Neurocomputing 180 (2016) 27–34

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A hybrid evolutionary decomposition system for time series forecasting

João F.L. de Oliveira [a,b,*], Teresa B. Ludermir [a]

[a] Centro de Informática (CIn), Universidade Federal de Pernambuco (UFPE), Pernambuco, Brazil
[b] Universidade de Pernambuco (UPE), Pernambuco, Brazil

ARTICLE INFO

Article history:
Received 28 March 2015
Received in revised form
10 July 2015
Accepted 10 July 2015
Available online 10 November 2015

Keywords:
Time series forecasting
Hybrid intelligent systems

ABSTRACT

In the field of time series forecasting, the combination of linear and nonlinear models has been explored to improve the accuracy of predictions when compared with the performance of individual models. Traditional forecasting methods such as the autoregressive integrated moving average (ARIMA) and exponential smoothing (ETS) are employed to map linear patterns in a time series while neural networks (ANNs) and support vector machines (SVMs) map nonlinear patterns. A traditional hybrid ARIMA–ANN system works by performing linear relationships in the data, considering the residual error a nonlinear component which is mapped by the ANN. The nature of a time series can be taken into consideration before applying a model. This work employs both linear and nonlinear patterns of a time series based on its volatility. This is explored using a hybrid evolutionary system composed by a simple exponential smoothing filter, ARIMA and autoregressive (AR) linear models and a SVR model. Particle swarm optimization is employed to optimize the order of the AR model, SVR parameters and number of lags of the time series. Experimental results show that the evolutionary hybrid system presented promising results in the forecasting domain.

1. Introduction and related work

Time series forecasting has become an important task in the field of machine learning and has applications in diverse areas including stock market, hydrology and business. Through the study of historical observations of a given variable it is possible to model its underlying process. A time series is a sequence of data points correlated in time and driven by a dynamical system. Univariate time series can be represented by a vector $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$, where y_t is the value measured at time-step t and n is the total number of observations.

Traditional linear models such as the autoregressive integrated moving average (ARIMA) [1] assume that a time series is composed by linear relationship of past data values and errors. In the same way, future values are also considered to be a realization of current and past observations and error terms. An ARIMA model has an autoregressive (AR) component and a moving average (MA) component which take into consideration past observations and past error terms respectively. Box and Jenkins [1] proposed a procedure to determine a suitable number of past observations and errors, often called the order of an ARIMA model. Since real-world time series are often nonlinear [2], it is not advised to assume that the time series is generated by a linear process. ARIMA model can model linear patterns in a time series, however its application in the presence of nonlinear patterns may not be adequate due to this limitation.

Artificial neural networks (ANNs) [3] and support vector regression (SVR) [4] are very important nonlinear techniques in the time series forecasting field. They can perform nonlinear function approximations and also generalize to future data. In order to obtain accurate models for a given problem, parameter setting plays a crucial role. A simple approach for parameter setting is based on trial and error, however underfitting may occur if the parameter space is not large enough or it can be a repetitive time-consuming task. Another approach is to employ optimization techniques including particle swarm optimization (PSO) [5] and genetic algorithms (GA) [6] to search for the best set of parameters. Furthermore, the application of nonlinear techniques in linear time series may produce mixed results [7], thus it is not advised to apply nonlinear techniques in any kind of data without previous analysis.

To overcome natural limitations from the aforementioned class of models and to improve the forecasting accuracy for time series, the combination of linear and nonlinear models in the same problem may be employed, since it is difficult to completely know *a priori* the characteristics of the real-world data in terms of linearity of patterns. In addition, a single model may not be adequate for

* Corresponding author.
E-mail address: fausto.lorenzato@upe.br (J.F.L. de Oliveira).

http://dx.doi.org/10.1016/j.neucom.2015.07.113
0925-2312/

every problem, since capturing different patterns equally well is a complex task. In order to address this problem, Zhang [8] proposed a hybrid ARIMA–ANN model to map different patterns in a time series by first capturing linear patterns using the ARIMA model and then employing an ANN on the residual errors. In this sense, the time series is considered a linear composition of linear and nonlinear patterns. Combinations employing ARIMA and SVR models were investigated by Pai and Lin [9], Xuemei et al. [10] and Zhu and Wei [11] in different applications of time series forecasting. Khashei and Bijari [7] explored the decomposition of the time series as a function composed by linear and nonlinear patterns. First an ARIMA model was employed and the residuals along with the original series were used in the ANN model. Babu and Reddy [12] used the nature of a time series to perform a decomposition of a time series in low-volatility and high-volatility components.

Even though these hybrid models are able to outperform each individual component, some studies report inconsistent results [7] due to assumptions made in the model specification stage, resulting in different approximations to the data generating process of a time series. For a given ANN architecture, there are many initial configurations of weights, therefore after the learning process is completed the final configuration may yield a high-bias network [13]. To increase the accuracy of forecasts, hybrid evolutionary systems have been employed. Hsieh et al. [14] used a particle swarm optimization (PSO) [15] approach to select SVR parameters in financial time series forecasting. Alwee et al. [16] used a hybrid ARIMA-SVR system optimized by two executions of the PSO algorithm, one for the selection of ARIMA parameters and the other for selection of SVR parameters. Huand and Dun [17] proposed a distributed PSO-SVM system with feature selection applied in classification problems.

Also an important parameter in time series forecasting is the number of past observations (dimension of lags) used in the construction of the model. The theorem of Takens [18] showed that for a given time series it is possible to model the series using a sufficient large dimension of lags. The dimension size m and the lag τ have to be determined previously with the purpose of improving the quality of the model. Some methods in the literature employ automatic lag selection [19–23] however there is not a single best method for all situations. From an ANN point of view, the input lag structure along with the number of hidden nodes and connection weights are essential for forecasting accuracy [13]. The SVR is very sensitive to its hyper-parameters setting, thus a set of hyper-parameters must be found and the complexity of the training model can be reduced with a suitable lag structure [24].

