0.44	77.42±0.18	79.52±0.16	61.34±1.41
		RandomRBF	31.86±0.37	32.02±0.34	31.05±0.23	31.86±0.32	21.81±0.10	30.89±1.54	**36.91±0.39**	23.12±0.74
30%	20K	Agrawal (F1-F5)	61.05±0.46	62.44±0.47	62.76±0.37	62.81±0.38	59.99±0.26	59.18±0.25	**63.94±0.29**	56.22±1.05
		Agrawal (F6-F10)	72.66±1.80	80.05±0.43	**81.54±0.51**	80.64±1.04	78.55±0.54	63.19±0.31	69.56±0.98	72.58±1.19
		LED	68.25±0.42	68.27±0.27	67.98±0.35	**68.47±0.23**	55.79±0.28	50.59±0.44	61.85±0.49	51.16±5.63
		Mixed	89.22±0.22	89.17±1.17	89.51±0.21	**89.65±0.22**	58.97±0.86	65.32±0.53	87.80±0.27	73.83±1.13
		Sine	86.19±0.54	**87.22±0.18**	87.07±0.16	87.08±0.19	68.60±1.12	64.13±0.37	85.22±0.32	77.80±0.83
		Waveform	78.48±0.34	78.71±0.30	78.58±0.44	**79.28±0.29**	76.74±0.22	76.96±0.28	78.83±0.32	74.73±0.77
		RandomRBF	31.14±0.44	**31.58±0.45**	31.13±0.46	31.37±0.40	30.02±0.58	28.95±1.34	31.23±0.51	27.07±1.05
	50K	Agrawal (F1-F5)	64.36±1.11	66.83±0.42	66.08±0.48	**66.87±0.40**	61.90±0.31	61.19±0.21	66.28±0.27	59.78±0.69
		Agrawal (F6-F10)	82.23±0.89	84.43±0.21	**84.50±0.29**	84.24±0.30	79.87±0.30	66.39±0.40	72.61±0.37	81.07±0.87
		LED	71.16±0.16	71.26±0.18	70.90±0.27	**71.46±0.15**	57.47±0.73	53.00±1.15	63.26±0.30	60.08±5.49
		Mixed	90.31±0.36	**90.99±0.13**	90.93±0.11	90.87±0.12	67.36±1.27	65.56±0.42	88.55±0.21	85.29±0.58
		Sine	88.74±0.23	**89.41±0.13**	89.37±0.12	88.88±0.16	82.30±0.60	64.03±0.45	86.06±0.16	85.61±0.33
		Waveform	79.03±0.25	79.53±0.22	79.58±0.22	**79.76±0.20**	78.11±0.14	77.44±0.24	79.49±0.28	76.57±0.38
		RandomRBF	31.57±0.38	**32.17±0.28**	31.13±0.30	31.88±0.27	30.80±0.49	30.71±1.45	31.45±0.29	26.05±1.08
	100K	Agrawal (F1-F5)	67.16±1.49	**70.27±0.34**	69.23±0.55	70.26±0.27	62.21±0.16	59.55±0.15	68.30±0.27	60.40±0.66
		Agrawal (F6-F10)	83.20±0.92	85.30±0.37	**86.08±0.16**	85.39±0.38	81.29±0.17	66.08±0.59	73.41±0.39	83.19±0.28
		LED	72.01±0.40	72.54±0.14	71.91±0.20	**72.66±0.13**	63.65±0.19	54.71±0.87	63.48±0.25	64.66±4.90
		Mixed	90.38±0.41	91.25±0.09	**91.26±0.08**	91.08±0.09	80.95±0.35	67.74±0.94	88.84±0.14	88.63±0.38
		Sine	90.12±0.08	90.64±0.09	**90.62±0.08**	90.25±0.15	83.25±0.36	64.52±0.45	86.76±0.14	87.65±0.19
		Waveform	79.29±0.20	79.69±0.17	79.81±0.15	**79.88±0.12**	79.05±0.10	78.39±0.19	79.83±0.15	77.00±0.20
		RandomRBF	32.16±0.39	**32.46±0.22**	31.13±0.16	32.35±0.21	32.27±0.28	30.66±1.32	31.69±0.24	25.09±1.04
		Rank	4.11	2.17	2.95	**1.96**	6.76	7.02	4.14	6.88

To perform the statistical analysis, initially the overall ranks were calculated for all experiment setup. In absolute terms, we can observe that, in the artificial datasets, RDDM, HDDM$_A$ and FHDDM were the three best-ranked methods, whereas BDF_RDDM was the best semi-supervised algorithm. On the other hand, in the real-world datasets, BDF_RDDM, RDDM, and FHDDM were the best three methods in this order.

