A Drift Detection Method Based on Active Learning

Albert França Josuá Costa
Federal Institute of Amazonas
Humaitá, Brazil
albert.costa@ifam.edu.br

Régis Antônio Saraiva Albuquerque
Institute Computing - ICOMP
Federal University of Amazonas
Manaus, Brazil
regis.albuquerque1@gmail.com

Eulanda Miranda dos Santos
Institute Computing - ICOMP
Federal Univeristy of Amazonas
Manaus, Brazil
emsantos@icomp.ufam.edu.br

***Abstract*—Several real-world prediction problems are subject to changes over time due to their dynamic nature. These changes, named concept drift, usually lead to immediate and disastrous loss in classifier's performance. In order to cope with such a serious problem, drift detection methods have been proposed in the literature. However, current methods cannot be widely used since they are based either on performance monitoring or on fully labeled data, or even both. Focusing on overcoming these drawbacks, in this work we propose using density variation of the most significant instances as an explicit unsupervised trigger for concept drift detection. Here, density variation is based on Active Learning, and it is calculated from virtual margins projected onto the input space according to classifier confidence. In order to investigate the performance of the proposed method, we have carried out experiments on six databases, precisely four synthetic and two real databases focusing on setting up all parameters involved in our method and on comparing it to three baselines, including two supervised drift detectors and one Active Learning-based strategy. The obtained results show that our method, when compared to the supervised baselines, reached better recognition rates in the majority of the investigated databases, while keeping similar or higher detection rates. In terms of the Active Learning-based strategies comparison, our method outperformed the baseline taking into account both recognition and detection rates, even though the baseline employed much less labeled samples. Therefore, the proposed method established a better trade-off between amount of labeled samples and detection capability, as well as recognition rate.**

Keywords — concept drift detection, active learning, virtual margins, uncertainty;

I. Introduction

Traditional machine learning methods address prediction problems based on two assumptions: (1) representative labeled data is abundantly available to describe the problems; and (2) there is no time-changing data involved. However, these assumptions are not realistic in many real-world problems. In terms of the first assumption, it is very common that only a limited amount of labeled instances is gathered, due to the extremely high labeling cost and since the labeling process is generally a human efforts-dependent task. On the other hand, in several problems, there is almost unlimited amount of unlabeled instances, especially when dealing with data stream applications.

In terms of time-changing data, this kind of data may be observed in different domains, since data can vary over time due to their dynamic generation process. In the literature, this phenomenon is named concept drift. The occurrence of concept drift generates immediate and disastrous loss in classifier's performance. In order to cope with drifts, many methods have been proposed, which may be divided into two groups: blind and informed methods [1].

Blind methods do not provide an explicit mechanism to detect concept drift. Instead, they try to handle this problem by regularly retraining the classifiers, even if drifts do not occur. In this case, there is usually high computational cost involved on retraining classifiers unnecessarily. In contrast to blind methods, the second group of drift detection strategies uses an explicit mechanism to accomplish the task. Nevertheless, most of these methods are based on performance monitoring. Thus, there may be a drawback to these approaches, since a significant performance decrease is necessary to allow the system to detect drift, even though this may reduce the predictor effectiveness.

The most widely known informed drift detector is called Drift Detection Method (DDM), proposed in [2]. This method monitors the evolution of the classifier's error rate based on the Probably Approximately Correct (PAC) learning model. This model assumes that a significant error rate increase may indicate the occurrence of concept drift. The authors point out that this method attains the best results when detecting sudden drift. In order to improve DDM's ability to detect gradual drift, the Early Drift Detection Method (EDDM), based on DDM, was proposed in [3]. The basic idea behind this method is to monitor the distance between two consecutive error rates generated by a classifier, instead of monitoring the variation of the error rate directly. A decrease on this distance may indicate the occurrence of concept drift, or a stationary environment otherwise.

DDM and EDDM are both incremental learning-based methods. Thus, each new instance is used to update the decision hypothesis. As a consequence, these methods expect instances true labels fully and immediately available after classification. However, there is no guarantee on this assumption, especially when data streams are involved, since labeled data streams are hardly readily available. In addition, both methods are also error monitoring-based, thus, they are supposed to detect drift only after a significant error rate increase. Taking into account these drawbacks and focusing on overcoming them, the Dissimilarity-based Drift Detection Method (DbDDM) was proposed in [4]. DbDDM performs drift detection by monitoring the dissimilarity between the distributions of past and current instances, which are grouped into clusters. Each new unknown instance is classified according to its closest reference cluster. Hence, this method does not depend on decreasing the classifier's performance to detect drift. Even though, it is necessary to update the reference

State Funding Agency of Amazonas (FAPEAM) through public call 002/2014 - RH-TI Capacitação/Amazônia Turmas Fora de Sede.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

clusters with all new instances. Thus, to perform cluster update, the true labels are immediately required after classification. Therefore, fully labeled data are also necessary in this method.

It is interesting to observe that all three methods previously mentioned ([2], [3] and [4]) are based on the assumption that as soon as a class is assigned to an instance, its true label is immediately available. In order to avoid this no widely guaranteed assumption, in this paper a new drift detector is proposed, which employs an Active Learning strategy to explicitly detect drifts in an unsupervised way. Here, instances' true labels are required only after drift detection. In addition, the new-labeled instances are used only to update the classifier, not to detect drifts. Moreover, as a consequence of Active Learning, the number of instances used to retrain the classifier after each detected drift is reduced.