In this paper a new hybrid evolutionary ARIMA-SVR model is proposed based not only on the decomposition of the time series in linear and nonlinear patterns but also taking into consideration the nature of a time series. A simple exponential smoothing (ETS) filter is employed to decompose the time series into low-volatility and high-volatility components, then an ARIMA model is applied to the low-volatility component, and the high-volatility component is decomposed into linear and nonlinear patterns by an AR-SVR hybrid model. The parameter selection and lag selection is performed by a PSO algorithm. The final forecast will be the summation of the predictions from each component, resulting in a more general and accurate model. Experiments are performed on 12 datasets from the Time Series Data Library [25] and one electricity price data from the Australian National Electricity Market [12]. The results show that the proposed method outperform the models of Babu and Reddy [12], Oliveira and Ludermir [23] and individual models (ARIMA, and Exponential Smoothing).

The remainder of this paper is organized as follows: Section 2 presents a brief description of some techniques employed in this work. In Section 3 the proposed method architecture is described. The experimental setup is presented in Section 4 and the conclusions and final remarks are presented in Section 5.

2. Literature review

Some methods used in this work are briefly described in this section.

2.1. Support vector regression

The SVR method was proposed by Vapnik [26] and it is based on the structured risk minimization (SRM) principle which performs the minimization of the upper bound of the generalization error through the use of a subset of data points called support vectors. Let $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$ be a training set where $\mathbf{x} \in \Re^d$ is the ith input vector, $y_i \in \Re$ is the ith prediction output of x_i, d is the embedding dimension of the time series and l is the number of training samples. The objective of SVR is to find the best function from a set of possible functions in the form:

$$\{f \mid f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b, w \in \Re^d, b \in \Re\} \tag{1}$$

where w is the weight vector estimated by minimizing the regularized risk function showed in Eq. (2) and $\mathbf{b}$ is a bias or threshold

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l} L(y_i, f(\mathbf{x}_i)) \tag{2}$$

In order to find the best function f, it is necessary to minimize the regularized risk, where $C > 0$ is a regularization factor, $\|\cdot\|$ is a 2-norm and $L(\cdot,\cdot)$ is a loss function. In order to induce sparsity in SVR the ϵ-insensitive loss function is presented in Eq. (3), which creates an ϵ-tube allowing some predictions to fall within the boundaries of this ϵ-tube:

$$L(y, f(\mathbf{x})) = \begin{cases} 0 & |f(\mathbf{x}) - y| < \epsilon \\ |f(\mathbf{x}) - y| - \epsilon & \text{otherwise} \end{cases} \tag{3}$$

With the ϵ-insensitive function the SVR is formulated as shown in the following equation:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}(\xi_i - \xi_i^*) \\ &\text{subject to} \quad \begin{cases} \mathbf{w}^T\mathbf{x_i} + b - y_i \le \epsilon + \xi_i \\ y_i - \mathbf{w}^T\mathbf{x_i} - b \le \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0, \quad i = 0 \ldots l \end{cases} \end{aligned} \tag{4}$$

where ξ and ξ^* are slack variables used to measure the errors occurred by values outside the boundaries determined by the ϵ-tube.

The use of kernels allows the SVR to perform nonlinear mappings into a higher dimensionality space. The regression function takes the form

$$f(\mathbf{x}) = \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)k(\mathbf{x}_i, \mathbf{z}) + b \tag{5}$$

where α and α^* are Lagrange multipliers and $k(\mathbf{x}_i, \mathbf{x})$ is a kernel function. The Gaussian kernel is widely employed in time series forecasting [23,27] ($k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\gamma^2}\right)$), where γ is a parameter of the Gaussian kernel. SVR can produce an unique global minimum solution [24] and depending on the kernel type it can model linear and nonlinear time series.

2.2. Particle swarm optimization

The PSO is an intelligent search algorithm inspired in the social behavior of individuals. The search process is conducted by each

moving particle which represent a possible solution for the problem. Each particle maintains a record of its best solution in a vector called **pbest**, and the best solution among all particles is called **gbest**. The particles move in the direction of the global best solution and its personal best solution.

To perform movements each particle calculates its velocity and updates its position for each iteration. Consider $pbest_i$, $gbest$ and g_i vectors representing the ith particle personal best solutions, global best solution and current solution. The velocity v_i is updated as follows:

$$\mathbf{v}_i(t+1) = w \times \mathbf{v}_i(t) + c_1 \times rnd() \times (\mathbf{pbest}_i - \mathbf{g}_i(t)) + c2 \times rnd() \times (\mathbf{gbest} - \mathbf{g}_i(t)) \quad (6)$$

where $\mathbf{v}_i \in [-v_{max}, v_{max}]$, $rnd()$ is a random function in the range [0, 1], the c_1 and c_2 constants represent the personal and social learning factors respectively and w is the inertia weight. The new solution is calculated as follows:

$$\mathbf{g}_i(t+1) = \mathbf{g}_i(t) + \mathbf{v}_i(t+1) \quad (7)$$

The discrete PSO was proposed by Kennedy and Eberhart [28] to operate on binary search spaces. Each dimension of particle $\mathbf{x_i}$ is a binary value ($\mathbf{g_i} \in \mathbb{B}^k$).

The calculation of the velocity showed in Eq. (6) remains unchanged, except that $\mathbf{g}_i$, $\mathbf{pbest}_i$ and **gbest** belong to the binary domain and $\mathbf{v}_i$ must be constrained to the [0, 1] interval as well. The constriction is obtained by using a sigmoid function $Sig(\mathbf{v}_i)$, and a new position is calculated as follows:

$$\text{if } \text{rnd}() < Sig(\mathbf{v}_i), \text{ then } \mathbf{g}_i = 1; \text{ else } \mathbf{g}_i = 0; \quad (8)$$

where $Sig(\mathbf{v_i}) = \frac{1}{1+e^{-\mathbf{v_i}}}$.

2.3. Hybrid models

The combination of models has become a common practice in the time series forecasting field. Real-world data is often composed by linear and nonlinear patterns, thus it is reasonable to perform a combination between linear and nonlinear models. Such combination is achieved by using a linear model to capture linear patterns in the data and the residual error is considered to be nonlinear. In the literature some approaches have been proposed in order to increase the accuracy of predictions.