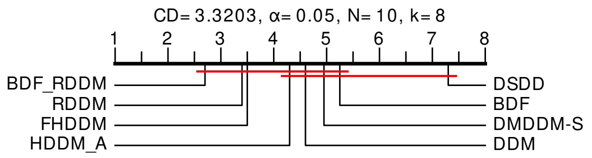
The *Nemenyi* posthoc test results are graphically displayed using diagrams where the Critical Difference (CD) is represented by bars connecting statistically similar methods and a horizontal ruler marks the ranks of all the analyzed methods.

Figure 1a represents the results in the datasets with abrupt drifts. RDDM was the best method, though without statistical difference to HDDM$_A$ and FHDDM. It is worth pointing out that the results of the gradual datasets and those separated by percentage of labels were omitted, but they are quite similar to those already presented, with only minor differences.

Finally, Figure 1b captures the results in the real-world datasets. In this evaluation, BDF_RDDM, RDDM, and FHDDM were all significantly superior to DSDD, being statistically similar to the other tested methods.



(a)



(b)

Fig. 1. Accuracy statistical comparison of all methods using F_F and the *Nemenyi* posthoc test on (a) datasets with abrupt drifts and partial labels (15% and 30%), with 95% confidence, and (b) real-world datasets.

TABLE II
MEAN ACCURACIES IN PERCENTAGE (%) WITH 95% CONFIDENCE AND HT BASE CLASSIFIER IN ARTIFICIAL DATASETS WITH GRADUAL DRIFTS