There are few papers in the literature using Active Learning to handle drift, regardless whether or not the detection is explicit. Among these methods, the Margin Density Drift Detection (MD3) proposed by Sethi and Kantardzic [5] is a SVM margin density variation-based method. MD3 assumes that variation's density higher than a certain drift threshold indicates drift. The main drawback to MD3 is the fact that this method works only with SVM. Based on [5], a new version of MD3 was proposed in [6], where an ensemble of classifiers is used, instead of a single classifier. This new version first divides the input space into K subspaces. Then, each subspace is used to train one classifier. The drift detection process takes place by measuring diversity among classifier members. If ensemble's disagreement increases when assigning classes to the unknown samples, this behavior may indicate a drift. In this way, it is possible to produce a generic Margin Density metric, avoiding the limitation observed in [5] and allowing MD3 to be applied with a large variety of learning methods.

According to the authors, since this new version of MD3 [6] may achieve high false positive rates, a drift confirmation module is added to the method. This module labels the N latest instances (current window) and compare the ensemble's accuracy between current window and a reference window. If a reduction on ensemble's accuracy is observed on current window, a drift is assigned. Thus, the main drawback to this method relies on the fact that the drift confirmation module leads MD3 to be a genuine supervised method.

The method proposed in this paper, called Drift Detection Method Based on Active Learning (DDAL), is also margin density-based and may work with any learning algorithm. However, different from the MD3 new version presented in [6], we propose an informed and unsupervised drift detection method, which applies Active Learning, namely the Fixed Uncertainty strategy. This strategy is used to select the most significant instances employing a concept of virtual margins, which is also adapted in this work. Thus, DDAL measures the variation density of the most significant instances to indicate whether or not a drift is observed.

The paper is organized as follows. The proposed method is explained in Section 2. Experiments and results are discussed in Section 3. Finally, in Section 4, conclusions and suggestions for future work are exposed.

II. THE PROPOSED METHOD

Our method is based on the hypothesis that variation's densities of the most significant instances selected by Active Learning may indicate drift. In this way, we try to avoid detecting drift based on classifier's performance or on fully labeled data. The proposed DDAL method is a batch-based approach that updates the classifier only after drift detection. Therefore, DDAL is an informed but unsupervised method.

DDAL is divided into two phases. The first phase involves the task of generating a classifier, while the second is concerned with three modules: drift detection, reaction, and classification, as shown in Figure 1. Initially, a classifier is generated using instances contained in a first batch, named training batch. In the second phase, our method monitors the density of the instances selected by Active Learning in order to detect the occurrence of concept drift.

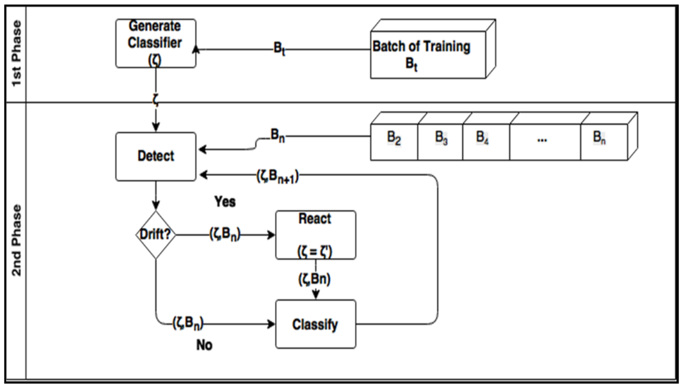

Fig. 1. Overview of DDAL. The method is divided into two phases. The first phase occurs just once, since it is devoted to classifier generation.

After generating a classifier (ζ), the first phase is completed. Then, the second phase starts on conducting the detection module and, optionally, the reaction and the classification modules, for each new batch of unlabeled instances. At the detection module, the method calculates the current density (δ) of the most significant instances selected using the concept of virtual margins projected by Fixed Uncertainty [7], which is an Active Learning strategy and it is described in the next section. Then, δ is compared to the historical maximum and minimum density values (δ_{max}, δ_{min}) and it replaces the historical values when it is greater than δ_{max} or lower than δ_{min}. Lastly, if the difference between δ_{max} and δ_{min} is greater than a drift threshold (θ), then a drift is signaled and the reaction module is triggered. Otherwise, classification module is performed.

At the reaction module, a new classifier (ζ') is generated based on the current batch of instances (B_n) in replacement to the current classifier (ζ). Only at this point the instances' true labels are required, since a new training set must be provided. On the other hand, at the classification module, samples of the current batch are classified and DDAL returns to the detection module to deal with the next batch of unlabeled instances. All these modules are detailed in the next sections. First, however, it is necessary to explain the concept of virtual margins adopted in this work.

A. *Virtual Margins*

The main contribution of this work is to use an Active Learning strategy to detect concept drift. Among several Active Learning techniques available in the literature, the Fixed Uncertainty strategy appears to be interesting due to the successful results attained in problems involving instance selection [7]. In order to use the Fixed Uncertainty strategy to select samples to detect concept drift, it is necessary to define the virtual margins concept. In this paper, virtual margins are interpreted as the projection of hyperplanes equidistant to the separating hyperplane based on a user defined uncertainty threshold (λ). Figure 2 shows virtual margins defined for λ = 0.95 and λ = 0.90. This figure shows linear virtual margins projections, however, we believe that non-linear virtual margins may also be attained, but a more complete research is necessary.

