Applied Soft Computing Journal 174 (2025) 113063

Contents lists available at ScienceDirect

Applied Soft Computing

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

Data preprocessing techniques and neural networks for trended time series forecasting

Ana Lazcano [a,*], Miguel A. Jaramillo-Morán [b]

[a] Faculty of Law, Business and Government, Universidad Francisco de Vitoria, Madrid 28223, Spain
[b] University of Extremadura, Department of Electrical Engineering, Electronics and Automation, School of Industrial Engineering, Spain

HIGHLIGHTS

- Differentiating time series before forecasting enhances prediction accuracy in complex, nonlinear economic data.
- Techniques like Empirical Mode Decomposition and Wavelet Transform improve ANN performance in time series forecasting.
- LSTM and MLP networks achieve higher accuracy with differentiated time series, surpassing models without preprocessing.

ARTICLE INFO

Keywords:
Differentiation
Forecasting
Preprocessing
MLP
LSTM

ABSTRACT

Research on time series forecasting continues to attract significant attention, particularly in the use of Artificial Neural Networks (ANN) due to their ability to model nonlinear behaviors. However, forecasting economic time series with steep upward trends presents challenges, often leading to poorly fitting predictions. This study addresses the issue by applying differentiation as a preprocessing step. Three real-world time series exhibiting this behavior were analyzed and forecasted using two neural network models—Long Short-Term Memory (LSTM) and Multilayer Perceptron (MLP)—with and without preprocessing. The differentiated series were further processed using techniques such as Empirical Mode Decomposition (EMD) and trend-fluctuation decomposition via Moving Average of Wavelet Transform. The results demonstrate that differentiation significantly enhances forecasting accuracy across all tested models, reducing errors by up to 30 % compared to models without preprocessing. This approach effectively mitigates trend-related distortions, leading to more reliable predictions in complex economic time series.

1. Introduction

Research on time series forecasting has gained significant attention in recent years, particularly with the integration of Artificial Neural Networks (ANN). These models have consistently demonstrated superior predictive accuracy compared to traditional statistical techniques, such as Autoregressive Integrated Moving Average (ARIMA) models. The use of artificial neural networks (ANN) for time series forecasting has been extensively studied in the literature, highlighting their ability to capture nonlinear relationships without requiring prior assumptions about data structure. Zhang and Qi [1] demonstrated in their study that neural networks can improve the forecasting of time series with seasonal and trend components, comparing them to traditional models such as ARIMA and showcasing their advantages in terms of accuracy. On the other hand, Lago et al. [2] conducted a comparative analysis between different deep learning methodologies and traditional models for forecasting spot electricity prices, demonstrating the superiority of the former in capturing complex market dynamics. Viviani et al. [3] focused on energy market forecasting, exploring the role of inferential statistics and machine learning in the context of the German electricity market, providing insight into how different approaches can complement each other to enhance predictive accuracy. Finally, Ubal et al. [4] investigated the ability of recurrent neural networks to model long-term dependencies in time series, showing how these architectures can improve forecasting in contexts where temporal relationships are essential.

However, the complexity of certain types of time series, such as economic time series, can make it difficult to obtain accurate predictions due to their highly volatile nature and susceptibility to multiple

* Corresponding author.
E-mail address: ana.lazcano@ufv.es (A. Lazcano).

https://doi.org/10.1016/j.asoc.2025.113063
Received 18 September 2024; Received in revised form 14 March 2025; Accepted 17 March 2025
Available online 22 March 2025

exogenous factors. These series are usually influenced by macroeconomic events, market cycles, government policies and unexpected phenomena, such as financial crises or regulatory changes, which introduce variability that is difficult to model. In particular, series with marked upward trends present additional challenges, as traditional models may suffer from overfitting or an inability to correctly capture the underlying dynamics, especially in periods of prolonged expansion. Furthermore, the existence of non-stationarity and the possible presence of structural changes in the data require advanced techniques, such as differentiating transformations or multiscale decompositions, to improve the predictive capacity of the model. The correct selection of hyperparameters, the use of hybrid approaches that combine neural networks with statistical techniques and the integration of contextual information can be key to improving the accuracy in the prediction of this type of series.

Building upon these findings, this study aims to further explore the effectiveness of neural networks in time series forecasting by conducting a series of controlled experiments. Specifically, we will evaluate the performance of different neural architectures, including Long Short-Term Memory (LSTM) networks and Transformer-based models, across multiple datasets with varying characteristics. The experiments will be designed to assess the impact of preprocessing techniques, such as Empirical Mode Decomposition (EMD) and differentiation, on the predictive accuracy of these models. Additionally, we will investigate the role of hyperparameter tuning, optimization strategies, and training data partitioning in improving forecasting reliability.

This work presents several contributions that address the challenge of forecasting economic time series with a steeply rising trend. These contributions are as follows:

1. Proposal of Differentiation as a Preprocessing Step – This work demonstrates that differentiation improves the forecasting accuracy of ANN models for steeply rising economic time series.
2. Comparison of Preprocessing Techniques – A systematic evaluation of preprocessing methods, including EMD and trend-fluctuation decomposition, is conducted on differentiated time series.
3. Empirical Validation with Real-World Financial Data – The methodology is applied to real economic time series, confirming that differentiation consistently reduces forecasting errors.
4. Performance Improvement in Neural Forecasting Models – The study shows that ANN models, specifically LSTM and MLP, achieve significantly lower prediction errors when differentiation is applied.
5. Bridging Economic Theory and Forecasting Techniques – The approach isolates and processes the trend component separately, aligning forecasting methodologies with economic principles for better interpretability.

The remainder of the paper is organized as follows. Section 2 shows a relevant bibliographic review on the topic to be studied, Section 3 provides a detailed description of the forecasting models used and their architecture, as well as the preprocessing techniques used, and the differentiation technique proposed to deal with the rising trend or the series. Section 4 provides a detailed description of the data used, the error metrics studied and the parameters of the neural networks. Section 5 focuses on the experiments carried out and the comparison of the results. Finally, Sections 6 and 7 describe the main conclusions and future lines of work.

2. Previous works

With the advance in research, a variety of ANN architectures have been explored for time series forecasting, ranging from classical models like the Multilayer Perceptron (MLP) to more advanced Deep Learning (DL) structures and Transformer models. The latter have achieved remarkable success in areas closely related to human intelligence, such as text recognition, translation, and generative artificial intelligence. Their ability to capture intricate temporal dependencies suggests their potential for time series prediction as well. However, despite their success in other domains, their application to economic and financial time series, which are characterized by high volatility and uncertainty, remains a challenge. Numerous studies have applied ANN models to economic and energy-related time series, given their significant practical implications. These applications include electricity market transactions [5], natural gas demand [6], stock market variables [7,8], electricity prices [9], CO_2 emission allowance prices [10,11] or electricity load forecasting [12,13].

However, economic time series present certain challenges due to their susceptibility to external influences, such as human decision making that is difficult to model. Although ANN models demonstrate high efficiency in handling these complexities, they still suffer from performance issues as it is not really reliable in real-world applications.

But it is common not to take into account one of the critical aspects of time series, which is the tendency to exhibit strong upward trends in economic and financial series, especially when there is a pronounced increase. [1,14], highlighting a problem little addressed in the literature. When the trend is moderate, many forecasting models can achieve reasonable accuracy; however, when the trend intensifies, traditional and ANN-based models often struggle to provide reliable forecasts. To address this, researchers have explored various preprocessing strategies aimed at transforming the original time series into a more predictable form. These approaches include statistical data treatment to remove irrelevant or erroneous [15]. Therefore, several tools should be applied to the specific time series to be processed in order to find out which is the best preprocessing strategy for that specific time series. A first step in this process is to perform a statistical treatment on the data to remove those that are irrelevant or erroneous, thus eliminating possible sources of confusion [16].