%Labels	SIZE	DATASET	DDM	HDDM$_A$	FHDDM	RDDM	DMDDM-S	BDF	BDF_RDDM	DSDD
		Agrawal (F1-F5)	59.31±0.30	60.13±0.30	60.43±0.35	60.62±0.37	53.40±0.20	57.95±0.41	**61.92±0.31**	51.82±1.14
		Agrawal (F6-F10)	70.76±1.69	74.98±1.32	**77.26±1.18**	76.15±2.01	69.98±0.18	61.33±0.47	67.47±1.04	66.60±2.63
		LED	63.77±0.58	63.29±0.41	63.22±0.45	**64.13±0.38**	37.21±0.47	47.26±0.52	60.88±0.38	32.98±3.61
	20K	Mixed	85.98±0.16	86.14±0.16	85.99±0.15	**86.23±0.16**	60.14±0.19	63.80±0.31	85.65±0.30	56.81±0.70
		Sine	83.45±0.47	83.88±0.24	83.68±0.25	**83.99±0.25**	62.03±0.17	62.08±0.56	83.31±0.33	59.07±0.79
		Waveform	77.15±0.39	77.48±0.45	77.54±0.39	**78.15±0.33**	57.79±0.87	75.94±0.36	77.22±0.38	58.49±2.52
		RandomRBF	31.18±0.51	31.24±0.56	30.28±0.62	31.14±0.49	21.66±0.16	28.86±1.31	**33.87±0.69**	26.47±1.38
		Agrawal (F1-F5)	61.20±0.72	62.86±0.46	63.33±0.36	63.66±0.40	53.21±0.11	59.73±0.60	**64.04±0.32**	54.10±0.87
		Agrawal (F6-F10)	76.30±1.86	81.63±0.50	**82.32±0.56**	81.62±1.37	71.17±0.12	64.40±0.31	70.98±0.66	70.33±0.98
		LED	69.50±0.19	69.34±0.21	69.12±0.21	**69.66±0.19**	38.08±0.43	49.68±0.69	67.36±0.36	32.00±4.81
15%	50K	Mixed	89.39±0.16	**89.52±0.13**	89.49±0.13	89.46±0.14	60.77±0.14	64.86±0.47	88.77±0.21	62.55±1.04
		Sine	87.06±0.20	**87.39±0.15**	87.29±0.14	87.24±0.16	62.57±0.13	62.89±0.37	86.05±0.20	66.81±1.13
		Waveform	78.83±0.24	78.86±0.15	78.96±0.27	**79.38±0.22**	58.32±0.67	77.13±0.27	78.93±0.27	61.27±1.98
		RandomRBF	31.63±0.42	31.81±0.38	31.14±0.43	31.61±0.36	21.70±0.15	29.11±1.80	**36.09±0.44**	25.62±1.05
		Agrawal (F1-F5)	64.29±1.03	66.62±0.48	65.78±0.54	**66.80±0.42**	53.48±0.15	61.60±0.48	66.15±0.35	55.89±0.69
		Agrawal (F6-F10)	82.69±0.39	**84.50±0.25**	84.70±0.24	84.52±0.25	71.52±0.07	65.94±0.37	73.02±0.58	72.24±0.89
		LED	71.29±0.15	71.25±0.14	70.88±0.18	**71.63±0.14**	38.43±0.32	50.72±0.92	70.37±0.16	25.38±3.81
	100K	Mixed	90.46±0.15	90.71±0.07	**90.72±0.07**	90.69±0.07	61.00±0.10	65.10±0.31	90.44±0.15	71.44±1.07
		Sine	88.72±0.11	**89.12±0.09**	89.11±0.08	88.93±0.11	62.83±0.09	63.70±0.56	87.71±0.21	75.74±0.77
		Waveform	79.04±0.21	79.52±0.20	79.50±0.17	**79.83±0.14**	58.33±0.58	77.43±0.18	79.49±0.15	61.74±1.15
		RandomRBF	31.50±0.47	31.99±0.29	31.10±0.23	31.80±0.31	21.67±0.09	30.65±1.49	**36.73±0.34**	23.48±0.92
		Agrawal (F1-F5)	60.57±0.36	61.52±0.36	62.30±0.34	62.27±0.41	59.90±0.27	58.96±0.27	**63.41±0.30**	56.22±1.15
		Agrawal (F6-F10)	71.69±1.81	78.94±0.72	**79.23±0.75**	79.12±1.40	78.42±0.47	63.03±0.32	68.69±1.01	71.90±1.30
		LED	67.81±0.26	67.30±0.30	66.88±0.36	**67.85±0.22**	55.92±0.26	50.54±0.39	60.92±0.41	53.56±5.16
	20K	Mixed	86.93±0.22	87.12±0.19	**87.15±0.18**	86.94±0.20	59.28±0.85	65.05±0.44	85.85±0.29	70.70±0.99
		Sine	84.86±0.26	85.18±0.21	**85.19±0.20**	85.00±0.20	68.12±1.43	63.87±0.39	83.45±0.29	77.17±0.78
		Waveform	78.11±0.30	78.10±0.33	78.27±0.30	**78.60±0.26**	76.74±0.23	76.73±0.31	78.10±0.34	74.53±0.97
		RandomRBF	31.35±0.47	**31.73±0.43**	30.63±0.40	31.31±0.39	30.22±0.52	29.01±1.35	31.28±0.49	27.06±1.09
		Agrawal (F1-F5)	64.22±1.06	**66.48±0.37**	65.99±0.47	66.39±0.35	61.64±0.30	61.12±0.18	65.82±0.23	59.61±0.42
		Agrawal (F6-F10)	81.94±0.98	84.04±0.34	**84.07±0.27**	83.98±0.31	79.93±0.29	66.19±0.50	72.52±0.39	81.05±0.80
		LED	71.02±0.17	70.91±0.17	70.51±0.24	**71.27±0.15**	56.87±0.46	53.01±0.96	63.06±0.32	58.62±5.98
30%	50K	Mixed	89.75±0.14	**89.92±0.10**	**89.92±0.09**	89.83±0.12	66.97±0.95	65.38±0.33	87.74±0.17	84.84±0.52
		Sine	88.24±0.21	88.55±0.12	**88.57±0.11**	88.39±0.14	81.69±0.49	63.97±0.42	85.57±0.22	85.26±0.29
		Waveform	78.90±0.21	79.27±0.22	79.18±0.17	**79.55±0.19**	78.09±0.12	77.47±0.22	79.37±0.24	76.49±0.39
		RandomRBF	31.59±0.44	**32.03±0.28**	31.10±0.28	31.95±0.32	30.67±0.56	31.43±1.31	31.42±0.26	25.95±0.95
		Agrawal (F1-F5)	67.35±1.55	69.91±0.34	68.93±0.58	**70.12±0.32**	62.28±0.19	59.43±0.14	68.19±0.27	61.16±0.85
		Agrawal (F6-F10)	83.39±0.79	85.00±0.31	**85.86±0.13**	85.21±0.40	81.11±0.17	66.00±0.54	73.31±0.49	82.91±0.24
		LED	72.13±0.25	72.28±0.15	71.75±0.17	**72.60±0.13**	63.71±0.18	55.08±0.95	63.23±0.22	61.64±5.46
	100K	Mixed	90.40±0.25	90.72±0.07	**90.73±0.07**	90.67±0.07	81.05±0.37	66.91±0.58	88.56±0.15	88.09±0.34
		Sine	90.09±0.10	90.20±0.09	**90.23±0.09**	90.10±0.12	82.98±0.36	64.61±0.54	86.50±0.16	87.50±0.27
		Waveform	79.46±0.18	79.68±0.18	79.76±0.16	**79.90±0.12**	79.05±0.10	78.39±0.15	79.85±0.13	76.89±0.25
		RandomRBF	31.97±0.32	**32.27±0.22**	31.05±0.17	32.12±0.20	32.18±0.31	30.76±1.36	31.73±0.27	24.88 ±1.15
		Rank	3.90	2.38	2.75	**2.10**	6.74	6.93	4.30	6.90