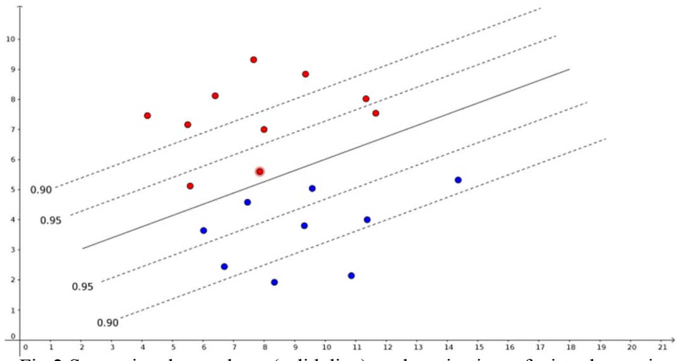

Fig.2.Separating hyperplane (solid line) and projection of virtual margins (dotted lines) with uncertainty threshold set to 0.95 and 0.90.

The projection of virtual margins creates a subspace of the input space where classifier's confidence is lower than λ. Here, confidence is defined as the classifier maximum posterior probability when performing a prediction. The higher the confidence value, the farther the instance will be from the separating hyperplane, and likely out of the subspace delimited by virtual margins. By contrast, when confidence value is lower than λ, the instance will be closer to the separating hyperplane and inside the subspace delimited by virtual margins.

Algorithm 1 summarizes the Fixed Uncertainty Active Learning strategy. In line 2, classifier's confidence is calculated for each sample x_i. This value is then compared to the uncertainty threshold. The sample x_i is assumed to be inside the virtual margins when the confidence value is lower than the threshold, or outside otherwise.

```
Algorithm 1:Fixed Uncertainty ($x_i$,$\lambda$,$\zeta$)
Output: significant $\in$ {true,false}
01. significant $\leftarrow$ false
02. maxposteriori $\leftarrow$ $\zeta(x_i)$
03. if(maxposteriori<$\lambda$)
04.     significant $\leftarrow$ true
05. endif
06. return significant
```

There are few work in the literature using Fixed Uncertainty. For instance, in [7], the authors use this strategy to select the most relevant samples to be labeled in order to avoid labeling the whole dataset, but no concept drift was investigated. The novelty in this paper is to adapt the Fixed Uncertainty strategy to select samples to detect concept drift by means of virtual margins. It is important to note that in [5] and [8], MD3 uses SVM's margins to detect concept drift, however, it is a method-dependent strategy, because it is designed to work only with SVM. In addition, as mentioned in previous section, a generic MD3 margin density metric based on ensemble diversity is proposed in [6], but this new MD3 version needs labeled samples to cope with drift. In our work, the virtual margin concept used allows the proposed method to be integrated with any learning method, as long as a classifier confidence is provided. The details of the second phase of DDAL are presented below.

B. *DDAL Algorithm Detection Module*

Once the virtual margins are defined, it is possible to describe how this concept is used to detect drift. This process is conducted by calculating the density of the most significant samples contained in each batch of unlabeled data selected by Fixed Uncertainty. Our hypothesis relies on the idea that density variation higher than a user defined drift threshold indicates drift. The detection module of DDAL is summarized in Algorithm 2.

```
Algorithm 2:DDAL – Detection Module ($B_n$, $\delta_{min}$, $\delta_{max}$, $\theta$, $\lambda$, $\zeta$)
Output: drift $\in$ {true,false}
01. while $x_i$ in $B_n$ do
02.     if (Fixed Uncertainty($x_i$,$\lambda$,$\zeta$)) then
03.         $\delta_{current} \leftarrow \delta_{current} + 1$
04.     endif
05. endwhile
06. $\delta_{current} \leftarrow \delta_{current}$ /size($B_n$)
07. if ( $\delta_{current} < \delta_{min}$ ) then
08.     $\delta_{min} \leftarrow \delta_{current}$
09. endif
10. if ( $\delta_{current} > \delta_{max}$ ) then
11.     $\delta_{max} \leftarrow \delta_{current}$
12. endif
13. if( ( $\delta_{max} - \delta_{min}$ ) > $\theta$ ) then
14.     drift $\leftarrow$ true
15. endif
16. return drift
```

The input to this module are batches of unlabeled data (B_n), a classifier (ζ, trained at the first phase of DDAL), maximum and minimum density historical values (δ_{max}, δ_{min}), and drift and uncertainty thresholds (θ, λ). The virtual margin density ($\delta_{current}$) for current batch is computed according to Algorithm 1. Thus, $\delta_{current}$ is compared to the maximum and minimum density historical values (lines 06 to 12, Algorithm 2). Finally, the proposed method calculates whether or not the difference between δ_{max} and δ_{min} is higher than the drift threshold (θ). For higher differences, a drift is assigned and the reaction module is activated. Otherwise, the classification module is triggered.

At the reaction module, when the detection module indicates a concept drift, a new classifier (ζ') is trained using samples contained in the current batch of instances (B_n). Then, the proposed method updates δ_{max} and δ_{min} according to the density on B_{n+1} and the classification module is activated. As previously mentioned, labeled samples are required only at the reaction module to generate ζ', and are not necessary to detect concept drift. Consequently, the proposed method reduces the labeling cost and it does not require instances true label available as soon as a class is assigned to them.