Zhang [8] considered a given time series y_t to be a composition of linear and nonlinear patterns as shown in Eq. (9), where L_t is the linear component and N_t is the nonlinear component:

$$y_t = L_t + N_t, \quad (9)$$

In his approach, an ARIMA model was employed to map linear patterns and the residual error e_t was obtained by subtracting the ARIMA predictions $\hat{L}_t$ from the actual data y_t as shown in the following equation:

$$e_t = y_t - \hat{L}_t \quad (10)$$

The residual error produced is assumed to have only nonlinear correlations, thus an ANN is employed to perform nonlinear mappings in the residual data. The final prediction ($\hat{y}_t$) is considered to be the summation of both predictions from the ARIMA ($\hat{L}_t$) and the ANN ($\hat{N}_t$) models, with respect to the linear and nonlinear patterns as shown in the following equation:

$$\hat{y}_t = \hat{L}_t + \hat{N}_t \quad (11)$$

Oliveira and Ludermir [23] also assumed a composition of time series as presented in Eq. (9), however a PSO algorithm was employed to determine the order of the ARIMA model, the selection of lags and the SVR parameters simultaneously. A hybrid PSO with discrete and continuous codifications was used to find the best set of parameters for the system. The number of lags was fixed, and the most relevant lags among the sequence were selected to build the SVR model. Linear and nonlinear patterns were mapped by an ARIMA model and SVR respectively.

Babu and Reddy [12] considered the volatility of a time series as a criteria for its decomposition in low-volatility and high-volatility components. According to [12] if a given stationary time series is truly Gaussian, the ARIMA model could provide a better fit. In agreement with the Jarque–Bera normality test, if a kurtosis value of a time series is 3 then the data is Gaussian and is considered to be low-volatile which is appropriate for the application of an ARIMA model. Sequences which not obtained a kurtosis value of 3 were considered to be high-volatile and were modeled by an ANN. Thus a time series is considered to be a sum of a low-volatility (o_t) component and a high-volatility component (h_t) as is presented in the following equation:

$$y_t = o_t + h_t \quad (12)$$

The kurtosis of a series is calculated as shown in the following equation:

$$kurtosis(y) = \frac{E((y - E(y))^4)}{(E((y - E(y))^2))^2}, \quad (13)$$

A moving average (MA) filter (Eq. (14)) was used to separate the time series y_t in two components such that one is less volatile and the other is highly volatile. The length q of the MA filter is adjusted each iteration until the kurtosis has a value of 3 and the decomposition is achieved. The time series resulting from the application of the MA filter is denoted as y_{ma}:

$$y_{ma} = \frac{1}{q} \sum_{i=t-q+1}^{t} y_i \quad (14)$$

The final forecasting $\hat{y}_t$ would be the sum of prediction from each model as presented in Eq. (15), where $\hat{o}_t$ and $\hat{h}_t$ are the predictions of the low-volatility and high-volatility components respectively

$$\hat{y}_t = \hat{o}_t + \hat{h}_t \quad (15)$$

3. Proposed method

The hybrid method proposed by Zhang [8] assumes that a time series can be decomposed in linear and nonlinear patterns. Babu and Reddy [12] considered the volatility of a time series as a criteria for its decomposition in low-volatility and high-volatility components. The moving average filter used in [12] has some drawbacks such as requiring more storage due to all of the q latest observations and applying same weights to the same latest data. Hence if the order (q) of the model is large, past observations will have the same importance as the latest observations [29]. Moreover the system lacks a refined search for model parameters, which could produce more accurate models.

Statistically, if a time series is Gaussian stationary an ARIMA model is more suitable for time series data [12]. The non-Gaussian component may present a richer data structure [30] and may be considered nonlinear [31]. However non-Gaussian patterns may also be captured by linear AR(1) models with the replacement of the innovation distribution by a non-Gaussian distribution [32] as shown in the following equation:

$$h_t = \phi h_{t-1} + \varepsilon_t \quad (16)$$

where ε_t is some non-Gaussian distribution. In this work we consider that a time series may be decomposed into a low-volatility component o_t and a high-volatility component h_t as shown in Eq. (12), in addition the high-volatility component can be further decomposed into linear a_t and nonlinear patterns b_t (Eq.

(17)). This procedure is also described in lines 2 – 4 in Algorithm 1

$$h_t = a_t + b_t \tag{17}$$

Linear component a_t is modeled as a function of its past values using an AR model of order p called AR(p) shown in Eq. (18) (lines 10 – 14). The errors (b_t) produced by the procedure shown in Eq. (19) are used as the nonlinear input for a SVR model which is going to be a function of its k past values (Eq. (20)) (lines 15–16)

$$\hat{a}_t = f(a_{t-1}, a_{t-2}, \ldots a_{t-p}) \tag{18}$$

$$b_t = h_t - \hat{a}_t \tag{19}$$

$$\hat{b}_t = g(b_{t-1}, b_{t-2}, \ldots, b_{t-k}) \tag{20}$$

The final forecast $\hat{y}_t$ will be the sum of the forecast from each component as shown in the following equation (lines 17–20):

$$\hat{y}_t = \hat{o}_t + \hat{a}_t + \hat{b}_t, \tag{21}$$

where $\hat{o}_t$ is the prediction of the low-volatility component using an ARIMA model (lines 6–9), and $\hat{a}_t$ and $\hat{b}_t$ are the linear and nonlinear predictions respectively, from the high-volatility component.

The decomposition of the time series in terms of volatility is achieved through the use of a simple exponential smoothing filter $s_t = \alpha y_t + (1-\alpha)s_{t-1}$ [29], where α is the smoothing factor and needs to be tuned in order to achieve Gaussian and non-Gaussian series. The low-volatility (Gaussian) component is modeled by an ARIMA(p',d,q'). An AR(p) is applied directly to the high-volatility component and the residual error is considered to be the nonlinear component.

Most relevant lags from the error produced in the previous step are selected and then modeled by a SVR model. Time series forecasting is based on past values in order to perform predictions, thus the behavior of past data must be taken into consideration. Let $y_t = \{4, 2, 5, 3, 6, 4, 7\}$ be a time series, the procedure converts a scalar time series into a vector $u_t \in \Re^k$ for some integer k, where k is the number of lags. Let the number of lags $k = 4$ and the lag mask $V = \{0, 1, 0, 1\}$, the prediction of the fifth value (6) will take into consideration four past values, however only two values were selected $\{2, 3\}$. The proposed hybrid system architecture is summarized in Fig. 1.