In addition to applying statistical tools to enhance the quality of the data for forecasting, aiming to obtain a more robust and predictable time series [17,18], another approach involves decomposing the original time series into a collection of sub-series, each exhibiting more consistent behavior, making them easier to forecast. This approach is based on the idea that many economic variables are influenced by social and environmental factors that exhibit periodic patterns. Consequently, the goal should be to decompose these time series into sub-series that collectively capture this periodic behavior. Each sub-series should therefore maintain a certain periodicity, related to specific frequencies inherent in the overall behavior of the time series. By doing so, these sub-series are expected to be more predictable, resulting in more accurate forecasts. One such technique is Empirical Mode Decomposition (EMD), which has been employed in various studies due to its ability to improve the performance of forecasting models. It has been successfully combined with neural models such as MLP [19], LSTM [20,21], or Transformers [22]. These studies consistently demonstrated that models using preprocessing techniques outperformed those without it.

On the other hand, complex models that combine several forecasting tools, with or without preprocessing, can be found in literature with the goal of providing accurate and reliable predictions of complex time series. In [23] an Ensemble EMD with a Gated Recurrent Unit (a simplification of the LSTM) and a Graph Convolutional Network (a Deep Learning model with a graph type organization) was defined to process time series of energy and traffic flows. The LSTM has been used together with a Convolutional Neural Network (CNN, another class of Deep Learning network) [24] to forecast short-term electric load or power generation in a photovoltaic plant [25]. In [26] it was combined with a MLP and XGBoost (an artificial intelligence classification algorithm) to predict consumption of a household user. In [20] a Genetic Algorithm was used to optimize an LSTM. In [27] an ARIMA-CNN-LSTM model was used to forecast carbon prices.

Nevertheless, there are some economic time series that are very difficult to forecast: those with a steeply rising trend. This behavior represents a difficult challenge for ANN because they are trained with a specific data set, but, once trained, if they are applied to forecast data that are much higher than those used for training, which will easily happen with series with a step rising trend, they fail to provide accurate predictions because they were not trained to process such data. This is a challenge for all forecasting tools, Lin et al. [28] introduced hybrid neural networks to learn trends in time series, improving prediction capabilities. Qi and Zhang [29] explored neural networks for modeling and forecasting trend time series, demonstrating their power in capturing underlying patterns, as they require a specific treatment of the data to allow reliable predictions despite that rising trend.

Therefore, the goal should be to isolate the rising trend of the time series so that it can be properly processed and easily predicted by the forecasting tool. Thus, the time series should be decomposed into its trend, which maintains its rising behavior, and its fluctuations around that trend. Then, the trend sub-series could be properly treated to overcome the challenge posed by its rising behavior.

There are not many works in scientific literature that specifically address this problem, so this paper proposes a tool to transform time series with such a steeply rising trend into another that might be easier to predict. This tool is differentiation, since the derived series of one with a rising trend can be expected to have a softer evolution with more or less bounded values. In this work it will be proved that the differentiation, not only of the trend subseries but also of the original one, will allow the forecasting tools, in this case ANN models, to provide accurate and reliable predictions of time series suffering from a steeply rising trend. It will be proven that differentiation improves the forecasting accuracy of the neural network both with and without preprocessing. A very simple preprocessing procedure will be able to significantly improve the accuracy of time series with a steeply rising trend.

3. Methodology

Predictions will be made using two ANN models that are widely used for time series forecasting: Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM). They will be used both with and without preprocessing, allowing a comparison of the performance of the different models obtained in this way.

The choice of MLP and LSTM neural networks is due to their great stability and demonstrated efficiency in time series prediction models, especially those that are based on previously treated data sets [30,31].

Nevertheless, there are many time series whose complex behavior does not allow the forecasting tools alone to provide predictions as accurate as desired. Many works have shown that the accuracy of the forecasting tools can be improved if the data are preprocessed before they are used, i.e., the original dataset is transformed into another one that is expected to be easier to forecast. This process is particularly useful when the dataset to be processed has a more or less periodic behavior, since the preprocessing can be designed to decompose it into a set of different series that can be linked to specific frequencies that could be more easily predicted.

In this work, data preprocessing will be carried out by three methods. Two of them will decompose the original time series into its trend and fluctuations using two different techniques: one with the moving average and the other with wavelet filtering. The third will decompose the time series into several subsets of data more or less related to specific frequencies: the Empirical Mode Decomposition.

Despite the fact that the inclusion of preprocessing in forecasting models has greatly improved the accuracy of neural networks as forecasting models. There are some time series that are very complex to predict, and these models usually fail to provide reliable predictions: those with a clear rising trend. This is because neural networks are trained with a particular dataset whose time evolution they learn, but when data with increasingly higher values are fed into the network, they tend to saturate the neurons' responses due to the fact that they all provide bounded outputs. To deal with this problem, in this work, a new step is added to the preprocessing process: differentiation. This process seeks to eliminate the rising trend of the time series by replacing it with its derivative, since it will provide a new dataset without this rising trend.

3.1. MultiLayer perceptron

MLPs can be considered as a classic neural model for time series forecasting, as they have been used to predict many types of time series [7–11,32]. This is because they have been proven to be universal approximators [33], i.e., they can approximate any continuous function with one hidden layer, provided it has enough neurons. They have a very simple structure that is easy to program [34,35], yet they continue to be the subject of recent research [36] because they have been able to provide reliable predictions.

The network's processing units, commonly referred to as neurons, are structured into multiple layers, which is why these architectures are called "multilayer" networks. The initial layer corresponds directly to the input data provided to the neural network, while the final layer is designed to produce an output that reflects the network's response to the processed input. Between these two, there are one or more hidden layers. Each neuron within these layers processes the information it receives from all the neurons in the preceding layer:

$$x_j^l = f\left(\sum_i w_{ji}^l x_i^{l-1} + b_j\right) \quad (1)$$

In this formulation, x_j^l denotes the output of neuron j in layer l. It is computed based on the outputs of all neurons in the preceding layer, x_i^{l-1}, each scaled by their respective weights w_{ji}^l, which represent the connections between neuron J and the neurons in the previous layer. For this reason, these layers are often referred to as dense layers [37]. The output x_j^l is determined by applying an activation function *f()* to the weighted sum of these inputs, with the addition of a bias term b_j. Commonly used activation functions in the hidden layers include the hyperbolic tangent and sigmoid functions, while the output layer typically employs a linear activation function.

The strength of multilayer Perceptrons (MLPs), as well as other neural architectures, lies in their capacity to learn patterns from data. Consequently, these models require a training phase before they can be deployed effectively. To accomplish this, the available dataset is split into a training set for learning and a validation set to assess the performance of the trained network. This practice is standard across all neural network models.

During training, input data is organized into pairs of inputs (patterns) and their corresponding desired outputs. The network processes the inputs and generates outputs, which are then compared against the target values to calculate an error metric. This error is propagated backward through the network to adjust the weights of each neuron in a manner that reduces the error. This process is executed using the well-known backpropagation algorithm [38].

3.2. LSTM

Long Short-Term Memory (LSTM) networks are a highly sophisticated neural architecture specifically designed to handle sequential data, such as text, speech, or translation tasks [39]. These networks incorporate an internal memory mechanism and feedback loops within neurons of the same layer. Due to their intricate structure, the processing units in LSTM networks are referred to as "cells" rather than neurons.

The structure of an LSTM cell is illustrated in Fig. 1, where the temporal dynamics of inputs ($\mathbf{x_t}$), outputs ($\mathbf{y_t}$), feedback connections ($\mathbf{y_{t-1}}$), and memory states (c_{t-1} and c_t) are depicted. Each cell includes three gates—input, forget, and output gates—that regulate the flow of data within the cell over time. These gates determine what portion of the information they govern should be used to update the cell's internal

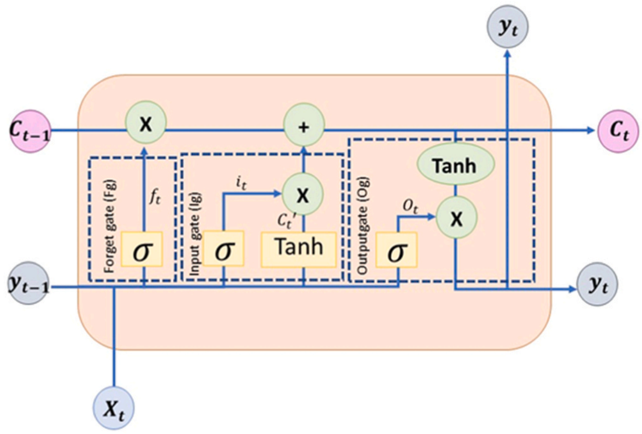


Fig. 1. LSTM neural network.

state and contribute to its output.