III. EXPERIMENTAL RESULTS

In this section, we describe the experimental results attained by our method. We measure the performance of the proposed method in terms of its capability on detecting drift, its achieved accuracy and the amount of labeled instances involved. We also compare the proposed method to three baselines: DDM [2], EDDM [3] and MD3 [5]. The first two methods are online learning-based and fully supervised methods, thus, very different from DDAL, which is a batch-based and unsupervised drift detector. However, these two methods are investigated due to the fact that they represent state-of-the-art approaches. The third baseline, MD3, shares several similarities with DDAL, e.g. it is a batch-based method and it employs Active Learning and margin density, but it is designed to work only with SVM. In order to conduct a fair comparison among the four methods, we measure the performance of DDM and EDDM as the mean calculated over batch of 500 instances - the batch size used in this work.

The experiments were performed using 4 synthetic and 2 real datasets, and were broken down into two main series. The first series was conducted focusing on setting up all parameters involved in our method to attain the best of its performance. For this, we have tested three classification algorithms: Decision Tree (DT), K-Nearest Neighbors (kNN) and Linear Support Vector Machine (lSVM). These classifiers were tested with different values of Drift and Uncertainty thresholds (λ, θ). The second series of experiments was performed to compare the proposed method to the three baselines, DDM, EDDM and MD3. All baselines were employed using default parameters defined by their authors.

The results attained in the two series of experiments are discussed in sections C and D. First, we present parameters settings on our experiments. It is also important to mention that we have used Matlab with Machine Learning toolbox to carry out all experiments.

A. Datasets

Synthetic datasets allow us to evaluate the behavior of DDAL in terms of detection rate, detection delay, accuracy and amount of employed labeled instances. The following synthetic datasets [9] were investigated in this paper. They are balanced datasets and each 1000 instances represent a concept [4].

1) Line: This dataset contains 2k instances divided into two classes according to a line equation. Each instance is represented by two attributes. Line dataset contains only one abrupt concept drift and is noise-free.

2) Circle: It contains 4k instances, divided into two classes according to a circle equation. Each instance is represented by two attributes. Circle dataset contains three gradual concept drifts and is noise-free.

3) Sine1: It is a dataset composed of 10k instances divided into two classes according to a sine equation. Each instance is also represented by two features. This dataset presents nine gradual concept drifts, is noise-free and, after every drift, the classes are reversed.

4) Gauss: This dataset contains 10k instances divided into two classes according to two Gaussian distributions. Each instance is represented by two attributes. Gauss dataset contains nine abrupt concept drifts, it presents noise and, after every drift, the classes are reversed.

The real datasets only allow us to evaluate the behavior of the proposed method taking into account accuracy and amount of used labeled instances, since in real datasets it is not possible to know where a concept drift occurs. This is a common characteristic in real applications. The following real datasets are investigated:

1) Elec: This dataset covers a period of two years on electricity prices in New South Wales (Australia) and it contains 45,321 instances divided into two classes: up and down. The last class presents 58% of prior probability. Each sample is described by eight features and one class attribute [10]. The data is supposed to be subject to concept drift due to chaging consumptions habits, unexpected events and seasonality [11].

2) SPAM: It contains 9,324 instances divided into spam or legitimate classes Each instance is represented by 499 binary attributes and one class attribute [12]. Drift in this set is supposed owing to change in patterns of mail.

The diversity of the investigated datasets is intended to lead us to demonstrate the robustness of the proposed method to variation on feature dimensionalities and concept drift types.

B. Parameters Selection

The proposed method depends on tuning four parameters to work effectively. The first parameter is the base classifier ζ type. As mentioned before, in this paper we investigate three classification algorithms: DT, kNN and lSVM in order to select the best classifier. The second parameter is the batch size (κ). In [13], the authors tested the following sizes: 400, 600, 800 and 1200 instances. In [8], the authors used batch sizes fixed at 600 instances. In their turn, in [5], the authors defined batch size proportional to the dataset size. However, this last approach is not realistic, because in data stream problems the real dataset size is not known in advance. Based on this context, we have tested three batch sizes: 300, 400 and 500 instances. The evaluation metric used was the percentage of correct drift detections (*#correct detection*/*#known drifts*).

Table I summarizes the outcome of the tests taking into account κ variation. These tests were conducted only on synthetic datasets, where drift is assured. We can observe in this table that the highest percentage of correct drift detections is achieved when κ is 500 (highlighted in bold). So, we set the batch size to 500 instances.

TABLE I. CORRECT DRIFT DETECTIONS WHEN VARYNG *K* VALUES.

Dataset	***K***		
	300	400	**500**
Line	29.97%	36.35%	**74.78%**
Circle	73.99%	50.00%	**94.44%**
Sine1	39.41%	54.65%	**60.31%**
Gauss	40.50%	48.83%	**51.02%**

TABLE II. DDAL'S PERFORMANCE CONSIDERING DIFFERENT CLASSIFIER TYPES

Dataset	***Decision Tree***	
	#Lost Drift (#Known Drift)	**Error Rate**
Line	1 (1)	0.2084
Circle	3 (3)	0.3682
Sine1	1 (9)	0.2046
Gauss	9 (9)	0.3682
SPAM	- (-)	0.1556
Elec	- (-)	0.2421
Dataset	***kNN***	
	#Lost Drift (#Known Drift)	**Error Rate**
Line	0 (1)	0.0225
Circle	0 (3)	0.0305
Sine1	0 (9)	0.0267
Gauss	1 (9)	0.1854
SPAM	- (-)	0.1679
Elec	- (-)	0.1714
Dataset	***lSVM***	
	#Lost Drift (#Known Drift)	**Error Rate**
Line	0 (1)	0.0140
Circle	0 (3)	0.3238
Sine1	0 (9)	0.0440
Gauss	1 (9)	0.3275
SPAM	- (-)	0.0700
Elec	- (-)	0.2065

The third parameter is the Uncertainty Threshold (λ), used by the active learning strategy to design virtual margins and to compute virtual margins density. In [7], considering a supervised context, it is used a fixed value to λ (λ=1). Nonetheless, there are no reference values in the literature when concept drift is involved. Previous experiments conducted in this work showed that when λ=1 or λ<0.60, the virtual margin density is either null or maximum. In both cases, there is a disastrous impact on the detector performance. Thus, we have analyzed uncertainty threshold values ranging from 0.60 to 0.95, with increment of 0.05.