3.1. Parameter optimization

The proposed hybrid system performs two different optimizations, one for the ARIMA model and the other for the AR and SVR models. In order to find a suitable order for the ARIMA model a step-wise procedure is employed [33] to explore the model space more efficiently.

The optimization of high volatility models is performed through the use of a PSO algorithm in a mixed optimization fashion regarding the discrete and continuous space of the problem. The particle codification is presented in Fig. 2.

The PSO algorithm has to perform two different tasks: (i) to find a suitable value of α in order to produce low-volatility and high-volatility series and (ii) to improve the accuracy of the AR-SVR system with both parameter and lag selection. The performance of each particle is measured through a fitness function which is defined in the following equation:

$$fitness_i = (|3 - kurtosis(\mathbf{y})|) + \frac{1}{m}\sum_{j=1}^{m}(h_t - \hat{h}_t^i)^2 \tag{22}$$

The problem was formulated to be a minimization of the fitness function, in which the minimization of the first term will produce a Gaussian series and the second term calculates the mean square error (MSE) of the high-volatility term.

3.2. Computational complexity

A common practice in complexity theory is to concentrate on the worst-case performance of the algorithm [34].

Algorithm 1. The proposed method.

1: Given a time series y_t, $t \in [1 \ldots n]$
2: **for** $t = 1$ to n **do**
3: $o_t = \alpha y_t + (1-\alpha)s_{t-1}$
4: $h_t = y_t - o_t$
5: **end for**
6: Determine the ARIMA(p',d,q') parameters
7: **for** $t = 1$ to n **do**
8: $\hat{o}_t = \phi_0 + \phi_1 o_{t-1} + \phi_2 o_{t-2}, \ldots, \phi_{p'} o_{t-p'} + \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2}, \ldots, \theta_{q'} \varepsilon_{t-q'}$
9: **end for**
10: Calculate $AR(p)$ parameters
11: **for** $t = 1$ to n **do**
12: $\hat{a}_t = \phi_1' h_{t-1} + \phi_2' h_{t-2} + \cdots + \phi_p' h_{t-p}$
13: $b_t = h_t - \hat{a}_t$
14: **end for**
15: Perform lag selection in the predefined k lags.
16: Perform SVR training with the parameters C, ϵ, γ.
17: **for** $t = 1$ to n **do**
18: $\hat{b}_t = SVR(b_{t-\tau_1}, \ldots, b_{t-\tau_r})$, where $r \leq k$
19: Produce final forecast $\hat{y}_t = \hat{o}_t + \hat{a}_t + \hat{b}_t$
20: **end for**

Through the analysis of the Algorithm 1, the exponential smoothing filter has a complexity of $O(n)$, where n is the number of data points in a given time series. The estimation of ARIMA order is carried out by a step-wise approach and the parameters estimation is performed by a MLE procedure over l parameters which has a complexity of $O(nl^2)$ [35], where l is the number of parameters for the ARIMA model, considering that the search process needs υ steps, thus the computational cost will be $O(\upsilon nl^2)$. The estimation of the AR model parameters has a computational complexity of $O(np^2)$, for estimating p parameters using MLE, where p is the order of the AR model. The lag selection procedure has complexity of $O(n)$. The LIBSVM [36] was employed for the implementation of SVR model and using k lags has a complexity of $O(nk)$. Reducing the number of lags, the complexity of the SVR model is decreased. The complexity of the proposed model will be $O(nk + \upsilon nl^2 + np^2)$, given all parameters.

However, the optimization of parameters requires an extra computational effort. The PSO search algorithm has a complexity of $O(MX)$ [37] considering X initial particles and M number of iterations. Each particle will essentially execute the proposed algorithm for M iterations (excluding the ARIMA estimation), producing a estimated complexity of $O(MX(nk + np^2) + \upsilon nl^2)$.

4. Experiments

The optimization of the ARIMA and ETS models was implemented using the Forecast package for **R** [33] and the remainder of the system was implemented in MATLAB. The experiments were performed along with the methods proposed by Babu and Reddy [12], Oliveira

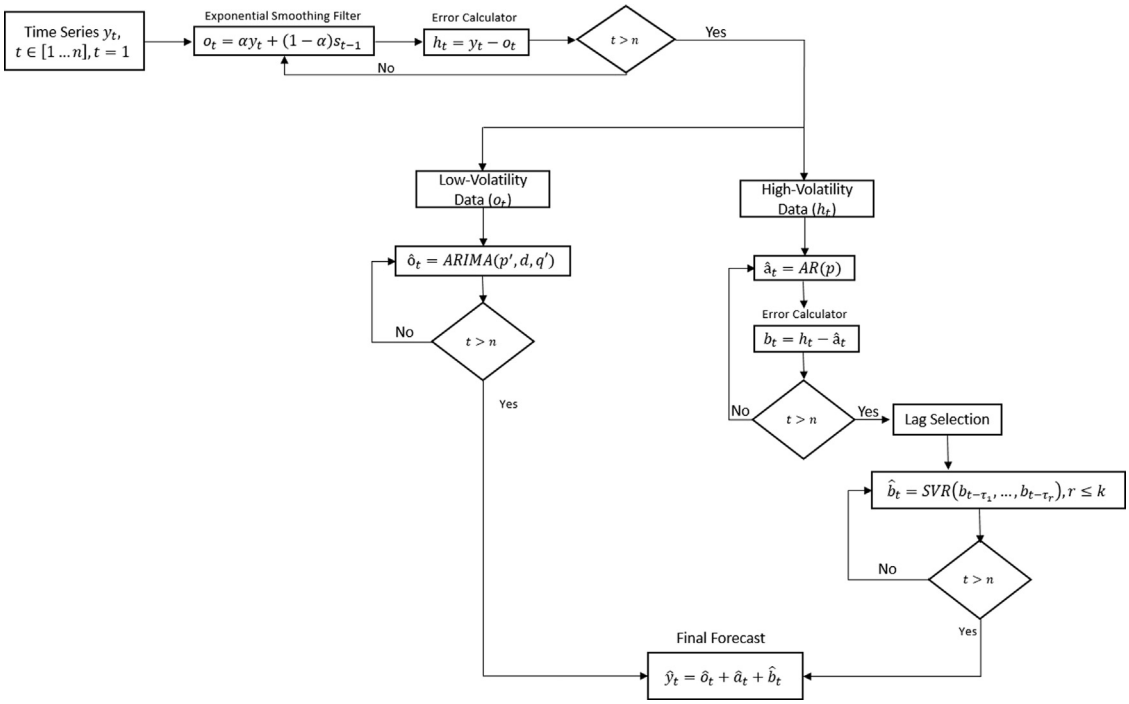


Fig. 1. Proposed hybrid system architecture.

