Time Series Forecasting in the Presence of Concept Drift: A PSO-based Approach

Gustavo H. F. M. Oliveira*, Rodolfo C. Cavalcante†, George G. Cabral‡, Leandro L. Minku§, and Adriano L. I. Oliveira*,
*Center de Informatica, Universidade Federal de Pernambuco, Recife, Brazil
†Nucleo de Ciencias Exatas, Federal University of Alagoas, Arapiraca, Brazil
‡Statistics and Informatics Department, Federal Rural University of Pernambuco, Recife, Brazil
§Department of Informatics, University of Leicester, Leicester, UK
ghfmo@cin.ufpe.br, rodolfo.cavalcante@arapiraca.ufal.br, george.gcabral@ufrpe.br, leandro.minku@leicester.ac.uk, alio@cin.ufpe.br

***Abstract*—Time series forecasting is a problem with many applications. However, in many domains, such as stock market, the underlying generating process of the time series observations may change, making forecasting models obsolete. This problem is known as Concept Drift. Approaches for time series forecasting should be able to detect and react to concept drift in a timely manner, so that the forecasting model can be updated as soon as possible. Despite the fact that the concept drift problem is well investigated in the literature, little effort has been made to solve this problem for time series forecasting so far. This work proposes two novel methods for dealing with the time series forecasting problem in the presence of concept drift. The proposed methods benefit from the Particle Swarm Optimization (PSO) technique to detect and react to concept drifts in the time series data stream. It is expected that the use of collective intelligence of PSO makes the proposed method more robust to false positive drift detections while maintaining a low error rate on the forecasting task. Experiments show that the methods achieved competitive results in comparison to state-of-the-art methods.**

I. Introduction

A time series is a collection of observations measured sequentially over time. In this kind of dataset, observations are ordered in time and typically present serial correlation. Several real-world processes measure data over time that can be modeled as time series, such as company payroll [1], stock price movements [2], exchange rates [3], temperatures of a city, electroencephalogram, among others.

In the literature, most approaches proposed for time series analysis model time series data as a static dataset. These methods typically process the time series in offline mode [4], through several passes on the historical data. However, in most real-world time series applications, data arrives sequentially as a stream. In this streaming scenario, data may flow at high speed and evolve over time, making offline modeling and forecasting methods innefective and innapropriate.

Dynamic data streams present several challenges for predictive models [5]. Since the data generation process may change over time, the historical observations are not always useful for defining future behaviors, and may even be detrimental. Changes in the underlying data generation process are known as concept drifts [6]. This phenomenon is very comon in real-world time series data streams. For instance, stock price time series may change due to changes in political and economical factors or changes in investors, psychology. In this case, predictive models trained with the historical data become obsolet in predicting future behaviors from the changing point.

Dynamic data streams require effective and efficient learning algoritms able to adapt to concept drift [7]. Effective in the sense that the algorithm has to maintain predictive performance by adapting to changes. Efficient in the sense that it should not unnecessarily trigger adaptation when this is not needed. Two major classes of drift handling methods are the passive and active concept drift handling methods [7]. Passive methods, also known as blind adaptive methods, adapt the learned model without any explicit detection of changes [4]. These methods may be not efficient. Active adaptive concept drift handling methods, on the other hand, use a mechanism to explicitly detect drifts. These methods may be more efficient, since they implement a detect-to-react mechanims. Another advantage of active methods is the transparency.

Active concept drift handling methods rely on monitoring some aspect of the data stream to explicitly detect changes in data. One of the most used strategies is to monitor the errors of a classifier or forecaster in an online fashion [8], [9], [10]. The main assumption of these methods is that, when a concept drift happens, the learned model is no longer effective in describing data behavior and its prediction error tends to increase. If the goal of the designer is to keep good forecasting performance, monitoring the error may be an effective drift detection scheme.

The main issue of existing active adaptive methods is that they typically monitor the error of a single forecasting method. If this single model was built as result of a poor training process, the drift detection may be inaccurate. This may be due to the fact that the error will reflect generalization problems, such as overfitting or underfitting, instead of changes in the data stream. Some detection delay may also be introduced in the process, since the error of the single forecaster may increase until some threshold level before a drift is detected, which degrades the overall forecasting performance. A drift detection approach based on monitoring the error of a single forecaster may also present a high false positive detection rate due to spurious variations in the error that are not caused by drifts.

In order to overcome those issues, we propose the use of an active adaptive learning system for time series forecasting based on monitoring the error of several forecasting models generated by swarm intelligence. In batch learning forecasting schemes, optimization algorithms, such as particle swarm optimization (PSO) [11] can be used to build forecasting models. PSO generates several particles representing different models, which are evolved to optimize the performance in training or validation data. When a concept drift happens, the corresponding function to be optimized changes, making optimal solutions outdated. Therefore, we hypothesize that when a change in the underlying time series generation process happens, the search space faced by PSO reflects this change. So, instead of monitoring the error of a single forecasting method, we can identify concept drift by monitoring a larger area of the optimization search space through the several forecasting models generated by PSO.