Finally, the fourth parameter (θ) defines the drift threshold. It refers to the input space change limit allowed before a drift is signaled [1]. MD3 [5] is the only reference using margin density variation to detect drift that we have found in the literature. Based on this reference, in this paper we investigated drift thresholds ranging from 0.005 to 0.095, with increment of 0.005. Therefore, we have carried out test with all 152 possible combinations of these parameters, for each classifier and dataset. Thus, 2736 tests were performed. In each test, drift detection rate and classifier's accuracy were measured.

It can be observed in Table II that, in terms of drift detection capability, the proposed method shows performance similar when using both kNN or lSVM classifiers. However, the same behavior was not observed with DT. The reason for this may be the DT high variance, since it is an unstable learning method. Despite DT results, considering that DDAL achieved similar results for kNN and lSVM, this may indicate that the proposed method works well with most of base classifiers.

It is important to point out that the main objective of this work is to evaluate DDAL's drift detection capability and not the classifier accuracy, i.e. the focus of this work is the drift detection phase and not the drift reaction or classification phase. Event though, the classification accuracy is evaluated in details in the second series of experiments. In addition, once DDAL achieved similar detection rates using both kNN and lSVM, as a tiebreaker, we used the error mean to choose kNN as base classifier to compare DDAL to the supervised baselines.

TABLE III. PARAMETERS SET UP

Dataset	**Parameters**		
	ζ	***λ***	***Θ***
Circle	kNN	0.60	**0.005**; 0.010; 0.015
		0.65	**0.005**; 0.010; 0.015
		0.70	**0.005**; 0.010; 0.015
		0.75	**0.005**; 0.010; 0.015
		0.80	**0.005**; 0.010; 0.015; 0.020; 0.025; 0.030; 0.035
		0.85	**0.005**; 0.010; 0.015; 0.020; 0.025; 0.030; 0.035
		0.90	**0.005**
		0.95	**0.005**
Sine1	kNN	0.80	**0.005**
		0.85	**0.005**
		0.90	**0.005**
		0.95	**0.005**
Gauss	kNN	0.80	**0.005**; 0.010
		0.85	**0.005**; 0.010
		0.90	**0.005**
		0.95	**0.005**
Line	kNN	0.60	**0.005**; 0.010
		0.65	**0.005**; 0.010
		0.70	**0.005**; 0.010; 0.015
		0.75	**0.005**; 0.010; 0.015
		0.80	**0.005**; 0.010; 0.015; 0.020; 0.025
		0.85	**0.005**; 0.010; 0.015; 0.020; 0.025
		0.90	**0.005**; 0.010; 0.015; 0.020; 0.025; 0.030; 0.035; 0.040; 0.045; 0.050
		0.95	**0.005**; 0.010; 0.015; 0.020; 0.025; 0.030; 0.035; 0.040; 0.045; 0.050
Elec	kNN	0.70	**0.005**; 0.010
		0.75	**0.005**; 0.010
Spam	kNN	0.70	**0.005**
		0.75	**0.005**
		0.80	**0.030**; 0.035; 0.040; 0.045; 0.050
		0.85	**0.030**; 0.035; 0.040; 0.045; 0.050

Table III summarizes the best values attained through exhaustive experiments when varying parameters ζ, λ and θ. It can be observed also that some combinations of values achieved identical results in terms of error rate and drift detection. The values highlighted in bold were used to compare

DDAL to the baselines, since they represent the optimized parameters.

C. *First Series of Experiments*

In this subsection, we show the results obtained by our proposed method tuned with the best parameters values selected in Table III. All results are shown in Fig. 3.