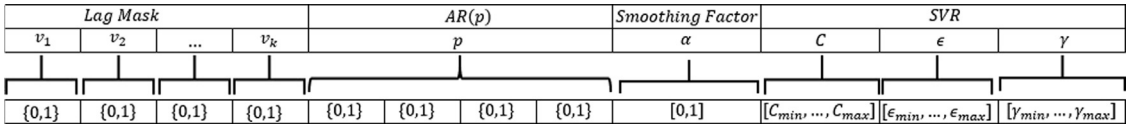


Fig. 2. Particle encodings.

and Ludermir [23], Zhang [8], ARIMA and Exponential Smoothing. In this study, 12 datasets from the Time Series Data Library [25] and one electricity price data from the Australian National Electricity Market [12] were used to assess the performance of the methods. The descriptions of the time series can be found in Table 1. Time series with trend patterns were first detrended and the trend patterns were modeled along with the high volatility component. Two performance measures were employed, which are the mean square error (MSE) and the symmetric mean absolute percentage error (SMAPE) presented in Eqs. (23) and (24):

$$MSE = \frac{1}{m}\sum_{j=1}^{m}(y_t - \hat{y}_t)^2 \tag{23}$$

The MSE is a very popular error measure in the time series forecasting domain, it can be used to guide the training phase of the forecasting methods. However its use is not advised for comparison of forecasting methods across different time series due to some limitations such as being sensitive to outlier data and being scale dependent [38]. The SMAPE has the advantage of being scale independent, providing an easy mechanism to perform comparisons across different time series

$$SMAPE = \frac{1}{m}\sum_{j=1}^{m}\frac{|y_t - \hat{y}_t|}{(|y_t| + |\hat{y}_t|)/2} \tag{24}$$

The number of lags k was determined through the analysis of the autocorrelation function (ACF) of each series. The number of neurons h for the methods of Zhang [8] and Babu and Reddy [12] was determined by performing executions in the range [3, 20] and the results present the configuration that achieved smallest MSE.

Table 1
Datasets description.

Dataset	Description
Passenger	International airline passengers
Stock	Annual common stock price, U.S
Colorado River	Annual Flows, Colorado River Lees Ferry
Dow-Jones	Dow-Jones Industrial Stock Price Index
Electricity	Electricity: electricity end use
IBM	IBM stock: daily
Lake Erie	Monthly Lake Erie Levels
Lynx	Annual number of lynx trapped
NSW	Daily Electricity Price
Pollution	Monthly shipment of pollution equipment
Wine	Monthly Australian wine sales
Sunspot	Wolfs Sunspot Numbers
Accidental death	Monthly Accidental deaths in USA

In the proposed method, the PSO algorithm performs a stochastic search to find the best set of parameters for the SVR model, the order of the AR model, the smoothing factor α and the best set of lags. The PSO algorithm is executed for 100 iterations, the population size was set to 30, the inertia weight (w), personal (c_1) and social (c_2) learning factors assume values stated in the simplified version [5] of the PSO algorithm $w=1$, $c_1=2$ and $c_2=2$ respectively. The search bounds for C, ϵ and γ are within the intervals $[10^4, 10^{-2}]$, $[10^{-1}, 10^{-8}]$ and $[10^3, 10^{-2}]$ respectively.

The methods are evaluated based on one-step-ahead predictions over 30 executions. Also, each series was divided in two subsets: in-samples and out-of-samples. The in-samples were employed in the model building phase, and in the evolutionary approaches this set is further divided into training and validation

sets, whereas the validation set is used to evaluate the model during the evolutionary process. The out-of-samples (test set) has the latest data and is used in the evaluation metrics, and the number of points used in each subset can be found in Table 2, along with the number of lags (k) used. Approximately 80% of the data was used in the in-samples and 20% for out-of-samples. All datasets were scaled in the [0, 1] range using Eq. (25), where $\min_y$ is the minimum value of the series y and $\max_y$ is the maximum value:

$$y' = \frac{y - \min_y}{\max_y - \min_y} \tag{25}$$

Table 2
Datasets division and number of lags.

Dataset	Training	Validation	Test	Number of Lags (k)
Passenger	92	29	23	12
Stock	65	20	14	10
Colorado River	452	149	143	12
Dow-Jones	516	165	143	50
Electricity	298	98	90	12
IBM	227	74	68	10
Lake Erie	366	120	114	12
Lynx	74	23	17	10
NSW	456	149	139	24
Pollution	84	26	20	12
Wine	117	38	32	12
Sunspot	178	58	52	11
Accidental death	48	15	9	12

Table 3
Mean square errors (MSE) for all datasets.