The forget gate decides whether the stored information, c_{t-1}, should be retained or removed (forgotten). To do this, it performs the weighted sum of the new inputs, $\boldsymbol{x}_t$, and the outputs of all neurons in the layer in the previous time step, $\boldsymbol{y}_{t-1}$, processed by a sigmoid function that provides a response between 0 ("memory" erased) and 1 ("memory" retained):

$$f_t = \sigma(\boldsymbol{W}^f[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^f) \quad (2)$$

In this expression $[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t]$ represents the input vector (a column vector) to the cell, which consists of the new inputs at time *t*, $\boldsymbol{x}_t$, combined with the outputs of all neurons in the same layer at time *t-1* (feedbacks), $\boldsymbol{y}_{t-1}$. The term $\boldsymbol{W}^f$ denotes the weight vector associated with the forget gate, which includes weights for both the new inputs and feedback signals, while b_f represents the corresponding bias. It is important to highlight that, in the neural network literature, weights are typically represented as matrices when describing a layer as a whole (with each row corresponding to a specific neuron). However, in the formula above, which describes the behavior of an individual neuron, the weight matrix reduces to a row vector. This row vector is then multiplied (via a dot or scalar product) by the column vector comprising the new inputs and feedback signals.

The input gate determines whether the internal state of the cell (its "memory") should be updated with new information. To make this decision, the weighted sum of the new inputs and feedback signals is passed through a sigmoid activation function, which regulates the proportion of the new information to incorporate into the cell's "memory":

$$i_t = \sigma(\boldsymbol{W}^i[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^i) \quad (3)$$

where, $\boldsymbol{W}^i$ and b^i are the corresponding weight vector and bias.

The information to be incorporated into the cell's "memory" is obtained by applying a hyperbolic tangent function (which produces values between −1 and +1) to the weighted sum of the new inputs to the cell and the outputs from other neurons in the same layer. This calculation includes a weight vector $\boldsymbol{W}^c$ and a bias term b^c:

$$c_t' = \tanh(\boldsymbol{W}^c[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^c) \quad (4)$$

The updated internal state is calculated as the sum of the portion of its previous value retained by the forget gate and the contribution from the new information regulated by the input gate. This can be expressed as:

$$c_t = f_t c_{t-1} + i_t c_t' \quad (5)$$

The output gate determines what fraction of the updated internal state is emitted as the cell's output. This value, ranging from 0 to 1, is calculated using a sigmoid activation function applied to the weighted sum (where $\boldsymbol{W}^o$ is the weight vector) of the new input data and the previous outputs from neurons in the same layer, along with a bias term (b^o). This process is analogous to the computations performed by the other gates (f_t and i_t):

$$o_t = \sigma(\boldsymbol{W}^o[\boldsymbol{y}_{t-1}, \boldsymbol{x}_t] + b^o) \quad (6)$$

The updated cell output is calculated as the portion of the hyperbolic tangent of the new internal state that is permitted to pass through by the output gate:

$$y_t = o_t \tanh(c_t) \quad (7)$$

LSTM networks are trained using a variation of the classic Backpropagation algorithm, specifically adapted to the recurrent architecture of this model. The training process incorporates two key modifications: Truncated Backpropagation Through Time (BPTT), which is employed to adjust the weights of the output units and output gates, and Real-Time Recurrent Learning (RTRL), which is used to update the weights of the cell inputs, input gates, and forget gates [40].

3.3. Data preprocessing

3.3.1. EMD

The Empirical Mode Decomposition (EMD) breaks down the original time series into a set of Intrinsic Mode Functions (IMFs) and a residue, based on the inherent characteristics of the time series, such as its maxima, minima, and zero crossings [41]. IMFs are directly associated with specific periodic components of the original time series. To be valid, they must satisfy the following conditions:

- The number of extrema and the number of zero crossings must differ by no more than one.
- The mean value of the envelope defined by the local maxima and the envelope defined by the local minima must be zero at each point.

The IMFs are derived through a heuristic procedure known as sifting, which decomposes the original time series ***x(t)***. This process can be summarized in the following steps [42]:

1. Maxima and minima are identified in the time series. From them, two envelopes are defined by interpolation: an upper one ***u(t)*** with the maxima and a lower one ***l(t)*** with the minima.
2. The mean value of the upper and lower envelopes is calculated:

$$\boldsymbol{m}(\boldsymbol{t}) = \frac{\boldsymbol{u}(\boldsymbol{t}) + \boldsymbol{l}(\boldsymbol{t})}{2} \quad (8)$$

3. The difference between *x(t)* and m(t) is calculated to obtain a detailed component:

$$\boldsymbol{h}(\boldsymbol{t}) = \boldsymbol{x}(\boldsymbol{t}) - \boldsymbol{m}(\boldsymbol{t}) \quad (9)$$

4. If h(t) fulfills the above two conditions, it is assumed to be the first IMF ($\boldsymbol{h}(\boldsymbol{t}) = \boldsymbol{c_1}(\boldsymbol{t})$) and the algorithm jumps to the next step. Otherwise, h(t) is considered as a new time series and is processed according to steps 1–3 until an IMF is obtained.
5. ***c₁(t)*** is subtracted from **x(t)** to obtain a new series without the frequencies associated with ***c₁(t)***:

$$\mathbf{x_n}(\mathbf{t}) = \boldsymbol{x}(\boldsymbol{t}) - \mathbf{c_1}(\mathbf{t}) \quad (10)$$

6. Steps 1–5 are repeated until no new IMF is obtained.

When the process stops, the last IMF is subtracted from the time series that provided it to obtain a residue ***h(t)=r(t)***.

The original time series is decomposed as:

$$\boldsymbol{x}(\boldsymbol{t}) = \sum_{i} \boldsymbol{c}_i(\boldsymbol{t}) + \boldsymbol{r}(\boldsymbol{t}) \tag{11}$$

The algorithm has two stop conditions. The first is for steps 1–3 and is reached when an IMF candidate meets the two conditions above or when a certain value for the IMF candidate variance is reached. The second is for the entire process and is applied when the residue is a constant, has a constant slope or contains only one extreme.

3.3.2. Trend decomposition

The decomposition into a trend and fluctuations has been studied in the literature [43,44], with the aim of using different forecast structures that can be better adapted to the behavior of the resulting datasets These studies tend to show different opinions about this type of decomposition and the results that will be obtained when applying them. This decomposition allows us to isolate the trend of the time series [45], so that it can be processed independently from the fluctuations. This is particularly interesting when the trend has a steeply rising behavior, as in the case of the series used in this work. One of the most widely used algorithms for this purpose is the moving average because of its simplicity. It must be adapted to the structure of time series, since only the data preceding the one to be transformed can be used to carry out the process. Thus, it has the form:

$$\boldsymbol{T}(\boldsymbol{t}) = \frac{1}{\boldsymbol{N}+1} \sum_{i=0}^{N} \boldsymbol{x}(\boldsymbol{t} - \boldsymbol{i}) \tag{12}$$

Where N is the number of values preceding the one to be transformed. Since the N data preceding this one are used to do obtain its new value, the first N values of the time series are removed, and the resulting trend series will have N data less than the original one. The fluctuation is obtained by subtracting this trend from the original time:

$$\boldsymbol{F}(\boldsymbol{t}) = \boldsymbol{x}(\boldsymbol{t}) - \boldsymbol{T}(\boldsymbol{t}) \tag{13}$$

In this expression, the first N data of the original series must be discarded to properly perform the subtraction. The resulting fluctuation series will also have N less data.