We expect this approach to improve drift detection and forecasting performance of the learning system, because monitoring a larger area of the search space may improve robustness against individual variations in the error caused by the training of a single model. In order to validate this hypothesis, we propose two drift detection approaches: (i) detection based on swarm behavior and (ii) detection based on the evaluation of some particles (called sensors). The proposed methods were evaluated with both artificial and real-world time series in terms of drift detection accuracy and forecasting performance.

The rest of this paper is organized as follows. Section II discusses some related work. Section III describes the proposed methods in detail. Section IV describes the experimental setup, the computational experiments performed and the results obtained. Section VI concludes the paper and gives directions for further research on this topic.

II. Related Work

Despite the fact that the concept drift problem is not a new research area, concept drift detection in time series forecasting is not widely investigated. Boracchi and Roveri [12] used the self-similarity feature to identify concept drifts in time series. The proposed approach measures the self-similarity between time series segments and uses the values of this feature as the change detector variable. The Intersection of Confidence intervals (ICI) test is used as concept drift test (CDT). ICI is used to monitor and detect changes in this feature as a proxy for changes in the time series generation process. However, this approach monitors only one aspect of time series behavior and its applicability is restricted to time series that present self-similarity. This work is concerned with drift detection, and does not investigate how to handle the detected changes in order to improve forecasting performance.

Cavalcante et al. [13] also applied a CDT to monitor time series features and identify concept drifts. They used a set of statistical time series features to characterize concepts and monitor the diferences between subsequent feature vectors extracted from the time series in an online manner. A CDT is then applied to the stream of differences to identify significant changes in the underlying time series generation process. This work is also concerned with drift detection and not with forecasting performance.

Cavalcante and Oliveira [10] investigated the use of an online, explicit drift detection method to handle concept drift in financial time series. They proposed the use of some existing CDTs to monitor the error of extreme learning machines (ELM) [14] to detect concept drifts in time series and adapt the learned model. However, they did not evaluate the drift detection accuracy of the proposed methods.

In the literature, several studies have investigated how to use bioinspired and collective intelligence to improve forecasting accuracy of time series predictors. Most of existing methods implicitly or explicitly assume that the environment modeled is stationary. With this assumption, their goal is to find local or global optima in the search space, which typically represents the best configuration for the forecasting algorithm. However, as discussed above, many real-world problems are dynamic, i.e., non-stationary. The function to be optimized can change over time, which makes the optimal solutions obsolete [15]. In the optimization literature, these problems are known as dynamic optimization problems (DOPs). In particular, some work has investigated the use of PSO to detect changes in benchmark optimization functions, such as the Moving Peaks Benchmark (MPB). Hu and Eberhart [16], Richter [17] and Fouladgar and Lotfi [18] proposed methods to detect changes in MPB by reevaluating sensors of the swarm. Other researchers, such as Janson and Middendorf [19], tried to detect changes in MPB based on the behavior of the swarm. Nasiri et al. [20], on the other hand, investigated how to combine these two strategies to detect changes.

Other researchers have investigated the use of PSO to build and train an artificial neural network to perform in dynamic environments [21], [22]. The proposed methods were applied to a dynamic classification problem. In these studies, the detection modules are based on sensors. These sensors evaluate some fixed point in search space in order to find some variation, which are interpreted as changes in the space. In the case of changes, the neural network is updated to model the new space. However, swarm intelligence has never been investigated in the context of time series forecasting with concept drift.

III. Proposed Methods

This work proposes and investigates the use of two approaches for concept drift detection based on the swarm behavior: (i) detection based on the whole swarm behavior (IDPSO-ELM-B); and (ii) detection based on the evaluation of some particles used as sensors (IDPSO-ELM-S). In both cases each swarm particle consists of an ELM artificial neural network model [14], which is trained based on the Improved Self-Adaptive Particle Swarm Optimization (IDPSO) [11]. The use of PSO was motivated by works like [23] where this technique performed better for training the ELM initial weights than the canonical random assignment. Furthermore, the algorithm IDPSO has been chosen because some works

report good results of the IDPSO in comparison to other state-of-art optimization methods.

In this work, IDPSO is responsible for finding the weights of the ELM connections between the input and the hidden layer. So, assuming that $m+1$ is the number of inputs (also considering the bias) for each ELM model and being n the size of the hidden layer, the IDPSO search is performed in a $(m+1)*n$ dimensional space. Figure 1 depicts the IDPSO particle coordinates representation.

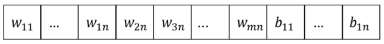

Fig. 1. Particle representation.

The training phase involves: (i) the swarm initialization, where each particle is assigned a random vector of input weights, and (ii) the swarm search, where the particles fly in the search space aiming at optimizing the fitness function. Mean Average Error (MAE) for the time series prediction on the training set was adopted as the fitness function. If the best training error among the whole swarm remains the same during a predefined number of consecutive swarm iterations, the search is interrupted. Finally, the particle with the lowest training error ($gBest$) is used as the forecasting model.

Once a model is elected for forecasting via the aforementioned training phase, this model must forecast each new arriving instance. In the future, when a concept drift takes place, the current forecasting model is reset and replaced by a new model trained using data that belong to the new concept [4]. However, while the system does not have access to enough examples belonging to the new concept for such retraining to start, the old forecasting model keeps being used to perform estimations. The number of new instances enough to define the new concept is a pre-defined parameter.