Dataset	Proposed	SVR-LS [23]	Babu [12]	Zhang [8]	ARIMA	ETS
Passenger	0.0016	**0.0013**	0.0020	0.0024	0.0094	0.0100
Stock	0.0073	0.0074	0.0904	0.0106	0.0123	**0.0064**
Colorado River	0.0030	0.0030	0.0029	0.0032	0.0048	**0.0028**
Dow Jones	**0.0005**	0.0014	0.0647	**0.0005**	**0.0005**	**0.0005**
Electricity	**0.0012**	0.0016	0.0024	0.0019	0.0059	0.0117
IBM ($\times 10^{-3}$)	0.6203	0.6072	0.8811	0.7095	0.7282	**0.5495**
Lake Erie	**0.0012**	**0.0012**	0.0018	**0.0012**	**0.0012**	0.0021
Lynx	0.0099	0.0168	0.0097	0.0063	**0.0054**	0.0115
NSW	**0.0018**	0.0020	0.0021	0.0025	0.0024	0.0028
Pollution	**0.0168**	0.0189	0.0215	0.0199	0.0208	0.0344
Wine	0.0201	0.0172	**0.0136**	0.0137	0.0325	0.0402
Sunspot	0.0148	0.0146	0.0147	0.0120	**0.0119**	0.0308
Accidental	**0.0034**	0.0073	0.0207	0.0048	0.0212	0.0215

Table 4
Individual results for the hypothesis test using the MSE. The results indicate which method is better, worse or equivalent to the proposed method using '+', '−' and '≈' respectively.

Dataset	SVR-LS [23]	Babu [12]	Zhang [8]	ARIMA	ETS
Passenger	≈	−	−	−	−
Stock	−	−	−	−	+
Colorado River	≈	+	≈	−	+
Dow Jones	−	−	−	−	+
Electricity	−	−	−	−	−
IBM	≈	−	−	−	+
Lake Erie	−	−	≈	−	−
Lynx	−	≈	+	+	−
NSW	−	−	−	−	−
Pollution	−	−	−	−	−
Wine	+	+	+	−	−
Sunspot	≈	≈	+	+	−
Accidental	−	−	−	−	−

The proposed model has higher computational complexity when compared to the other methods due to the parameter optimization stage. However less human intervention is needed and the predictions are more accurate. In fact, the only *a priori* knowledge about the data is the number of lags k needed for the SVR forecasts.

The analysis is divided in three parts: the first and second are based on the MSE and SMAPE error measures respectively and the last performs comparisons across all datasets.

4.1. Analysis based on MSE

The results with respect to the MSE are presented in Tables 3 and 4. Each value in Table 3 is the mean MSE of 30 executions of the experiment and the lowest values are presented in bold. In order to perform a thorough analysis, a Wilcoxon signed-rank test is applied to compare the proposed method

Table 5
Symmetric mean absolute percentage error (SMAPE) $\times 10^{-2}$ for all datasets.

Dataset	Proposed	SVR-LS [23]	Babu [12]	Zhang [8]	ARIMA	ETS
Passenger	0.0482	**0.0413**	0.0529	0.0586	0.1192	0.1282
Stock	0.1106	0.1122	0.4273	0.1237	0.1315	**0.0951**
Colorado River	0.4799	0.4597	0.4732	0.4988	0.6654	**0.4416**
Dow Jones	0.0244	0.0370	0.3210	0.0251	0.0245	**0.0241**
Electricity	**0.0374**	0.0424	0.0559	0.0489	0.0824	0.1293
IBM	0.1080	0.1079	0.1196	0.1165	0.1188	**0.1020**
Lake Erie	0.0578	0.0571	0.0766	**0.0570**	0.0579	0.0831
Lynx	0.6372	0.7577	0.7469	0.5877	**0.5818**	0.5829
NSW	**0.2943**	0.3135	0.3485	0.3861	0.3769	0.3705
Pollution	0.1868	**0.1854**	0.2036	0.2038	0.2197	0.2561
Wine	0.2287	0.2041	0.1991	**0.1979**	0.2382	0.3150
Sunspot	0.3300	0.3303	0.3450	**0.3289**	0.3602	0.4812
Accidental	**0.0918**	0.1456	0.3055	0.1149	0.2601	0.2645

Table 6
Individual Results for the Hypothesis test using the SMAPE. The results indicate which method is better, worse or equivalent to the proposed method using '+', '−' and '≈' respectively.

Dataset	SVR-LS [23]	Babu [12]	Zhang [8]	ARIMA	ETS
Passenger	+	−	−	−	−
Stock	−	−	−	−	+
Colorado River	+	≈	≈	−	+
Dow Jones	−	−	−	−	+
Electricity	−	−	−	−	−
IBM	≈	−	−	−	+
Lake Erie	+	−	+	−	−
Lynx	−	−	≈	+	+
NSW	−	−	−	−	−
Pollution	≈	−	−	−	−
Wine	+	+	+	−	−
Sunspot	≈	≈	≈	−	−
Accidental	−	−	−	−	−

Table 7
Ranks of all methods using MSE and SMAPE.

Method	Rank MSE	Rank SMAPE
Proposed	55.6154	58.1769
SVR-LS [23]	72.9231	66.6974
Babu [12]	104.0872	108.3205
Zhang [8]	81.5282	85.4974
ARIMA	110.8077	117.7308
ETS	118.0385	106.5769

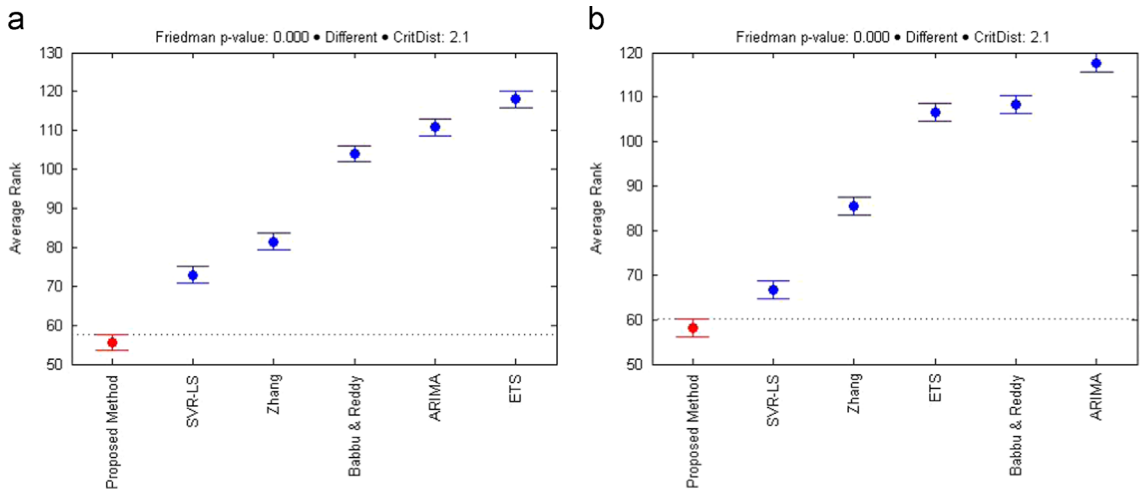


Fig. 3. Hypothesis testing for all datasets in terms of MSE (a) and SMAPE (b).

against the other methods employed in the experiment with a confidence level of 95%.