VI. CONCLUSIONS

In this research, extensive experimentation was carried out to investigate the possibility of using *supervised* concept drift detectors in *semi-supervised* learning environments, mainly driven by the idea of executing the drift detector only on the labeled instances.

To enable this idea in practice and also be able to correctly evaluate the results, it was necessary to add new functionalities to the MOA framework. The implemented features permit hiding from the methods the labels of a subset of the instances of fully-labeled datasets, based on configurable parameters under the control of the user.

All the experiments used Hoeffding Tree (HT) as base learner, combined with four supervised concept drift detectors and three semi-supervised methods, using a reasonably large number of artificial and real-world datasets, configured with 15%, 30% and 100% of instances with labels. In the results, the performances of the supervised methods RDDM, HDDM$_A$ and FHDDM were better in the datasets with both abrupt and gradual drifts, in all tested configurations. As for the semi-supervised methods, the best results were those of BFD_RDDM.

The reported results allow us to claim that the use of supervised detectors in semi-supervised environments is possible, can deliver excellent results, and might mark a paradigm change for future research in the area.

In the future, this research can be extended with experiments in other semi-supervised scenarios testing other competitive supervised classifiers and detectors, also including the use of ensembles of classifiers such as Boosting-like Online Learning Ensemble (BOLE) [31], Fast Adaptive Stacking of Ensembles (FASE) [32], Online AdaBoost-based M1 (OABM1) [33], etc., combined with concept drift detectors such as RDDM. Another possible way to proceed is to adapt supervised concept drift detectors in other existing semi-supervised algorithms, similar to how it was done in this research with BDF and RDDM and proved promising.