3.3.3. Wavelet filtering

The wavelet transform performs a local analysis in time and frequency [46]. A multiscale transformation is performed by scaling and translating the wavelet function along the original dataset. It performs a decomposition of a signal similar to that of the Fourier transform, with the difference that the latter only performs a translation of the basic (sin and cos) functions. The wavelet transform, on the other hand, decomposes the signal *x(t)* into a sum of scaled and translated basis functions called wavelets:

$$x(t) = \sum_{s} \sum_{\tau} w_{s\tau} \psi_{s\tau}(t) \tag{14}$$

In this expression $\psi_{s\tau}(t)$represents the scaled (s) and translated (τ) version of the mother wavelet ψ:

$$\psi_{s\tau}(t) = 2^{-s/2} \psi(2^{-s/2} t - \tau) \tag{15}$$

The mother function ψ can have different forms (Daubechies, Haar, Coiflet, Meyer, Symmlet…) that define different transformations. They all have values different from zero in a reduced time interval, the width of which can be modified (scaling) and placed (translating) at different points of the whole set of time values.

The coefficients $w_{s\tau}$ are obtained from the expression:

$$w_{s\tau} = \frac{1}{\sqrt{s}} \int_{-\infty}^{\infty} \psi^* \left(\frac{t-\tau}{s} \right) x(t) dt \tag{16}$$

In this equation *x(t)* is the original signal in the time domain. The function ψ^* represents the complex conjugate of the scaled and translated original wavelet as shown in Eq. (15).

The wavelet transform of a signal provides a representation in both the time and frequency domains. This transformation allows for the extraction of relevant information from the signal by properly processing the information provided by the coefficients $w_{s\tau}$. Therefore, the original time series can be processed in the transformed space, allowing the wavelet transform to behave like a filter by properly modifying the values of these coefficients. Unlike traditional filters, which operate directly on the original signal without any kind of transformation, the wavelet transform decomposes the signal by means of scaled and translated versions of the mother wavelet, providing the possibility of a richer and more precise processing of this signal in the transformed space. This option allows wavelets to be used as filters [47]. For the problem at hand, this filtering process will perform a sort of low-pass filter to remove the high frequencies and retain only the low ones. Thus, the resulting series will consist only of the low frequencies, i.e., those that describe the trend of the original series. In this way, the original time series can be decomposed into trend (low-frequencies) and fluctuations (high-frequencies), as it was done with the moving average. To do this, a three-steps process can be used to remove the higher frequencies:

1. The original signal *x(t)* is transformed into the wavelet domain providing a set of wavelet coefficients $w_{s\tau}$.
2. A thresholding process is performed to remove those coefficients that are below a certain threshold.
3. The original signal is reconstructed using only the remaining coefficients to obtain an approximation to the original signal in which the noise has been removed.

3.4. Time series derivatives

Differentiation of a time series allows us to measure the instantaneous rate of change of a signal *x(t)* as a function of time t. This derivative describes how the signal changes at each point with respect to time. By calculating this derivative, a new time series is obtained that describes how the original signal varies at each point in time:

$$d_x(t) = \frac{d}{dt} x(t) \tag{17}$$

This formula represents the derivative of the signal *x(t)* with respect to time. This new time series should have a stationary behavior without the rising trend that the original one had, so that it has a bounded behavior that is easier to forecast. However, since the time series is a discrete sequence of data, the concept of differentiation cannot be applied directly. Instead, it can be approximated by a finite difference as the increment of ***x(t)*** divided by the corresponding time increment. Assuming a time increment of 1, it has the form:

$$\boldsymbol{d}_x(t) = \frac{\Delta \boldsymbol{x}(t)}{\Delta t} = \boldsymbol{x}(t) - \boldsymbol{x}(t-1) \tag{18}$$

It is worth noting that the new derived time series loses its first element. From expression (18) it can be concluded that once the derived series is forecast, the corresponding non-derived forecast will be reconstructed by summing the forecast derivative to the data preceding it.

4. Experimentation

This research seeks to achieve more accurate predictions in time series with steeply rising trends, as current neural network models have serious problems predicting values significantly higher than those with which they were trained. Despite the extensive existing literature on predicting economic time series, few address the prediction difficulties when the training and test sets differ greatly in their maximum values [48].

To test the validity of the solutions, the preprocessing tool proposed to overcome this problem, differentiation, will be tested with different forecasting structures, and the results obtained will be compared with those obtained with the same models without differentiation.

Three time series with a clear rising trend will be forecast with the same forecasting tools. First, the two selected neural models, together with the preprocessing structures, will predict these three time series (for the sake of comparison, the neural models will also predict them without preprocessing) to show that they are not able to provide accurate predictions. Then the same models will be applied to the same time series but now with the differentiation step included. It will be applied directly to the time series when using the neural model alone and the neural model with EMD. However, it will be applied to the trend series when Moving Average and Wavelet Transform are applied, since the rising trend of the series has been isolated in the trend series obtained with these two tools.

All the forecasting models applied to the time series studied will use past data to predict a single future value, i.e., they will process past prices to predict a single future price. The number of input data used in each model must be determined with a trial-and-error process. They will be those immediately preceding the one to be predicted.

Several structures of each neural network have been trained to find those that give the most accurate results, which are the ones presented in this paper. The MLP has a single hidden layer (with the hyperbolic tangent as activation function) and an output layer (with a linear activation function). The hidden layer has 3 neurons, while the output layer has only one neuron because only one future datum is predicted. Three past data prior to the one to be predicted are used as inputs to the network. The LSTM consists of an LSTM followed by a fully connected network and a regression network as output. The LSTM layer has 5 neurons, while the other two layers have only one neuron as only one future data is predicted. The network receives one only historical data as input, as the "internal memory" stores the information regarding past data.

All the simulations were performed using the software package MATLAB R2024a on a personal computer endowed with an Intel i9 processor and an RTX4070 graphics card.

4.1. Dataset

This section describes the data used in the evaluation process of the different forecasting models tested. Three time series with a clear rising trend will be forecasted with the neural networks and the preprocessing techniques presented. All of these structures will be applied to both the derived and non-derived time series.

As mentioned above, when using neural networks for time series forecasting, the data set must be divided into two subsets: one for training and another for validation. In this work, the first 75 % of the data is used for training and the last 25 % for validation.

The first time series studied corresponds to the Apple stock index with daily data from 7 May 2004–8 May 2024 with a total of 5218 observations. It is shown in Fig. 2. In Fig. 3 it is decomposed into its trend, obtained with the Moving Average, together with the corresponding standard deviation. It can be seen the clear rising trend of the series, especially in the most recent data. This fact highlights the difficulty of the forecasting models to provide accurate predictions, since these last data are the ones selected to validate the model, while the first ones are used for training. Therefore, the forecasting models will be trained with data with a moderate rising trend, while the data used for validation show a steeper behavior with values much higher than those used for training. It is clear that the forecasting models will have significant difficulties in predicting these latter data.

The second time series corresponds to the Nvidia stock index with daily data for the last 20 years from May 6, 2004 to May 8, 2024 (5219 data). It is shown in Fig. 4. Like the previous one, this series reflects a steep upward trend, especially in the last years, with a behavior even steeper than that of Apple. The difference between the first and the last dataset behavior is more extreme than for the Apple series, with very low values for the fist and very high values for the last. These differences can be seen more clearly in Fig. 5. Both time series were obtained from the Refinitv Eikon platform.