Algorithm 1 presents the overall training procedure. The present work proposes two methods for concept drift detection in time series forecasting and Algorithm 1 is a general procedure applicable to both methods. This general procedure receives as inputs the data stream X, the size of the training set n, the swarm size and specific parameters of the methods that will be hereafter presented in Sections III-A and III-B. Initially, the training set (tr) is built and a swarm of ELM models is used to search for the best forecasting model. Next, statistics of the models errors for the training set are computed to support the concept drift detection in the future. At this point, the best particle is used as the forecasting model and the system monitors the data stream in order to detect changes. Once a concept drift occurs, a new swarm of ELMs is trained.

Figure 2 depicts the on-line forecasting process for a time series containing several simulated concept drifts. The yellow time intervals represent the periods in which the system is performing the on-line forecast. When the system detects a concept drift (blue vertical dashed line for a true positive detection and pink vertical dashed line for a false positive case), the retraining period starts and the system begins gathering data from the new concept in order to train the new forecasting model.

Algorithm 1 Overall procedure

1: **Input:** X, n, swarm_size, spec_parameters
2: tr = X($t : t+n$)
3: Execute on tr the IDPSO swarm S = $\{p_1, p_2$ .. $p_{swarm\_size}\}$
4: Compute the swarm error and determine the best model $gBest$
5: **for** each new arriving data x_t **do**
6: $\hat{x}_t$ = prediction given by the $gBest$ model
7: changed = DetectChange($spec\_parameters$, $\hat{x}_t$)
8: **if** $(changed)$ **then**
9: Wait for the next n training examples and set tr = X($t : t+n$)
10: Execute on tr the IDPSO swarm S = $\{p_1, p_2$ .. $p_{swarm\_size}\}$ using tr
11: Compute the swarm error on tr and determine the best model $gBest$
12: **end if**
13: **end for**

A. Concept Drift Based on Swarm Behavior (IDPSO-ELM-B)

Given the convergence of the swarm for a particular solution, it is possible to assume that the swarm has learned the current concept. Being S a swarm composed of particles p_i and each of these particles compute their own vector of prediction errors for the n observations contained in the training set (tr), the mean μ^p and standard deviation σ^p of the errors of each particle p are computed. This method is based on the assumption that the average error changes when a new concept is introduced [8]. Therefore, the average training error provided by the particles form a distribution that statistically changes as a new concept takes place. To monitor the swarm error behavior, the mean $\mu^S = 1/swarm\_size \sum_p \mu^p$ and the standard deviation $\sigma^s = \sqrt{1/swarm\_size \sum_p (\mu^p - \mu^s)^2}$ of the average errors of the particles are employed.

In order to detect a concept drift, the detection module adapts the equations introduced in the ECDD [9] which applies the Exponentially Weighted Moving Average (EWMA), which consists in an estimator of the mean of a sequence of variables such that the more recent variables are considered more important. Therefore, the detection condition is $Z_t > \mu^S + c * \sigma_{Z_t}$, where $Z_t = (1-\lambda) * Z_{(t-1)} + \lambda * e_t^s, \quad s.t. \quad t > 0$ and $\sigma_{Z_t} = \sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})\sigma^S}$. In this Equation, the predefined constant c increases or decreases the tolerance level for the forecasting error. λ indicates the importance level given to more recent prediction errors and e_t^s stands for the average swarm error for a single new example arriving at time t.

Ultimately, the assumption that this method overcomes the state-of-art methods, since it operates monitoring a statistical error distribution, in contrast to other methods, that monitor only a single model outcome, is investigated.

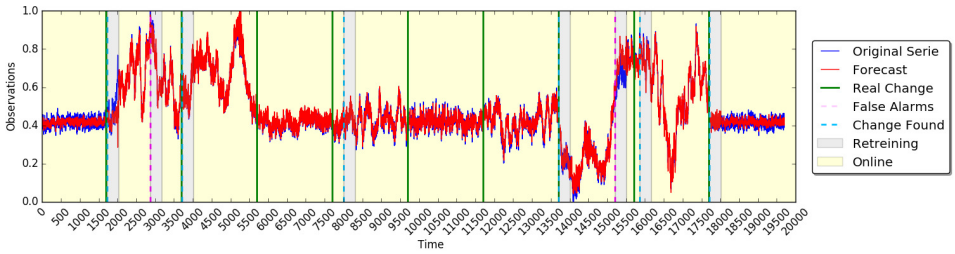

Fig. 2. Procedure for online forecasting in time series with concept drift.

B. Concept Drift Based on Sensors (IDPSO-ELM-S)

For this method, the idea is to pick a set of best particles (sensors - S) and monitor these particles in an isolated manner. This method has two main differences compared to the IDPSO-ELM-B: (i) only the best particles are considered and (ii) a concept drift is detected only when all the sensors agree. In this case, a concept drift only takes place when $Z_t > \mu^p + c * \sigma_{Z_t}, \quad \forall \quad p \in S$ - where $Z_t = (1-\lambda) * Z_{(t-1)} + \lambda * e_t^p$ and $\sigma_{Z_t} = \sqrt{\frac{\lambda}{2-\lambda}(1-(1-\lambda)^{2t})\sigma^p}$ - is satisfied for all the sensors.