TABLE III
MEAN ACCURACIES IN PERCENTAGE (%) WITH 95% CONFIDENCE INTERVALS AND HT BASE CLASSIFIER IN REAL-WORLD DATASETS

%Labels	DATASET	DDM	HDDM$_A$	FHDDM	RDDM	DMDDM-S	BDF	BDF_RDDM	DSDD
15%	Airlines	63.18	63.87	65.01	64.77	65.13	64.10	**66.35**	58.74
	Connect4	70.09	70.52	70.43	70.77	69.81	70.57	**73.55**	65.87
	Covertype	75.50	**78.50**	78.13	78.42	73.72	73.21	78.01	66.26
	Spam_data	**90.27**	89.64	89.23	89.57	89.85	84.54	89.99	76.67
	Weather	68.69	68.72	68.81	68.88	69.09	**69.94**	68.94	67.92
30%	Airlines	63.69	64.59	65.38	64.82	65.18	67.56	**68.24**	63.17
	Connect4	71.64	72.29	72.38	72.46	70.08	70.08	**73.58**	69.25
	Covertype	78.50	79.98	**80.41**	80.24	74.19	59.74	68.07	78.76
	Spam_data	**91.01**	90.80	90.53	90.47	90.04	83.49	88.93	88.36
	Weather	70.29	69.63	70.05	69.99	69.23	69.94	**70.35**	69.76
	Rank	4.60	4.30	3.50	3.40	4.95	5.25	**2.70**	7.30

ACKNOWLEDGMENT

Roberto S. M. Barros is supported by research grant number 305539/2022-1 from Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq). Silas Santos was previously supported by a post-doctorate grant from CAPES, Brazil.

DISCLAIMER

The views and opinions expressed in this research are solely those of the authors and do not necessarily reflect the official policy or position of the company.

REFERENCES

[1] M. H. Le Nguyen, H. M. Gomes, and A. Bifet, "Semi-supervised learning over streaming data using MOA," in *IEEE International Conference on Big Data*, 2019, pp. 553–562.

[2] T. T. Lu, "Fundamental limitations of semi-supervised learning," Master's thesis, University of Waterloo, 2009. [Online]. Available: http://hdl.handle.net/10012/4387

[3] V. Macário, "Um novo algoritmo de agrupamento semi-supervisionado baseado no fuzzy c-means," Master's thesis, Universidade Federal de Pernambuco, 2009, in Portuguese.

[4] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive online analysis," *Jour. Mach. Learn. Res.*, vol. 11, pp. 1601–1604, 2010.

[5] Z. Ahmadi and H. Beigy, "Semi-supervised ensemble learning of data streams in the presence of concept drift," in *Hybrid Artificial Intelligent Systems (HAIS)*, ser. LNCS. Springer, 2012, vol. 7209, pp. 526–537.

[6] A. Haque, L. Khan, and M. Baron, "SAND: Semi-supervised adaptive novel class detection and classification over data stream," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016, pp. 1652–1658.

[7] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl *et al.*, "Constrained k-means clustering with background knowledge," in *Proceedings of 18th Intern. Conf. on Machine Learning (ICML)*, vol. 1, 2001, pp. 577–584.

[8] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised clustering by seeding," in *Proceedings of 19th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2002, pp. 19–26.

[9] P. Li, X. Wu, and X. Hu, "Mining recurring concept drifts with limited labeled streaming data," in *Proceedings of 2nd Asian Conference on Machine Learning*, 2010, pp. 241–252.

[10] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. of 5th Berkeley symp. on mathematical statistics and probab.*, vol. 1. Oakland, CA, USA., 1967, pp. 281–297.

[11] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the 11th ACM annual conference on Computational learning theory (COLT'98)*, 1998, pp. 92–100.

[12] P. Monteiro, E. Soares, and R. S. M. Barros, "Co-op training: a semi-supervised learning method for data streams," in *Proc. of IEEE Intern. Conf. on Syst., Man and Cybern. (SMC)*, Melbourne, 2021, pp. 933–938.

[13] Y. Zhou and S. Goldman, "Democratic co-learning," in *Proc. 16th IEEE Intern. Conf. on Tools with Artif. Intellig. (ICTAI)*, 2004, pp. 594–602.

[14] G. Ditzler and R. Polikar, "Semi-supervised learning in nonstationary environments," in *Proceedings of IEEE International Joint Conference on Neural Networks (IJCNN)*, San Jose, CA, USA, 2011, pp. 2741–2748.