The third time series corresponds to the European Union Allowance Prices of CO_2 prices obtained from the European Energy Exchange (EEX), Leipzig, Germany. The data were collected from April 22, 2005 to December 28, 2022, with a total of 4555 observations. It is shown in Fig. 6, while the corresponding trend series and standard deviation are

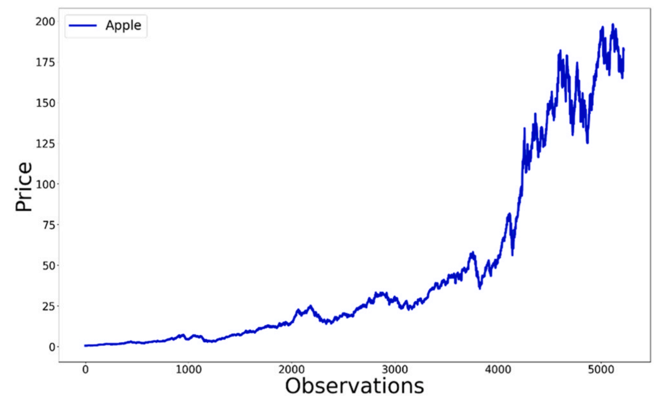


Fig. 2. Apple prices.

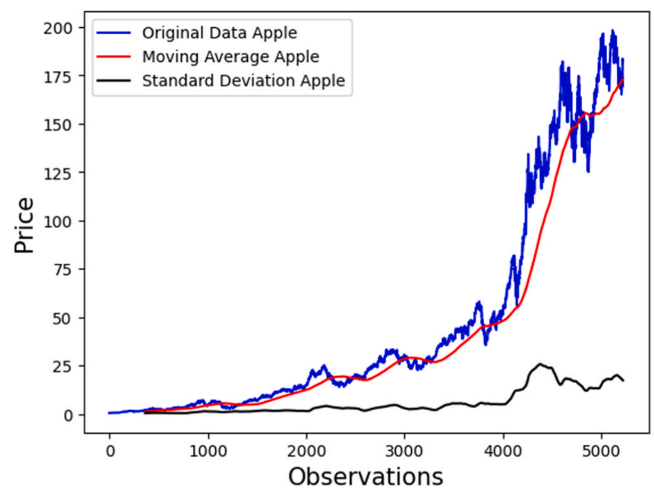


Fig. 3. Apple trend and standard deviation of data.

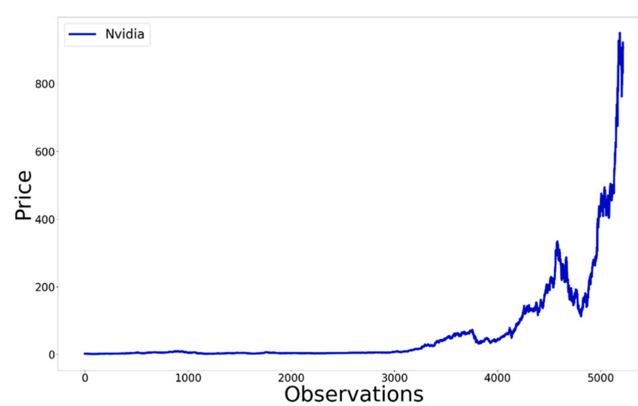


Fig. 4. Nvidia Time Serie.

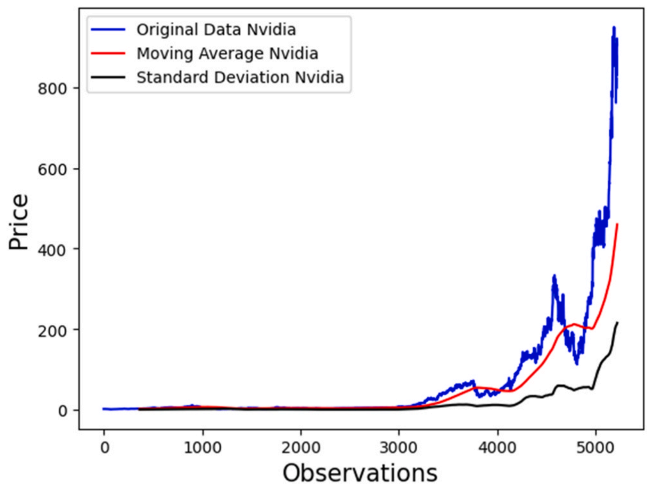


Fig. 5. Nvidia trend and standard deviation of data.

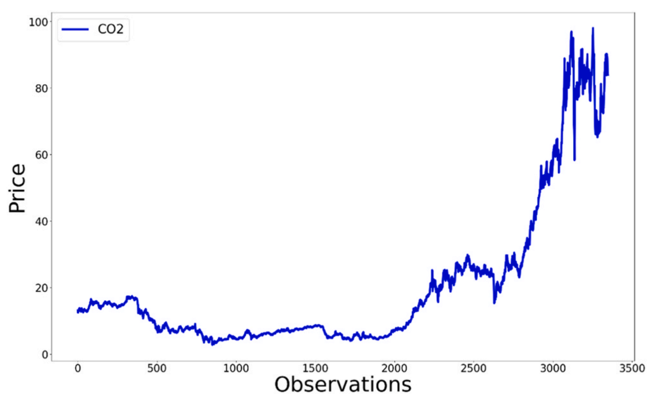


Fig. 6. CO2 Time Serie.

shown in Fig. 7. This series has a slightly different trend, as the initial and final values are not as different as in the previous two series. In addition, the initial values, those used for training, have large variations, suggesting that the prediction models may be able to properly handle the higher values with a steeply increasing trend of the final values (those used for validation).

For a more detailed and comprehensive comparison of the performance of the preprocessing and DL techniques, the time series of the

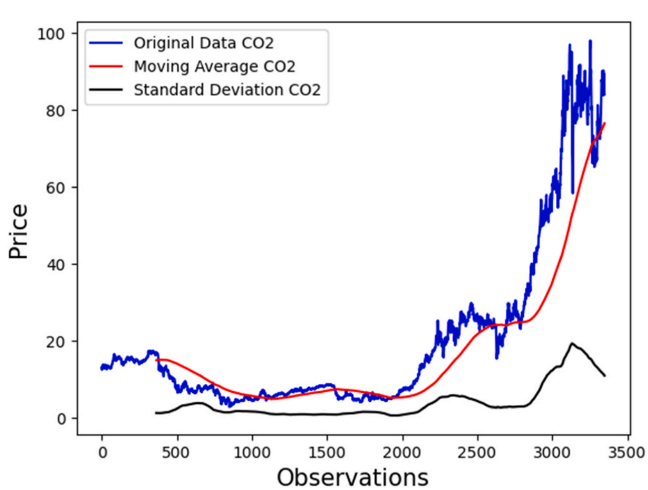


Fig. 7. CO_2 trend and standard deviation of data.

S&P 500 index is also analyzed. The data covers a period from February 23, 2004, to February 23, 2024, comprising a total of 5220 observations, which provides a robust dataset for evaluation. This particular time series is of special interest because it exhibits a trend that is not strongly increasing, making it a valuable case for testing the adaptability and accuracy of the methods under study. The subtle nature of the trend challenges the models to detect and predict patterns that are less pronounced compared to more dynamic or volatile datasets. Therefore, the results obtained from this analysis can shed light on the strengths and limitations of the techniques when applied to similar financial time series with relatively mild trends. (Figs. 8 and 9).

Another time series with a trend that is not very pronounced but displays significant variations is the one shown in Figs. 10 and 11, corresponding to the West Texas Intermediate (WTI) oil price, which serves as a key reference index for global oil prices. The data for this series has been collected from Refinitiv Eikon, spanning a substantial period from January 10, 1983, to June 15, 2022. This time frame includes a total of 10,288 observations, providing an extensive dataset for experimentation and analysis. The large sample size ensures the ability to capture various market dynamics, including both short-term fluctuations and long-term patterns. Moreover, the inclusion of this series is particularly valuable as it allows the evaluation of techniques on data characterized by high variability and complex movements, making it a challenging but insightful case for the study.

4.2. Error metrics

The error metrics will allow us to efficiently measure the precision provided by the models. For this study, the metrics most commonly used in predicting time series with neural networks are selected [49].

Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(x(t) - \hat{x}(t))} \tag{19}$$

Mean Squared Error (MSE):

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(x(t) - \hat{x}(t))^2 \tag{20}$$

Mean Absolute Percentage Error (MAPE):

$$MAPE = \frac{1}{N}\sum_{t=1}^{N}\frac{|x(t) - \hat{x}(t)|}{|x(t)|}x100 \tag{21}$$

R-squared (R^2):

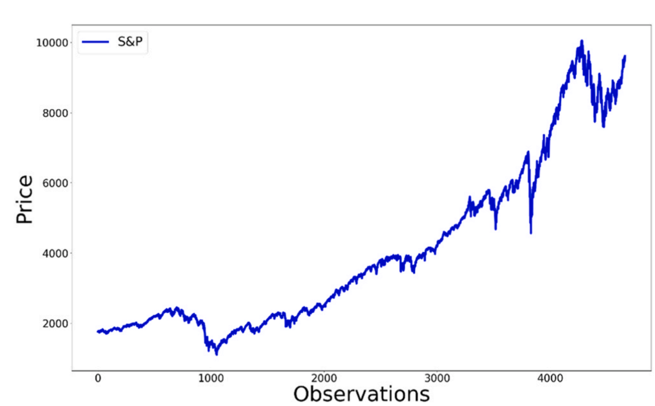


Fig. 8. S&P Time Serie.

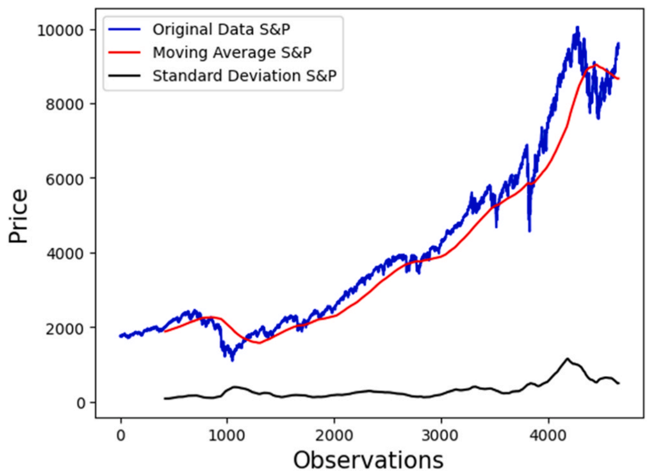


Fig. 9. S&P trend and standard deviation of data.

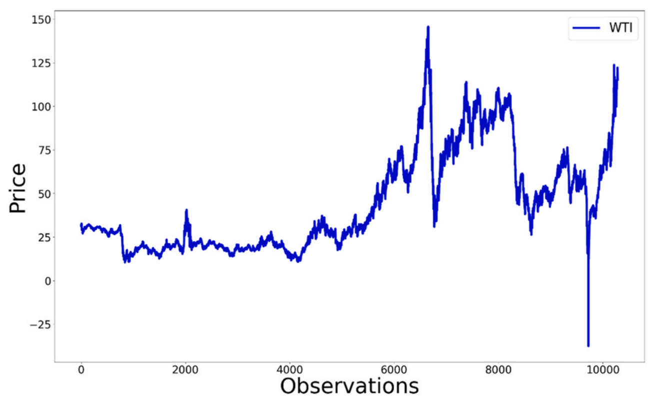


Fig. 10. WTI Time Serie.

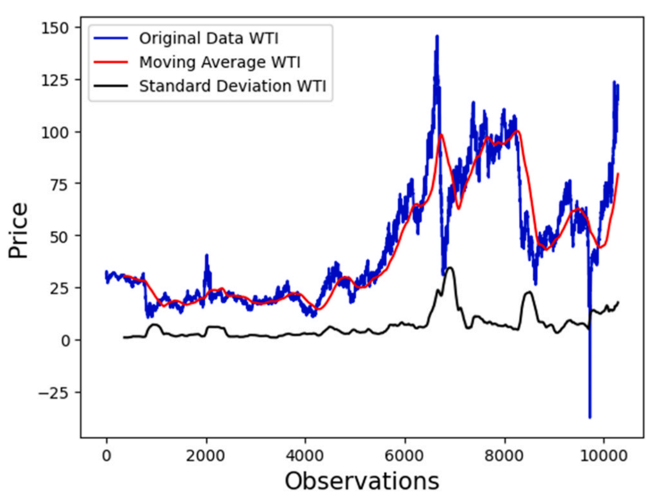


Fig. 11. WTI trend and standard deviation of data.

$$R^2 = 1 - \frac{\sum_{t=1}^{N} \left(x(t) - \widehat{x}(t)\right)^2}{\sum_{t=1}^{N} \left(x(t) - \overline{x}(t)\right)^2} \tag{22}$$

In these equations, *x(t)* is the t-th element of the time series, while $\widehat{x}(t)$ represents the predicted value, $\overline{x}(t)$ the mean value of the time series and N the number of elements it has.

The error in the predictions is measured using the MAPE and RMSE metrics, which allow us to know the accuracy of the forecasting model, the ideal being the lowest possible value since it represents the least error in the predictions.

Similarly, MSE and R^2 allow us to know the performance of the models by examining the variation of the dependent variable based on the prediction of the independent variable. The closer the value is to 1, the better performance the R^2 metric will reflect.

R^2 is a widely used metric to evaluate the fitting performance of nonlinear models and can be complemented by other metrics such as MAPE, MSE, and RMSE. It allows an intuitive and direct comparison with previous studies, thus facilitating the interpretation of results for a wider audience [50]. Although R^2 is traditionally used with linear models, its application to nonlinear models can provide valuable information about the proportion of variability explained by the model. Widely recognized and understood as a very valuable metric, R^2 enhances the understandability of the results. Its use in conjunction with MAPE, MSE, and RMSE ensures a robust and comprehensive assessment of model performance and addresses the potential limitations of R^2 by providing multiple perspectives on model accuracy and error [51,52], being sometimes described in the literature as a more informative metric than the rest in regression problems by quantifying the extent to which the dependent variable is determined by the independent variables. It can take a negative value [53] in cases of a poor fit to the problem posed [54].

The choice to use different metrics is because it is advisable to use multiple performance metrics to ensure the validity of Machine Learning models [55], as this can help overcome some of the limitations of individual metrics, being used in the literature for different types of time series, including those with a marked trend [28,56,57].

4.3. Parameters

When using neural networks, it is necessary to define some parameters (hyperparameters) that control the way the learning process is carried out. It is important to set them correctly in order to obtain an efficient and accurate neural network from the training process. The most important ones are:

- Learning Rate: It defines the rate of variation of the weights in the learning process. A small value will define an accurate but slow algorithm while a higher value will define a faster but less accurate process. A balanced value should be set.

- Batch Size: It specifies the number of samples to analyze before updating the network weights.

- Epoch: This is the number of times the entire dataset will be processed by the model in the training phase.

- Optimization Algorithm: It defines the learning algorithm used to train the neural model. The Adam algorithm is one that is commonly used to train LSTMs. It combines the advantages of RMSProp, which is similar to gradient descent, with momentum [58]. It has been used in this work. For MLP the classical training algorithm is Levenberg-Marquardt, which is a combination a combination of the gradient descent and the Gauss-Newton. It has been chosen for this work [59].

The choice of the values of these hyperparameters will be subject to computational limitations, but with the aim of avoiding both overfitting and underfitting, two effects that will endanger the predictive capacity of the neural model.

The methods for choosing hyperparameters have been widely studied in the literature. Machine learning models can be estimated through iterative trial and error [60], a methodology used in detailed experiments given the characteristics of the data since methods such as grid search and random search, which seek to optimize the process automatically, lose its efficiency quickly increases by increasing the number of hyperparameters [61], making the training of the models costly both in time and in computational cost.

Goodfellow [62] gave some recommendations on the most appropriate choice of parameters for making predictions with neural networks, and provided a comparison of different options. The choice will be subject to computational limitations, but with the goal of avoiding both overfitting and underfitting, two effects that will jeopardize the predictive ability of the neural model. Table 1 shows the main hyperparameters used in this work.