Different from the IDPSO-ELM-B method, individually each particle uses the mean of the errors (μ^p) and standard deviation (σ^p) on the training set and the current error of the particle (e^p). As all best particles must agree that a concept drift has been detected, we expect this method to reduce the number of false positive drift detections further with respect to IDPSO-ELM-B.

IV. Experimental Setup

This section describes the objectives of the experiments, the metrics and the parameter choice used in the experiments. The objectives are the following: (i) Evaluate the accuracy of the concept drift detection in comparison to state-of-the-art methods; (ii) evaluate the accuracy of the proposed methods for the forecasting task in the presence of concept drift.

A. Concept Drift Detection Ability

The first objective of the experiments consists in comparing the proposed methods in terms of concept drift detection accuracy. The idea is to confirm the hypothesis that monitoring the statistics of the swarm error is effective for decreasing the number of false positive (FP) occurrences. Furthermore, this work tries to check the reliability of the methods also in terms of delay and transparent operation for the user [24] (i.e., the ability to explicitly inform whether a concept drift has occurred or not). Some methods automatically adapt to the new concept without informing the occurrence of the concept drift, these methods are not considered transparent.

As concept drift is not widely investigated in time series forecasting, there is a lack for time series datasets affected by concept drift. So, for these experiments, 120 artificial time series containing concept drifts were generated. For each time series, each detection method was executed 10 times and the computed averages were used for comparison. Using synthetic series makes possible a more accurate analysis of the performance of the concept drift detection methods since the points where the concepts change are known in advance. These datasets are detailed in Section IV-C.

In order to enable the first analysis of the experiment, the methods DDM [8], ECDD [9] and FEDD [13] are compared to the proposed methods according to a curve of False Positive Rate (FPR) vs Detection Delay (DD) introduced in [25]. These curves show the behavior of the compared methods for all the parameter configurations investigated, so there is no need to fix or chose the best parameters for each method. Each point in the curve corresponds to a different parameter setting. The best methods are those with curves close to the axes of the graph, which are those with small delay and false positive detections. For the comparison to be fair, all the experimented detection methods used ELM artificial neural networks as the regression methods.

The parameters used in this experiment are as follows: IDPSO-ELM-B: c = 0.25, 0.5, 0.75, 1; IDPSO-ELM-S: c = 0.25, 0.5, 0.75, 1; ELM-ECDD: c = 0.25, 0.5, 0.75, 1; ELM-FEDD: c = 0, 0.25, 0.5, 0.75 and; ELM-DDM: ω = 3, 5, 6, 8. The constant c controls the tolerance for the forecasting error. High values for c increase the detection delay (DD) rate. If a low value is given, less tolerance to error is allowed, thus increasing the FPR rate. The constant ω behaves similarly to c, however, for the method DDM.

Aimed at assigning a proper value for the number of observations for training the regression models, the values 200, 300, 400, 500 were experimented and the value 300 yielded the best results. So, this value was then fixed for the remaining of the experiments. For the methods employing IDPSO, preliminary experiments were performed to find the best parameters regarding the forecasting accuracy. The best parameters were as follows: 50 iterations; 30 particles (swarm size); 3 iterations without improvement as stop criterion; $\omega_{initial} = 0.8$; $\omega_{final} = 0.4$; c_1 = c_2 = 2; $max\_position$ = $max\_velocity$ = 0.3; and $min\_position$ = $min\_velocity$ = -0.3;

B. Time Series Forecasting Ability

The second goal of the experiments is to assess the accuracy of the proposed methods in terms of time series forecasting error in the presence of concept drift. This experiment is aimed

at validating the hypothesis that using swarms to monitor the concept drift phenomenon may improve the forecasting performance.

The same experimental procedure described in Section IV-A was used for this experiment. However, for this case, the prediction error metric Mean Absolute Error (MAE) was considered. Furthermore, to statistically attest the significance of the results, the Friedman non parametric test, with significance level α = 0.05 was used, as proposed by [26]. Friedman's test was used to rank the algorithms for each dataset separately and if the null hypothesis was rejected (ranks are significantly different) a Nemenyi test was employed by comparing all algorithms with each other, performing pairwise tests. The Nemenyi test was then used with significance level α = 0.05.

Preliminary tests were performed in order to obtain the best ELM parameters considering the forecasting accuracy. The best found configuration used 10 hidden neurons (h), 5 input neurons (l) and the sigmoid as activation function. Moreover, the remainder of the parameters were assigned according to Section IV-A.

In addition to the synthetic data sets used to address the first objective of the experiments (section IV-A), three time series containing stock market index, described in Section IV-C, were used for this experiment. For each time series, 30 executions were performed. The number of executions was chosen due to the small number of time series for this experiment. This amount of executions enables a more reliable analysis of the results. A comparison to an ELM network without concept drift detection was also performed in order to highlight the benefits of concept drift detection.

Lastly, the parameters that yielded the best error prediction for each method on the real world time series were: IDPSO-ELM-B: c = 0.25; IDPSO-ELM-S: c = 0.25; ELM-ECDD: c = 0.25; ELM-FEDD: c = 0.25 e; ELM-DDM: ω = 3. The best ELM topology used h = 10 and l = 5.