The results are presented in Table 4 indicating which method is better, worse or equivalent to the proposed method using '+', '−' and '≈' accordingly.

In accordance with the results, the proposed method outperformed all techniques in four cases (Electricity, NSW, Pollution and Accidental). The proposed method achieved better forecasts when compared to: SVR-LS [23] in eight cases; Babu [12] nine cases; Zhang [8] eight cases; ARIMA 11 cases; ETS nine cases.

4.2. Analysis based on SMAPE

The results are also analyzed under the SMAPE criterion which is very important in the time series forecasting domain. The same analysis performed in Section 4.1 is conducted using the SMAPE criterion. Table 5 presents the mean SMAPE over 30 executions of the methods. Employing an empirical analysis the proposed method only achieved lowest errors on three datasets, however when statistically compared with the other techniques some results are similar.

Table 6 shows that the proposed technique achieved lower SMAPE than the other techniques in most datasets.

4.3. Final analysis

Models were evaluated using Friedman and Nemenyi hypothesis tests [39]. Friedman's test was used to rank the algorithms for each data set separately and if the null hypothesis is rejected (ranks are significantly different) a Nemenyi test is employed [40] by comparing all algorithms with each other, performing pairwise tests. Consider a group of ω algorithms, μ datasets and a critical value q_α based on a studentized ranged statistics [40], if the average rank of each algorithm differ by at least a critical distance CD (Eq. (26)), we consider that their performance is significantly different

$$CD = q_\alpha \sqrt{\frac{\omega(\omega+1)}{6\mu}} \tag{26}$$

Using the significance value $\alpha = 0.05$, the value for q_α is 2.8497. A group of algorithms $\omega = 6$ is used and $\mu = 13$ datasets are employed, thus the computed critical distance $CD = 2.0911$, in terms of MSE and SMAPE.

Pairwise testing in Nemenyi test is performed by applying the differences of the ranks within each error measure. When comparing the proposed method with the SVR-LS [23] under the MSE, the absolute value of the difference ($dif = 17.3077$) of the ranks must be achieved and then compared with CD. Since $dif > CD$ then we can conclude that their difference is significant. Table 7 shows the ranks for the hypothesis testing.

Each measure ordered by rank is presented in Fig. 3. In general, the proposed method achieved the best results when compared to the other techniques employed in the experiments in both measures (MSE and SMAPE).

5. Conclusion

In this work a hybrid evolutionary decomposition system for time series forecasting was proposed. The system first decomposes the time series with respect to its low and high-volatility patterns with an exponential smoothing filter, and then decomposes the high-volatility term in linear and nonlinear patterns. It is assumed that there are linear and nonlinear patterns in the high volatility terms, thus an AR and SVR models are used respectively. The parameter optimization was performed through the employment of a PSO algorithm in order to produce more accurate forecasts. The proposed model achieved promising results increasing however the computational complexity in comparison with other techniques.

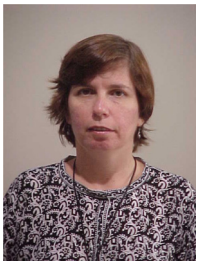
The experimental analysis was conducted using 13 datasets from the literature and six methods were considered: the proposed method, the PSO-ARIMA-SVR-LS [23], Zhang's method [8], Babu and Reddy's method [12] and an ARIMA and exponential smoothing (ETS) methods. The results showed that no method was the best for all datasets and that in some cases a linear model achieved the best results (lowest errors). Considering the datasets analysed, the proposed method obtained the best overall results in terms of accuracy of predictions and demonstrated promising results for future work. Friedman and Nemenyi hypothesis tests were employed to give a robust evaluation of results and indicated that the proposed method achieved the best rank among the methods.

In order to increase performance of the system, deep learning methods could be employed in order to model deep data structures [41]. Also, the exponential smoothing method could be further explored in a decomposition system due to its performance in the experiments.