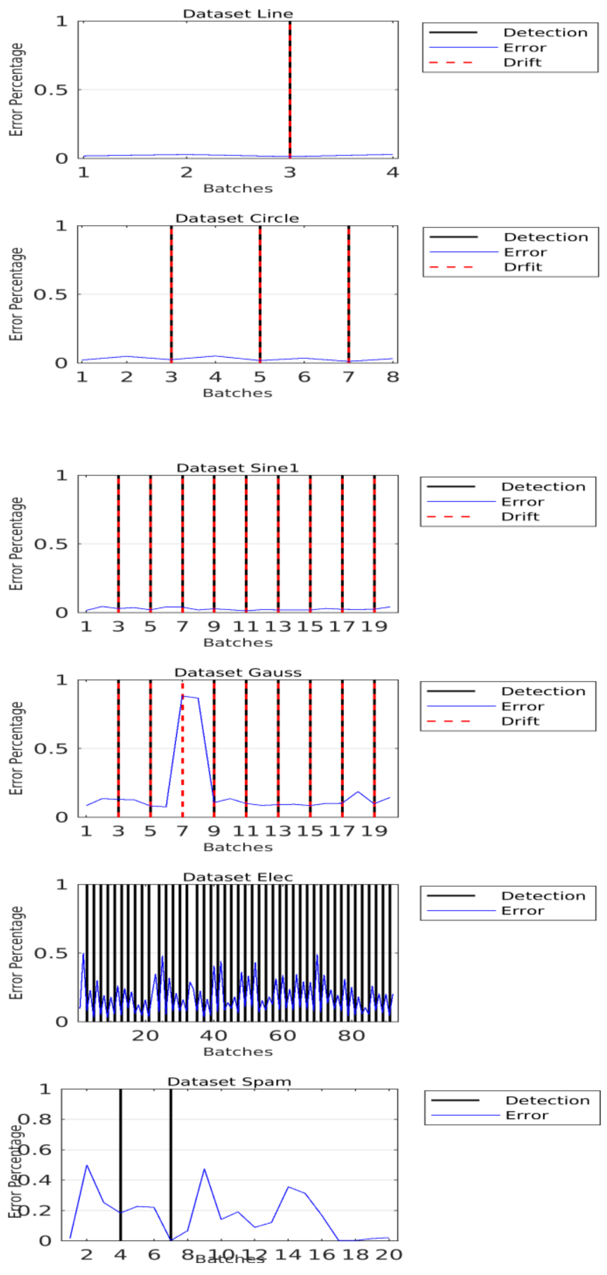

Fig. 3. Behavior of the proposed method in terms of true drift detection and prequential error. Red line represents known drifts and black line represents detected drift. The blue curve represents the average prequential error rate for each batch.

Since the results are shown in terms of prequential error rate, it is important to explain the prequential error rate concept. In [14], this error rate is defined as the sum of loss function of a model: $S_i = \sum_i^n L(y_i, \hat{y}_i)$, where y_i is the predicted output, $\hat{y}_i$ is the real output and n denotes the total number of instances.

Line, which was the first and smallest dataset investigated presents only one known drift. It may be observed that DDAL detects drift without delay, false or missing detection. Its prequential error rate is below 3%. In terms of circle dataset, where there are three known drifts, the proposed method also detects all drifts without delay, false or missing detection. In terms of prequential error rate, DDAL reaches 3.05% on average. These two first datasets present different types of concept drift, abrupt and gradual respectively. Even though, the proposed method was able to detect all drifts while keeping low prequential error rates in both cases. It is worth noting that both datasets are noise-free and do not present reversed classes.

Sine1 is also a noise-free dataset, but its classes are reversed after each drift. This characteristic represents an additional difficulty for the proposed method, since DDAL is based on classification confidence, which is a measure not sensible to class reversal. Yet, the proposed method obtained low average prequential error rate (2.67%) and was able to detect all nine known drifts with no false or miss detection.

The last synthetic dataset analyzed (Gauss) presents nine drifts, noise and class reversed after every drift. In this dataset, the proposed method missed one drift and achieved the highest average prequential error rate (18.54%) among all synthetic datasets. This behavior may be due to noise data.

As mentioned in the beginning of this section, for real datasets, we analyze the proposed method only in terms of accuracy and amount of used labeled instances. For the accuracy analysis, DDAL detected forty-four drifts and attained 17.14% as average prequential error rate in Elect dataset. In its turn, for Spam dataset, the proposed method detected only two drifts and achieved 16.79% of average prequential error rate.

Table IV summarizes all results taking into account amount of used labeled samples, delay and false detections. Additionally, it shows the average prequential error rate attained for each dataset. Detection delays are provided in parenthesis, where (0) means that there was no delay, (x) means that there was no detection and (-) means that the dataset does not contain any known drift (real datasets).

TABLE IV. SUMMARY OF DDAL'S BEHAVIOR CONSIDERING DIFFERENT MEASURES.

Dataset	#Drift	% Labels	False	Missing	Mean error	Delay
Line	1	33.33	0	0	0.0224	(0)
Circle	3	42.87	0	0	0.0305	(0)(0)(0)
Sine1	9	47.36	0	0	0.0267	(0)(0)(0) (0)(0)(0) (0)(0)(0)
Gauss	8	42.10	0	1	0.1854	(0)(0)(x) (0)(0)(0) (0)(0)(0)
Elect	44	49.09	-	-	0.1714	-
Spam	2	11.33	-	-	0.1679	-

D. Second Series of Experiments

In this second series of experiments, we statistically compare the proposed method to three baselines: DDM [2] (DT), EDDM [3] (DT and kNN) and MD3 [5] (lSVM), using their default parameters. The authors of the two first baselines highlighted that their methods are tailored to work with incremental learning classifiers. In this way, to compare DDAL to DDM and EDDM, we used kNN as base classifier. In addition, in order to conduct a fair comparison among all evaluated methods, we measure the performance of DDM and EDDM as a mean over each batch of 500 instances. Finally, since MD3 only works with linear SVM, the version of DDAL used to be compared to MD3 employs linear SVM as base classifier.

The statistical analysis performed is related to the methods average prequential error rate. We employed the Wilcoxon non-parametric statistical hypothesis test. This choice is due to: 1) There is no guarantee of a normal data distribution; and 2) The size of samples are, in general, lower than 30. Moreover, the null hypothesis assumes the same distribution of median, whereas the alternative hypothesis relies on different median distribution, with a 5% significance level, as shown in Equation 1.

$$
\begin{gathered}
H_0\!: median_1 = median_2, \\
H_1\!: median_1 \neq median_2, \\
\alpha = 5\%.
\end{gathered}
\tag{1}
$$

The results of the statistical tests are summarized in Tables V, VI and VII, where the following values are highlighted: #D(#K) - Number of Detections (Known Detection); AE (SD) - Average Error (Standard Deviation), %L - percentage of labeled instances. Considering the average prequential error rate as the variable of interest, values in bold indicate the lowest AE, while underlined values indicate significantly different results attained by the compared methods.