C. Datasets Description

This section details the datasets used in the experiments. Four types of artificial time series were generated: (i) linear; (ii) non linear; (iii) seasonal; and (iv) hybrid (containing all the three previous concepts). For each of the four cases, 30 time series containing 20000 observations were generated. Each dataset is composed of 10 concepts, each one comprising 2000 observations. The linear time series were simulated using an autoregressive (AR) process defined by: $x_t = a_1 x_{(t-1)} + a_2 x_{(t-2)} + .. + a_p x_{(t-p)} + w_t$, where $a_1, .., a_p$ are parameters of the AR model and $w_t : t = 1, .., n$ is a Gaussian white noise where the variables w_t belong to a normal distribution and are Independent and Identically Distributed (IID).

The non linear time series used in the experiments were generated by two non linear smooth models, given by [13]: $x_t = [a_1 x_{(t-1)} + a_2 x_{(t-2)} + a_3 x_{(t-3)} + a_4 x_{(t-4)}] * [1 - exp(-10x_{t1})]^{-1} + w_t$ and $x_t = a_1 x_{(t-1)} + a_2 x_{(t-2)} + \left[a_3 x_{(t-1)} + a_4 x_{(t-2)}\right] * \left[1 - exp\left(-10x_{t-1}\right)\right]^{-1} + w_t$. The seasonal linear time series were simulated by a linear model using variables (s) indicating the number of periods of the series. This model contains s seasons defined by $x_t = m_t + \beta_{(1+mod(t-1,s))} + w_t$ where m_t represents the tendency, $\beta_{(1+mod(t-1,s))}$ the seasonal factor and w_t the Gaussian noise.

The parameters of the models were varied in order to represent concept drift. Table I contains the investigated parameters as well as the respective equations. In order to simulate recurrent concepts, from the sixth concept onwards, the concepts generated were repeated. Figure 3 illustrates some time series containing the referred concepts.

TABLE I
PARAMETERS USED ON THE TIME SERIES MODELS

		α	p
Linear	1	{0.42, 0.28, 0.005}	3
	2	{0.003, -0.005, 1.0}	3
	3	{0.018, 0.95, 0.032}	3
	4	{0.11, 0.32, -0.028, 0.038, 0.48}	5
	5	{0.032, 0.41, -0.24, -0.22, 1.0}	5
	6	{0.14, -0.29, 0.0025, 1.0}	4
		α	Model
Non linear	1	{0.55, 0.024, 0.41, 0.009}	NL 1
	2	{0.059, 0.086, 0.62, 0.21}	NL 2
	3	{0.47, 0.57, 0.14, -0.19}	NL 1
	4	{0.55, 0.024, 0.41, 0.009}	NL 1
	5	{0.059, 0.086, 0.62, 0.21}	NL 2
	6	{0.55, 1.0, 0.0028, -0.58}	NL 2
		β	s
Seasonal	1	{34, 32, 30, 28, 26, 24, 22, 24, 26, 28, 30, 32}	12
	2	{34, 26, 18, 10, 18, 26, 10}	7
	3	{34, 26, 18, 10, 18, 26}	6
	4	{34, 26, 18, 10, 2, -6, -14, -6, 2, 10, 18, 26}	12
	5	{34, 10, -14, 10}	4
	6	{38, 28, 18, 8, 0, -8,-18,-8, 0, 8, 18, 28}	12
		α, β	Model
Hybrid	1	{0.003, -0.005, 1.0}	AR
	2	{Last 3 observations}	Seasonal
	3	{0.059, 0.086, 0.62, 0.21}	NL 2
	4	{0.018, 0.95, 0.032}	AR
	5	{Last 3 observations}	Seasonal
	6	{0.55, 0.024, 0.41, 0.009}	NL 1

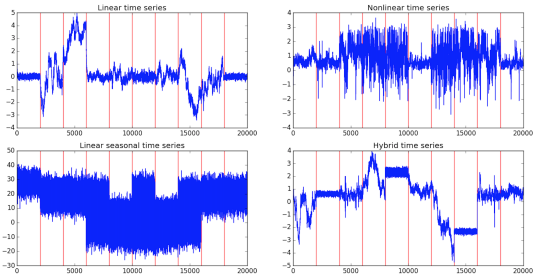

Fig. 3. Examples of the four time series types used in this work.

Lastly, three real time series were experimented: Dow Jones from January-29-1985 to May-12-2017, Nasdaq from May-02-1985 to May-12-2017 and S&P 500 from May-15-1950 to May-12-2017. These time series are available on Yahoo Finance [1].

V. Results and Discussion

This section presents the results of the experiments that are organized as follows: (i) concept drift detection evaluation; (ii)

[1]https://finance.yahoo.com/

time series forecasting ability in the presence of concept drift evaluation and; (iii) time series forecasting ability for real time series evaluation.

A. Concept drift detection evaluation

Figure 4 compares the proposed methods with the methods DDM, ECDD and FEDD regarding concept drift detection.