References

[1] G.E. Box, G.M. Jenkins, G.C. Reinsel, Time Series Analysis: Forecasting and Control, Wiley.com, New Jersey, 2013.
[2] G. Zhang, B. Eddy Patuwo, M.Y. Hu, Forecasting with artificial neural networks: the state of the art, Int. J. Forecast. 14 (1) (1998) 35–62.
[3] S. Haykin, N. Network, A comprehensive foundation, Neural Netw. 2 (2004).
[4] H. Drucker, C.J. Burges, L. Kaufman, A. Smola, V. Vapnik, et al., Support vector regression machines, Adv. Neural Inf. Process. Syst. 9 (1997) 155–161.
[5] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference on Neural Networks, 1995, vol. 4, IEEE, Perth, WA, 1995, pp. 1942–1948.
[6] D.E. Goldberg, et al., Genetic Algorithms in Search, Optimization, and Machine Learning, vol. 412, Addison-wesley Reading Menlo Park, 1989, Boston, MA.
[7] M. Khashei, M. Bijari, A novel hybridization of artificial neural networks and ARIMA models for time series forecasting, Appl. Soft Comput. 11 (2) (2011) 2664–2675.
[8] G.P. Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing 50 (2003) 159–175.
[9] P.-F. Pai, C.-S. Lin, A hybrid ARIMA and support vector machines model in stock price forecasting, Omega 33 (6) (2005) 497–505.
[10] L. Xuemei, D. Lixing, D. Yuyuan, L. Lanlan, Hybrid support vector machine and ARIMA model in building cooling prediction, in: 2010 International Symposium on Computer Communication Control and Automation (3CA), vol. 1, IEEE, Tainan, 2010, pp. 533–536.
[11] B. Zhu, Y. Wei, Carbon price forecasting with a novel hybrid ARIMA and least squares support vector machines methodology, Omega 41 (3) (2013) 517–524.
[12] C.N. Babu, B.E. Reddy, A moving-average filter based hybrid ARIMA–ANN model for forecasting time series data, Appl. Soft Comput. 23 (0) (2014) 27–38.
[13] V. Berardi, G. Zhang, An empirical investigation of bias and variance in time series forecasting: modeling considerations and error evaluation, IEEE Trans. Neural Netw. 14 (3) (2003) 668–679.
[14] H.-I. Hsieh, T.-P. Lee, T.-S. Lee, A hybrid particle swarm optimization and support vector regression model for financial time series forecasting, Int. J. Bus. Adm. 2 (2) (2011) 48–56.
[15] A. Engelbrecht, Fundamentals of Computational Swarm Intelligence, vol. 1, Wiley, NY, 2005.
[16] R. Alwee, S.M. Hj Shamsuddin, R. Sallehuddin, Hybrid Support Vector Regression and Autoregressive Integrated Moving Average Models Improved by Particle Swarm Optimization for Property Crime Rates Forecasting with Economic Indicators, vol. 2013, The Scientific World Journal, 2013, p. 11, article ID 951475.
[17] C.-L. Huang, J.-F. Dun, A distributed PSO-SVM hybrid system with feature selection and parameter optimization, Appl. Soft Comput. 8 (4) (2008) 1381–1391.
[18] F. Takens, Detecting strange attractors in turbulence, in: Dynamical Systems and Turbulence, Warwick 1980, Springer, 1981, pp. 366–381.
[19] M. Shen, W.-N. Chen, J. Zhang, H.-H. Chung, O. Kaynak, Optimal selection of parameters for nonuniform embedding of chaotic time series using ant colony optimization, IEEE Trans. Cybern. 43 (2) (2013) 790–802.
[20] N. Kourentzes, S.F. Crone, Frequency independent automatic input variable selection for neural networks for forecasting, in: 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, IEEE, 2010, pp. 1–8.
[21] P. Cortez, Sensitivity analysis for time lag selection to forecast seasonal time series using neural networks and support vector machines, in: 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, IEEE, 2010, pp. 1–8.
[22] G.H. Ribeiro, P. de M Neto, G.D. Cavalcanti, R.I. Tsang, Lag selection for time series forecasting using particle swarm optimization, in: 2011 International Joint Conference on Neural Networks (IJCNN), San Jose, CA, IEEE, 2011, pp. 2437–2444.
[23] J.F. Lorenzato de Oliveira, T.B. Ludermir, A hybrid evolutionary system for parameter optimization and lag selection in time series forecasting, in: 2014 Brazilian Conference on Intelligent Systems (BRACIS), 2014, pp. 73–78.
[24] C.J.C. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Discov. 2 (2) (1998) 121–167.
[25] R. Hyndman, Time Series Data Library, 2010. URL ⟨http://data.is/TSDLdemo⟩.
[26] V. Vapnik, The Nature of Statistical Learning Theory, Springer, New York, 2000.
[27] R. Adhikari, A neural network based linear ensemble framework for time series forecasting, Neurocomputing 157 (0) (2015) 231–242.
[28] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, vol. 5, IEEE, Orlando, FL, 1997, pp. 4104–4108.
[29] S. Makridakis, S. Wheelwright, R. Hyndman, Forecasting : Methods and Applications, 3rd edition, Wiley, Hoboken, NJ, 1998.
[30] M. Rosenblatt, Gaussian and Non-Gaussian Linear Time Series and Random Fields, Springer Science& Business Media, New York, 2000.
[31] J. Fan, Q. Yao, Nonlinear Time Series: Nonparametric and Parametric Methods, Springer, New York, 2013.
[32] G.K. Grunwald, R.J. Hyndman, L. Tedesco, R.L. Tweedie, Theory and methods: non-Gaussian conditional linear AR (1) models, Australian and N. Z. J. Stat. 42 (4) (2000) 479–495.
[33] R.J. Hyndman, Y. Khandakar, Automatic time series forecasting: the forecast package for R, J. Stat. Softw. 27 (3) (2008) 1–22.
[34] T. Jansen, Analyzing Evolutionary Algorithms: The Computer Science Perspective, Springer Science and Business Media, Berlin Heidelberg, 2013.
[35] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, 2nd edition, John Wiley& Sons, New York, 2000.
[36] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011) 27:1–27:27.
[37] J. Kennedy, J.F. Kennedy, R.C. Eberhart, Swarm Intell., Morgan Kaufmann, San Francisco, CA, 2001.
[38] J. Armstrong, F. Collopy, Error measures for generalizing about forecasting methods: empirical comparisons, Int. J. Forecast. 8 (1) (1992) 69–80.
[39] M. Hollander, D.A. Wolfe, E. Chicken, Nonparametric statistical methods, Nonparametric Statistical Methods, John Wiley& Sons, Hoboken, NJ, 1999.
[40] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[41] M. Lngkvist, L. Karlsson, A. Loutfi, A review of unsupervised feature learning and deep learning for time-series modeling, Pattern Recognit. Lett. 42 (0) (2014) 11–24.

João F.L. de Oliveira received the B.Eng. degree in computer engineering from the University of Pernambuco, Brazil, in 2009, and received the M.Sc. degree in computer science from the Federal University of Pernambuco in 2012. He is currently a Professor at Universidade de Pernambuco, Brazil, and a Ph.D. student at Universidade Federal de Pernambuco, Brazil. His research interests include artificial neural networks, pattern recognition, hybrid intelligent systems, evolutionary computation, time series prediction.



Teresa B. Ludermir received the Ph.D. degree in Artificial Neural Networks in 1990 from Imperial College, University of London, UK. From 1991 to 1992, she was a lecturer at Kings College London. She joined the Centro de Informática at Universidade Federal de Pernambuco, Brazil, in September 1992, where she is currently a Professor and head of the Computational Intelligence Group. She has published over 200 articles in scientific journals and conferences, three books in Neural Networks and organized two of the Brazilian Symposium on Neural Networks. Her research interests include weightless Neural Networks, hybrid systems and applications of Neural Networks.