TABLE V. COMPARING DDM AND DDAL DRIFT DETECTION METHODS. VALUES IN BOLD INDICATE THE LOWEST AVERAGE PREQUENTIAL ERROR RATE, AND UNDERLINED VALUES INDICATE SIGNFICANTLY DIFFERENT RATES ACCORDING TO THE WILCOXON STATISTICAL TEST.

Dataset	***ρ-value***	***DDM***			***DDAL***		
		#D(#K)	AE (SD)	%L	#D(#KD)	AE (SD)	%L
Line	0.1429	2(1)	0.0678 (0.0502)	100	1(1)	**0.0224 (0.0082)**	33.33
Circle	0.0155	3(3)	0.0811 (0.0507)	100	3(3)	**0.0305 (0.0141)**	42.87
Sine1	<0.0005	10(9)	0.0769 (0.0389)	100	9(9)	**0.0267 (0.0097)**	47.36
Gauss	<0.0005	10(9)	**0.1751 (0.0442)**	100	8(9)	0.1854 (0.2373)	42.10
SPAM	0.5295	7(-)	**0.1220 (0.0516)**	100	2(-)	0.1679 (0.1539)	11.33
Elec	0.0130	26(-)	0.1879 (0.0349)	100	44(-)	**0.1714 (0.1219)**	49.09

Table V shows the comparison between DDM and DDL. For all datasets, both methods produced close prequential error rates. However, DDAL is significantly better on Circle, Sine1 and Elec datasets. In terms of percentage of labeled samples used, since DDM is a fully supervised-method, all labels are required, whereas the proposed method used from 11.33% to 49.09% labeled instances. In addition, DDM generated higher false detection rate, while DDAL presents a missing detection.

Table VI shows the comparison between EDDM and the proposed method. These results are very similar to the previous results. Again, DDAL is significantly better on Circle, Sine1 and Elec datasets, in terms of prequential error rates. Moreover, EDDM is a fully supervised-method, thus using all labeled samples. In its turn, the proposed method needed from 11.33% to 49.09% labeled instances. These results indicate that our proposed unsupervised method is robust enough to correctly detect drifts at the right moment, as a supervised method is able to do. On the other hand, DDAL is more affected by noise data than both DDM and EDDM.

TABLE VI. COMPARING EDDM AND DDAL DRIFT DETECTION METHODS. VALUES IN BOLD INDICATE THE LOWEST AVERAGE PREQUENTIAL ERROR RATE, AND UNDERLINED VALUES INDICATE SIGNFICANTLY DIFFERENT RATES ACCORDING TO THE WILCOXON STATISTICAL TEST.

Dataset	***ρ-value***	***EDDM***			***DDAL***		
		#D(#K)	AE (SD)	%L	#D(#KD)	AE (SD)	%L
Line	0.2286	1(1)	0..0542 (0.0457))	100	1(1)	**0.0224 (0.0082)**	33.33
Circle	0.0031	3(3)	0.1015 (0.0630)	100	3(3)	**0.0305 (0.0141)**	42.87
Sine1	<0.0005	10(9)	0.0721 (0.0361)	100	9(9)	**0.0267 (0.0097)**	47.36
Gauss	0.0094	10(9)	**0.1548 (0.0451)**	100	8(9)	0.1854 (0.2373)	42.10
SPAM	0.5200	5(-)	**0.1177 (0.0529)**	100	2 (-)	0.1679 (0.1539)	11.33
Elec	0.0372	52(-)	0.1853 (0.0330)	100	44(-)	**0.1714 (0.1219)**	49.09

TABLE VII. COMPARING MD3 AND DDAL DRIFT DETECTION METHODS. VALUES IN BOLD INDICATE THE LOWEST AVERAGE PREQUENTIAL ERROR RATE, AND UNDERLINED VALUES INDICATE SIGNFICANTLY DIFFERENT RATES ACCORDING TO THE WILCOXON STATISTICAL TEST.

Dataset	***ρ-value***	***MD3***			***DDAL***		
		#D(#K)	AE (SD)	%L	#D(#KD)	AE (SD)	%L
Line	0.0040	1(1)	0.0781 (0.0268)	17.64	1(1)	**0.0148 (0.0170)**	33.33
Circle	<0.0005	2(3)	0.3322 (0.1438)	16.21	3(3)	**0.0305 (0.0141)**	42.87
Sine1	<0.0005	3(9)	0.4158 (0.4154)	9.27	9(9)	**0.0443 (0.0139)**	47.36
Gauss	0.0781	4(9)	0.4872 (0.2206)	12.37	9(9)	**0.3261 (0.0940)**	42.10
SPAM	0.0285	3	0.1389 (0.1424)	9.97	8(-)	**0.0741 (0.1264)**	11.33
Elec	<0.0005	17	0.3444 (0.0779)	30.55	42(-)	**0.2113 (0.1002)**	46.86

Finally, Table VII shows a comparison between MD3 and DDAL (SVM as base classifier). In this case, if we take into account prequential error rates, the proposed method is significantly better than MD3 in all datasets. Here, there is an interesting result considering the percentage of labeled samples used. MD3 required much less labeled samples when compared to the proposed method. However, it is possible to note that this behavior is directly related to the MD3 lower drift detection capability on the datasets investigated in this paper. For instance, in Sine1, only 3 drifts were identified out of a total of 9 drifts, as it can be seen in Table VII. Therefore, our method establishes a better trade-off between amount of used labeled samples and detection capability. DDAL rates in Table VII are different from those shown in Tables V and VI due to employing SVM as base classifier.