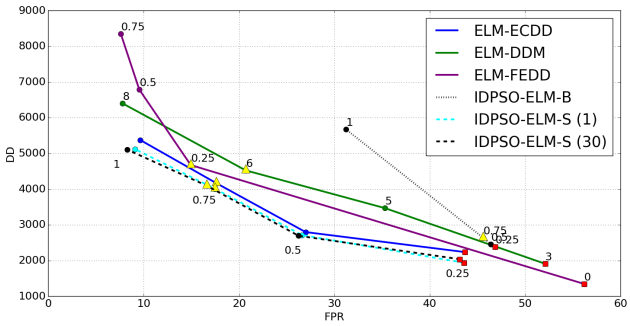

Fig. 4. Evolution of the models in terms of false positive rate and detection delay when varying the models parameters. Each value near the curve points indicates the parameters used for the model. For each method, the red squares present the minimum prediction error and the yellow triangles represent the best trade-off between DD and FPR (i.e., the nearest points to the perfect case, both, DD and FPR, equal to zero).

According to the analysis carried out in [25], the most efficient method is the one whose the curve is nearer to the coordinate (0, 0). This figure shows that the method IDPSO-ELM-S (30) presented the better results in terms of concept drift detection, substantially outperforming the methods ELM-DDM e ELM-FEDD and slightly improving the results in comparison to the ELM-ECDD.

Figure 5 shows the influence of the number of sensors on the number of false alarms and a predefined error tolerance parameter $c = 0.75$. The base detector is the case where the number of sensors is equal to 1 (depicted in the first bar). It is possible to conclude that increasing the number of sensors decreases the number of false alarms. Furthermore, the drift delay (DD) is not affected. This confirms the hypothesis that monitoring several swarm particles is effective for improving the number of false positive cases.

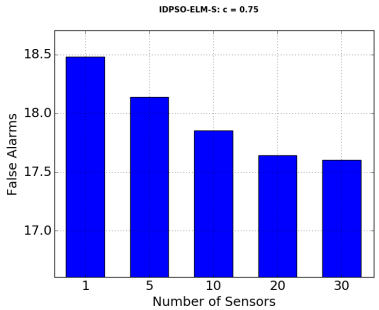

Fig. 5. Influence of varying the number of sensors in the false alarms number.

It is important to note that the proposed method based on swarm behavior (IDPSO-ELM-B) did not yield a good detection curve. It can be explained by the fact that the IDPSO-ELM-B result is based in the average outcome of the swarm. This approach consider good solutions, however, gives the same importance to bad solutions in the swarm (particles), i.e., particles presenting a high error level may substantially affect the final result. On the other hand, the methods ELM-ECDD and ELM-DDM monitor the error only for one model that is supposed to be a good model.

B. Time Series Forecasting Ability in the Presence of Concept Drift Evaluation

This experiment is aimed at validating the hypothesis that using the concept drift detection improves the forecasting error given that once a concept has changed it is possible to generate new, and more suitable, models for the new concept. Table II presents the results for the models with better forecasting accuracy. This is the models represented by red squares in figure 4. The best values, statistically tested, are presented in boldface.

According to Table II, IDPSO-ELM-S(1) overcomes ELM-ECDD (Figure 6 can better depict this performance). The only difference between the methods is the use of swarms for training the IDPSO-ELM-S(1). So, this demonstrates the benefit of using swarms. Furthermore, this table also shows that method IDPSO-ELM-B presented the best average result. This can be explained by the low delay. As the concept drift is detected earlier the model can be updated faster, thereby avoiding using an outdated model for data of an unknown concept. In addition, having high false alarms is outweighed by the fact that the old model continues to be used until there are new examples to train a new model. This enables the proposed method to be robust to false alarms.

The Friedman test was conducted to confirm the results and according to it, with a p-value of $2.57e{-}28$ the null hypothesis is rejected, i.e., the methods are statistically different. Furthermore, the Nemenyi test, presented in Figure 6, confirms that the proposed methods are statistically better than the ELM-ECDD, ELM-DDM and ELM-FEDD.

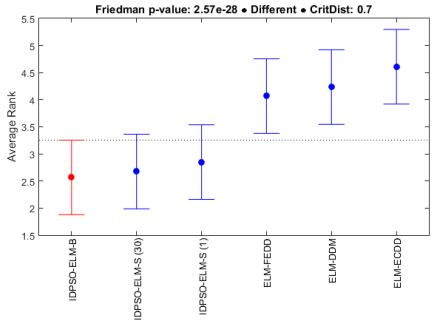

Fig. 6. Nemenyi test for the best models obtained for the artificial datasets.

C. Time Series Forecasting Ability for Real Time Series Evaluation

Table III presents the results obtained for the real time series where the best results are in boldface. According to this table,