The importance of the choice of parameters is also pointed out by [63], who focused on how to estimate the parameters as long as they are not dependent on each other, which is only possible when the number of parameters is small or otherwise it would involve a high computational cost. The research carried out by Smith [64] highlights how errors can be minimized with an adequate selection of hyperparameters, analyzing different ways to maximize the network performance.

5. Experimental results and discussion

The three time series analyzed are processed by the two selected Artificial Neural Network models, which have shown high efficiency in predicting economic time series with a higher precision than that of similar models [26,65,66]. They have been combined with the three preprocessing techniques described and applied to the three time series analyzed both with and without differentiation. For the sake of comparison, the two neural models have been also applied directly without preprocessing.

Differentiation has been applied directly to the original series when no preprocessing is applied (direct) and when EMD decomposition is used. On the other hand, when the series is decomposed into trend and fluctuations, differentiation is applied only to the former, as it preserves the rising trend of the series, which is not present in the latter. Therefore, since differentiation is applied to deal with this rising trend, it should be applied only to the trend series.

The EMD algorithm provided a decomposition into 10 IMFs and 1 residue in all cases. The decomposition provided by the Moving Average was performed with a window of 5 past data. That corresponding to the Wavelet Transform was obtained with a db5 wavelet (Daubechies wavelet with five vanishing moments) with a decomposition level of 1, i. e., only one decomposition is performed, which gives two components: a coarse approximation (low frequencies) and a detail signal (high frequencies). The latter one is removed, and the inverse transform is then applied only to the coarse approximation to obtain the denoised series, the trend. Both the EMD and Wavelet Transforms are built-in functions provided by the software used for the simulations, MATLAB, whose parameters were properly adjusted to obtain the best performance (several window sizes, wavelet functions, and decomposition levels were tested, and the best performance was obtained with the aforementioned function and values).

In Tables 2, 3, 4, 5 and 6 corresponding to the Apple, Nvidia, CO_2, S&P and WTI series respectively, the performance indices obtained with all the simulations carried out are presented. The results in each table have been grouped according to whether the time series were derived or not, in order to better compare the accuracy improvement obtained when using differentiation.

In all cases, the results obtained when the time series were not derived show the problems the neural models have in dealing with time series with a clear rising trend. No model was able to provide predictions that could be considered as useful. They are very poor. Only MLP with EMD was able to provide error indices with the CO_2 time series that could be considered as accurate enough to assume that this forecasting structure is reliable. When EMD was used with LSTM to forecast the same time series, the results obtained were not as good. It is worth noting that the models with EMD provided error indices that were slightly better than those provided by the other models. However, when differentiating the time series, EMD no longer provides the best

Table 1
Parameters selected.

Model	Learning rate	Batch	Epoch	Hidden Layers	Optimizer
LSTM	0.001	25	100	2	Adam
MLP	0.001	25	100	1	Levenberg-Marquardt

Table 2
Apple results.

		Metrics			
Time Serie	Preprocessing	MAPE	RMSE	MSE	R2
LSTM	Direct	43.043	77.457	5999.583	−2.182
	TF	51.775	98.153	9634.043	−3.622
	Wavelet	46.471	79.963	6394.197	−2.431
	EMD	**34.137**	**57.907**	**3353.330**	**−0.799**
MLP	Direct	41.359	74.002	5476.414	−1.937
	TF	30.120	56.298	3169.529	−0.720
	Wavelet	31.918	59.773	3572.913	−0.916
	EMD	**22.741**	**44.226**	**1956.015**	**−0.0490**
LSTM derived	Direct	1.379	2.452	6.23	0.996
	TF	2.033	3.481	12.118	0.993
	Wavelet	**1.290**	2.337	5.465	**0.997**
	EMD	1.358	**2.326**	**5.412**	**0.997**
MLP derived	Direct	1.367	2.463	6.068	0.996
	TF	1.472	2.645	6.997	0.996
	Wavelet	**1.196**	2.257	5.097	**0.997**
	EMD	1.203	**2.053**	**4.216**	**0.997**

Table 3
Nvidia results.

Metrics					
Time Serie	Preprocessing	MAPE	RMSE	MSE	R2
LSTM	Direct	36.583	232.717	54157.52	−0.460
	TF	59.465	291.492	84967.80	−1.292
	Wavelet	50.775	245.331	687.57	−0.605
	EMD	**36.131**	**78.125**	**6103.677**	**0.839**
MLP	Direct	41.671	232.991	54284.87	−0.448
	TF	38.511	228.307	52124.15	−0.394
	Wavelet	26.417	198.723	39490.91	−0.0533
	EMD	**14.519**	**38.624**	**1491.85**	**0.960**
LSTM derived	Direct	2.338	9.275	86.0280	0.997
	TF	3.446	12.948	167.663	0.995
	Wavelet	**2.258**	9.215	84.932	**0.997**
	EMD	2.401	**9.147**	**83.682**	**0.997**
MLP derived	Direct	2.177	8.659	74.991	0.997
	TF	2.685	11.680	136.433	0.996
	Wavelet	2.168	9.094	82.78	0.997
	EMD	**1.925**	**8.22**	**64.196**	**0.998**

Table 4
CO_2 results.

Metrics					
Time Serie	Preprocessing	MAPE	RMSE	MSE	R2
LSTM	Direct	24.245	27.578	760.580	−0.253
	TF	20.147	22.337	498.978	0.177
	Wavelet	24.157	26.469	700.632	−0.154
	EMD	6.641	6.148	37.800	0.937
MLP	Direct	17.937	20.786	432.059	0.288
	TF	14.465	17.095	292.249	0.518
	Wavelet	20.171	23.747	563.927	0.070
	EMD	1.994	1.752	3.070	0.994
LSTM derived	Direct	2.070	**1.432**	**2.053**	**0.996**
	TF	3.360	2.282	5.209	0.991
	Wavelet	**1.957**	1.466	2.151	**0.996**
	EMD	2.562	1.729	2.989	0.995
MLP derived	Direct	2.033	1.454	2.116	0.996
	TF	2.376	1.696	2.879	0.995
	Wavelet	**1.933**	**1.389**	**1.931**	**0.996**
	EMD	2.061	1.452	2.110	0.996

Table 5
S&P results.

Metrics					
Time Serie	Preprocessing	MAPE	RMSE	MSE	R2
LSTM	Direct	19.842	1011.594	1023322.348	−1.555
	TF	14.217	742.283	550984.201	−0.376
	Wavelet	18.427	945.805	894547.776	−1.234
	EMD	**5.276**	**281.332**	**79147.527**	**0.802**
MLP	Direct	5.305	288.848	83433.179	0.791
	TF	6.206	349.505	122154.027	0.694
	Wavelet	3.436	187.442	35134.608	0.912
	EMD	**0.959**	**59.548**	**3545.983**	**0.991**
LSTM derived	Direct	0.877	46.628	2174.157	0.994
	TF	1.250	63.088	3980.090	0.990
	Wavelet	**0.744**	**40.554**	**1644.650**	**0.995**
	EMD	0.794	42.001	1764.063	0.995
MLP derived	Direct	0.867	45.968	2113.091	0.994
	TF	0.866	46.100	2125.247	0.994
	Wavelet	0.691	37.225	1385.702	0.996
	EMD	**0.679**	**36.438**	**1327.729**	**0.996**

Table 6
WTI results.