[15] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence: SBIA 2004*, ser. LNCS. Springer, 2004, vol. 3171, pp. 286–295.

[16] I. Frías-Blanco, J. del Campo-Ávila, G. Ramos-Jiménez, R. Morales-Bueno, A. Ortiz-Díaz, and Y. Caballero-Mota, "Online and non-parametric drift detection methods based on hoeffding's bounds," *IEEE Trans. on Knowledge and Data Eng.*, vol. 27, no. 3, pp. 810–823, 2015.

[17] G. Ross, N. Adams, D. Tasoulis, and D. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191–198, 2012.

[18] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Jour. Amer. stat. assoc.*, vol. 58, no. 301, pp. 13–30, 1963.

[19] A. Pesaranghader and H. Viktor, "Fast Hoeffding drift detection method for evolving data streams," in *Machine Learning and Knowledge Discovery in Databases*, ser. LNCS, vol. 9852. Springer, 2016, pp. 96–111.

[20] R. S. M. Barros, D. R. L. Cabral, P. M. Gonçalves Jr, and S. G. T. C. Santos, "RDDM: Reactive drift detection method," *Expert Systems with Applications*, vol. 90, pp. 344–355, 2017.

[21] A. Haque, L. Khan, M. Baron, B. Thuraisingham, and C. Aggarwal, "Efficient handling of concept drift and concept evolution over stream data," in *IEEE 32nd Intern. Conference on Data Engineering (ICDE)*, 2016, pp. 481–492.

[22] O. A. Mahdi, E. Pardede, N. Ali, and J. Cao, "Fast reaction to sudden concept drift in the absence of class labels," *Applied Sciences*, vol. 10, no. 2, pp. 606:1–16, 2020.

[23] F. Pinagé, E. M. dos Santos, and J. Gama, "A drift detection method based on dynamic classifier selection," *Data Mining and Knowledge Discovery*, vol. 34, no. 1, pp. 50–74, 2020.

[24] R. N. Gemaque, A. F. J. Costa, R. Giusti, and E. M. Dos Santos, "An overview of unsupervised drift detection methods," *Data Mining and Knowledge Discovery*, vol. 10, no. 6, pp. e1381:1–18, 2020.

[25] M. Baena-Garcia, J. Del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method," in *Proc. of 4th Intern. Work. Knowledge Discov. from Data Streams*, 2006, pp. 77–86.

[26] B. I. F. Maciel, S. G. T. C. Santos, and R. S. M. Barros, "MOAManager: a tool to support data stream experiments," *Software: Practice and Experience*, vol. 50, no. 4, pp. 325–334, 2020.

[27] R. S. M. Barros and S. G. T. C. Santos, "A large-scale comparison of concept drift detectors," *Inform. Sciences*, vol. 451, pp. 348–370, 2018.

[28] J. I. G. Hidalgo, B. I. F. Maciel, and R. S. M. Barros, "Experimenting with prequential variations for data stream learning evaluation," *Computational Intelligence*, vol. 35, pp. 670–692, 2019.

[29] D. Ienco, A. Bifet, I. Žliobaitė, and B. Pfahringer, "Clustering based active learning for evolving data streams," in *Discovery Science*, ser. LNCS, vol. 8140. Springer, 2013, pp. 79–93.

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

[31] R. S. M. Barros, S. G. T. C. Santos, and P. M. Gonçalves Jr., "A boosting-like online learning ensemble," in *Proc. of IEEE Internat. Joint Conf. on Neural Networks (IJCNN)*, Vancouver, Canada, 2016, pp. 1871–1878.

[32] I. Frías-Blanco, A. Verdecia-Cabrera, A. Ortiz-Díaz, and A. Carvalho, "Fast adaptive stacking of ensembles," in *Proceedings of the 31st ACM Symposium on Applied Computing (SAC)*, Pisa, Italy, 2016, pp. 929–934.

[33] S. G. Santos and R. S. M. Barros, "Online adaboost-based methods for multiclass problems," *Artif. Intellig. Rev.*, vol. 53, pp. 1293–1322, 2020.