TABLE II
MEAN ABSOLUTE ERROR FOR THE ARTIFICIAL DATASETS

Parameter - best MAE						
Dataset	**ELM-FEDD**	**ELM-DDM**	**ELM-ECDD**	**IDPSO-ELM-B**	**IDPSO-ELM-S (1)**	**IDPSO-ELM-S (30)**
Linear	0.02 (0.006)	0.02 (0.006)	0.024 (0.01)	**0.018 (0.003)**	0.019 (0.003)	0.02 (0.004)
Non Linear	0.045 (0.007)	0.045 (0.002)	0.045 (0.002)	0.044 (0.002)	0.044 (0.002)	**0.043 (0.002)**
Seasonal	0.066 (0.008)	0.066 (0.002)	0.067 (0.002)	0.064 (0.002)	0.064 (0.002)	**0.064 (0.001)**
Hybrid	0.066 (0.009)	0.056 (0.034)	0.07 (0.052)	**0.034 (0.008)**	0.04 (0.015)	0.04 (0.014)

the proposed methods outperform the remaining methods in terms of Mean Absolute Error (MAE). This result is confirmed by the Friedman test with p-value $1.32e-72$ which indicates that the methods are different. Furthermore, Figure 7 details how different the methods are. The Figure 7 indicates that the methods IDPSO-ELM-B and IDPSO-ELM-S (30) statistically presented better results than the methods ELM-ECDD, ELM-DDM, ELM-FEDD and ELM.

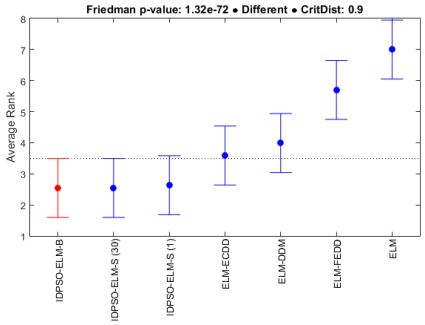

Fig. 7. Nemenyi test for the best models obtained for the real datasets.

Figure 8 depicts some of the problems faced by the methods in the real time series. The method ELM suffered from the first problem in the figure. In this case, the method did not treat the concept drift and hence is not updated. Therefore, the model becomes obsolete. This problem can be seen as an extreme case of miss-detections.

The problem 2 in Figure 8 depicts a case where a very subtle concept drift takes place. In this case, the model remains effective, i.e., the error rate remains acceptable. Such situations may difficult the task of concept drift detection for methods based on the model error, however, if the model remains efficient it is not necessary to perform a new training phase.

A known issue about the ELM-FEDD is that it uses the FEDD to monitor the concept drift phenomenon. Unlike the other considered approaches, the FEDD is not concerned about the model error, it is based solely on the time series values to tackle the concept drift problem. It presents a good accuracy regarding the concept drift detection, however, the model may become inadequate even if a concept did not change. The problem 3 in Figure 8 depicts this case. The detection module is able to detect the concept changes, however, the model degradation is not considered.

Another problem depicted in Figure 8 is the high rate of false positive cases (problem 4). Usually, detection modules based on the error such as ECDD and DDM are based only on the error level to monitor a change of concept. This may often not reflect a real concept change, since detection is dependent on the level given to the detection parameter, which can result in many false alarms. Assigning to this task an individual analysis of various models as in the IDPSO-ELM-S method may result in more robust decisions regarding the false alarm rate.

VI. CONCLUSION

This work has introduced two PSO-based approaches for time series forecasting for problems with concept drift. With this aim, ELM neural networks partially trained by the IDPSO swarm intelligence method are used as regressors models. The methods explicitly inform the system when a concept drift occurs so that the regressors models can be updated with the new concept. The first method is called IDPSO-ELM-B (behavior) and monitors statistics of the whole swarm. The second one is called IDPSO-ELM-S (sensors) and monitors statistics of a set of best particles.

The experiments reported in this paper compared the proposed methods with other state-of-the-art methods, such as ELM-ECDD and ELM-DDM and also to another method developed in this work for comparison purposes, called ELM-FEDD. The results showed that the proposed method IDPSO-ELM-S(30) yielded the best concept drift detection accuracy and a good predictive accuracy.

In addition, some important aspects were discovered and we can conclude that: (i) using several sensors improves the FPR rate without affecting the DD rate; (ii) the best concept drift detection method does not necessarily implicates in the best forecasting performance; (iii) low drift delay rates result in better predictions, since the model detects the change earlier and can update itself faster; and, (iv) using swarms to train the weights of the first ELM network layer improves the forecasting accuracy in relation to train using the canonical method (i.e., randomly assignment of values).

Two possible future works are (i) using the entire swarm in order to create an ensemble of forecast models and (ii) storing old forecast models to consider recurring concepts and avoid unnecessary retraining.

REFERENCES

[1] A. L. Oliveira and S. R. Meira, "Detecting novelties in time series through neural networks forecasting with robust confidence intervals," *Neurocomputing*, vol. 70, no. 1, p. 7992, 2006.

TABLE III
MEAN ABSOLUTE ERROR FOR THE REAL DATASETS

Dataset	ELM	ELM-FEDD	ELM-DDM	ELM-ECDD	IDPSO-ELM-B	IDPSO-ELM-S (1)	IDPSO-ELM-S (30)
Dow Jones	2.5206 (2.3946)	0.0495 (0.0214)	0.0112 (0.003)	0.0119 (0.003)	**0.0098 (0.0029)**	0.0102 (0.0024)	0.0104 (0.0024)
NASDAQ	4.374 (3.2291)	1.1112 (1.082)	0.0082 (0.0022)	0.0076 (0.0013)	0.0071 (0.0028)	0.0064 (0.0015)	**0.006 (0.002)**
S&P 500	16.2136 (12.0193)	0.5796 (0.5999)	0.0051 (0.0013)	0.0055 (0.0011)	0.0053 (0.0014)	0.0064 (0.002)	**0.0051 (0.001)**