IV. CONCLUSION

This paper proposed DDAL, a drift detector method based on Active Learning, precisely the Fixed Uncertainty strategy. In the proposed method, this technique of Active Learning, which is used to select samples, was tailored to work as a support to detect drift in an unsupervised way. Such an adaptation was performed by virtual margins measured in terms of classifier confidence, i.e. maximum posterior probability.

The assumption is that this approach allows drift detection to be performed with no need to monitor error rate, thus, avoiding the dependence on classifier's performance decrease and on labeled samples. Besides, the proposed method proved to be base classifier independent, since any learning method may be employed as a classifier in DDAL, even though we have investigated only kNN, SVM and Decision Tree.

Experiments were carried out on seven different datasets, including synthetic and real data. These experiments showed that the proposed method attained error rates similar, or even better, than supervised baselines (DDM and EDDM), while achieving similar drift detection rates. When compared to an unsupervised baseline (MD3), DDAL produced better error and detection rates. As a consequence, our method establishes a better trade-off between amount of labeled samples and detection capability.

It is worth noting that DDAL expands MD3 to be used with any learning algorithm, besides SVM. Work proposed in [6] also focuses on this objective. Both methods use active learning to deal with concept drift, but DDAL is an unsupervised drift detection method, whereas [6] is supervised since it is based on monitoring the error rate of the N lastly instances. In addition, DDAL employs the Fixed Uncertainty strategy to work with virtual margins, while the work presented in [6] uses classifier ensemble diversity to attain the same objective.

For future work, we intend to use classifier ensembles composed of both homogeneous and heterogeneous base classifiers, instead of single classifiers. This objective is due to the fact that ensembles generally make different and independent errors, which means that classifier ensembles may produce more accurate decisions [15]. In this way, the classification module may be improved by using classifier ensembles. Furthermore, classifier ensemble allows calculating the mean of uncertainty, providing higher accuracy about sample uncertainty [7].

REFERENCES

[1] M. Sayed-Mouchaweh. "Learning From Data Streams in Dynamic Environments ,", VIII, 75, Springer International Publishing, 2016.

[2] J. Gama, P.medas, G. Castillo and P. Rodrigues. "Learning with local drift detection".*In Advanced Aritifical Intelligence,* vol. 3171 of Lecture Notes in Computer Science. Springer Berling/Heidelberg, pp. 286-295, 2004.

[3] M. Baena-Garcia, J. Del Campo-Vila, R. Fidalgo and A. Bifet. "Early drift detection method", *In ECML PKDD 2006, Workshop on Knowledge Discovery from Data Streams*, pp. 77-86, 2006.

[4] P, Felipe, A. and Dos Santos, Eulanda, M. "A dissimilarity-based drift detection method". *In Tools with Artificial Intelligence (ICTAI)*, IEEE 27thInternational Conference on, pp.1069-1076, 2015.

[5] T.S. Sethi and M. Kantardzic. "Don't pay for validation: Detecting drifts from unlabeled data using margin density", *INNS Conference on Big Data*, Volume 53, pp. 103-122, 2015.

[6] T.S. Sethi and M. Kantardzic. "On the reliable detection of concept drift from streaming unlabeled data". Expert Systems with Applications, 82, 77-99. Elsevier, 2017.

[7] I. Zliobaite, A. Bifet, B.Pfahringer, G. Holmes. "Active Learning with Drifting Streaming Data", IEEE Transactions on Neural Netowrks and Learning Systems, Vol.25 (1), pp.27-39, 2014.

[8] E. Arabmakki, M. Kantardzic, T. S. Sethi. "RLS-A Reduced Labeled Sampes Approach for Streaming Imbalanced Data with Concept Drift", *Information Reuse and Integration (IRI) 2014, IEEE 15th Internation Conference on*, pp.779-786, 2014.

[9] F. L. Minku, A. White, and X. Yao. "The impact of Diversity on Online Ensemble Learning in the Presence of Concept Drift", *IEEE Trans. Knowledge and Data Eng.* Vol. 22, no. 5, pp. 730-742, May, 2010.

[10] M. Harries. "Splice-2 comparative evaluation: Electricity princing". Technical report, The University of South Wales, 1999.

[11] I. Zliobaite. "How Good is the Electricity Benchmark for Evaluating Concept Drift Adaptation", arXiv:1301.3524, 2013.

[12] I. Katakis, G. Tsoumakas and I. Vlahavas. "Tracking recurring contexts using ensemble classifiers: an application to email filtering". *Knowledge and Information Systems*, 2010.

[13] E. Arabmakki, M. Kantardzic, T. S. Sethi. "Ensemble Classifier for Imbalanced Streaming Data Using Partial Labeling", *Information Reuse* ational Publishing, 2016.

[14] A. P. Dawid. "Statistical Theory: The Prequential Approach", *Journal of The Royal Statistical Society. Series A(General)*. Vol. 147, no. 2, pp. 278-292, 1984.

[15] A. H R. Ko., R. Sabourin and A. S. Brito Jr. "From dynamic classifier selection to dynamic ensemble selection". *Pattern Recognition*. Vol. 41, pp. 1718-1731, 2008.