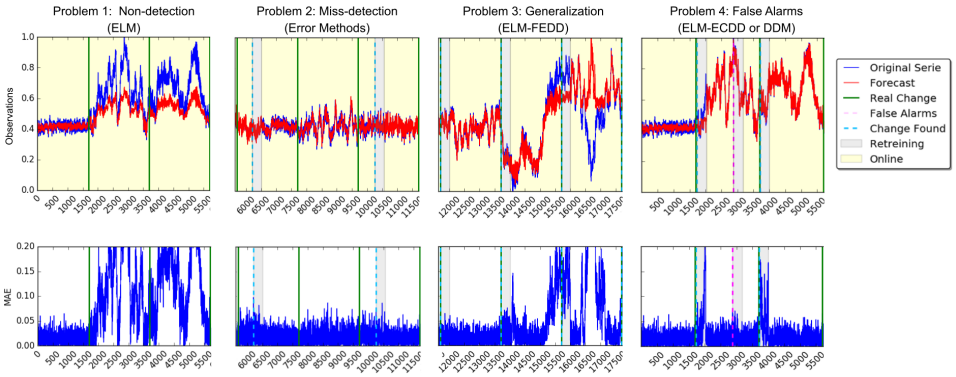

Fig. 8. Examples of the problems likely to occur in time series forecasting in the presence of concept drift. The upper blocks depict the time series and the predicted values. The remaining blocks show the error behavior for each respective upper block.

[2] R. C. Cavalcante and A. L. Oliveira, "An autonomous trader agent for the stock market based on online sequential extreme learning machine ensemble," in *Int, Joint Conf. on Neural Networks*. IEEE, 2014, p. 14241431.

[3] R. F. de Brito and A. L. Oliveira, "Sliding window-based analysis of multiple foreign exchange trading systems by using soft computing techniques," in *International Joint Conference on Neural Networks*. IEEE, 2014, p. 42514258.

[4] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 44, 2014.

[5] I. Zliobaite, A. Bifet, B. Pfahringer, and G. Holmes, "Active learning with drifting streaming data," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 1, pp. 27–39, 2014.

[6] J. Gama, "A survey on learning from data streams: current and future trends," *Progress in Artificial Intelligence*, vol. 1, no. 1, pp. 45–55, 2012.

[7] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in non-stationary environments: A survey," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 12–25, 2015.

[8] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Brazilian Symposium on Artificial Intelligence*. Springer, 2004, pp. 286–295.

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

[10] R. C. Cavalcante and A. L. Oliveira, "An approach to handle concept drift in financial time series based on extreme learning machines and explicit drift detection," in *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015, pp. 1–8.

[11] Y. Zhang, X. Xiong, and Q. Zhang, "An improved self-adaptive PSO algorithm with detection function for multimodal function optimization problems," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[12] G. Boracchi and M. Roveri, "Exploiting self-similarity for change detection," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 3339–3346.

[13] R. C. Cavalcante, L. L. Minku, and A. L. Oliveira, "FEDD: Feature Extraction for Explicit Concept Drift Detection in time series," in *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016, pp. 740–747.

[14] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.

[15] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 6, pp. 1–24, 2012.

[16] X. Hu and R. C. Eberhart, "Adaptive particle swarm optimization: detection and response to dynamic systems," in *Evol. Computation, 2002. CEC'02. Proc. of the 2002 Congress on*, vol. 2. IEEE, 2002, pp. 1666–1670.

[17] H. Richter, "Detecting change in dynamic fitness landscapes," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 1613–1620.

[18] N. Fouladgar and S. Lotfi, "A novel approach for optimization in dynamic environments based on modified cuckoo search algorithm," *Soft Computing*, vol. 20, no. 7, pp. 2889–2903, 2016.

[19] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer for noisy and dynamic environments," *Genetic Programming and Evolvable Machines*, vol. 7, no. 4, pp. 329–354, 2006.

[20] B. Nasiri, M. Meybodi, and M. Ebadzadeh, "History-driven particle swarm optimization in dynamic and uncertain environments," *Neurocomputing*, vol. 172, pp. 356–370, 2016.

[21] A. Rakitianskaia and A. P. Engelbrecht, "Training neural networks with pso in dynamic environments," in *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*. IEEE, 2009, pp. 667–673.

[22] A. S. Rakitianskaia and A. P. Engelbrecht, "Training feedforward neural networks with dynamic particle swarm optimisation," *Swarm Intelligence*, vol. 6, no. 3, pp. 233–270, 2012.

[23] F. Han, H.-F. Yao, and Q.-H. Ling, "An improved evolutionary extreme learning machine based on particle swarm optimization," *Neurocomputing*, vol. 116, pp. 87–93, 2013.

[24] R. C. Cavalcante, R. C. Brasileiro, V. L. Souza, J. P. Nobrega, and A. L. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Expert Systems with Applications*, vol. 55, pp. 194–211, 2016.

[25] C. Alippi, G. Boracchi, and M. Roveri, "Hierarchical change-detection tests," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 2, pp. 246–258, 2017.

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