Metrics					
Time Serie	Preprocessing	MAPE	RMSE	MSE	R2
LSTM	Direct	**2.425**	**2.299**	**5.283**	**0.989**
	TF	3.039	2.611	6.819	0.986
	Wavelet	3.252	2.887	8.335	0.983
	EMD	4.517	3.651	13.332	0.973
MLP	Direct	2.010	2.046	4.185	0.991
	TF	1.921	1.874	3.510	0.993
	Wavelet	**1.605**	2.008	4.030	0.992
	EMD	1.637	**1.521**	**2.313**	**0.995**
LSTM derived	Direct	2.081	2.068	4.276	0.991
	TF	2.707	2.403	5.774	0.988
	Wavelet	**1.708**	**1.889**	**3.568**	**0.993**
	EMD	2.445	2.412	5.819	0.988
MLP derived	Direct	1.976	2.087	4.356	0.991
	TF	1.863	1.848	3.415	0.993
	Wavelet	1.631	1.824	3.325	0.993
	EMD	**1.787**	**1.577**	**2.486**	**0.995**

performance. Now the decomposition into trend and fluctuations obtained with the wavelet filtering is the one that provides the more accurate predictions.

From the data presented in Tables 2, 3, 4, 5 and 6 it can be seen that differentiation is able to significantly improve the accuracy in all cases and for all time series. Only in one case, MLP with EMD with the CO2 time series, the prediction without differentiation gave slightly better results than those obtained with it. In the rest of the cases, the predictions were clearly bad and could not be used for any type or task: the accuracy was so poor that they could not provide any information on how these time series will evolve. These results prove that the smoothing effect provided by differentiation of the series allows an efficient training in which the shocks presented by the data can be more easily analyzed and predicted when the validation data are presented to the network. In other words, the models trained on the differentiated time series are able to generalize their behavior regardless of their trend and evolution.

In the case of the S&P and WTI series, a superior result is seen in the EMD methodology, for all models, only being surpassed by the direct methodology for the LSTM model without deriving from the WTI series, showing evidence that preprocessing substantially improves the predictions.

To better visualize the behavior of the predictions provided by the different forecasting structures tested, they are shown in a series of graphs (Figs. 12–23). Due to the large number of models tested, only a selection is shown. In each figure, graph a) shows the neural model without preprocessing, while graph b) shows the model consisting of the neural model and EMD. This preprocessing was chosen because it was the one that showed the "best" (actually less bad) performance in forecasting without differentiation. The forecasting model without preprocessing has also been presented to show that this simple model was able to perform as well as those with preprocessing when differentiation is used, as shown in Tables 2–6.

To assess the presence of statistically significant differences and to

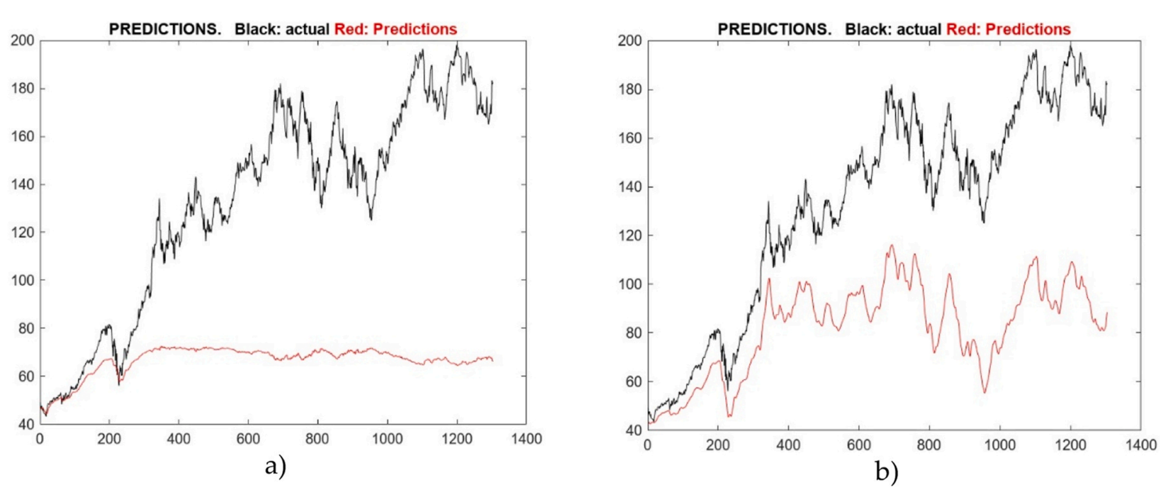


Fig. 12. Predictions with LSTM for the Apple time series. a) LSTM, b) LSTM+EMD.

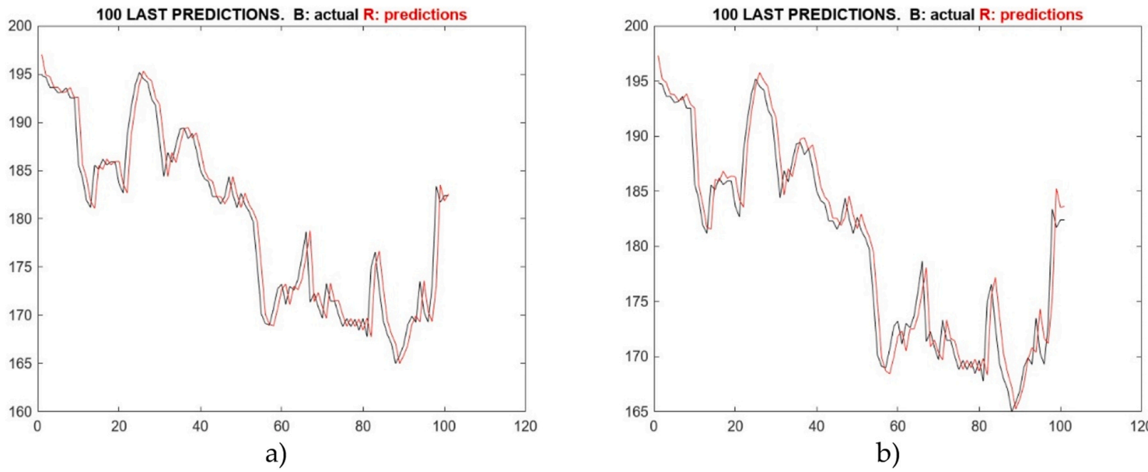


Fig. 13. 100 last predictions with LSTM for the derived Apple time series. a) LSTM, b) LSTM+EMD.

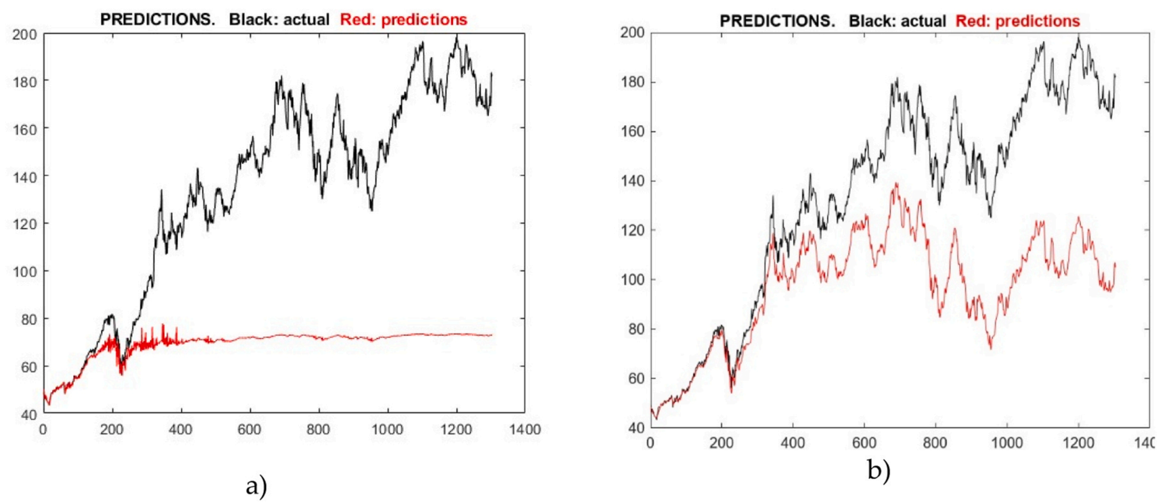


Fig. 14. Predictions with MLP for the Apple time series. a) MLP, b) MLP+EMD.

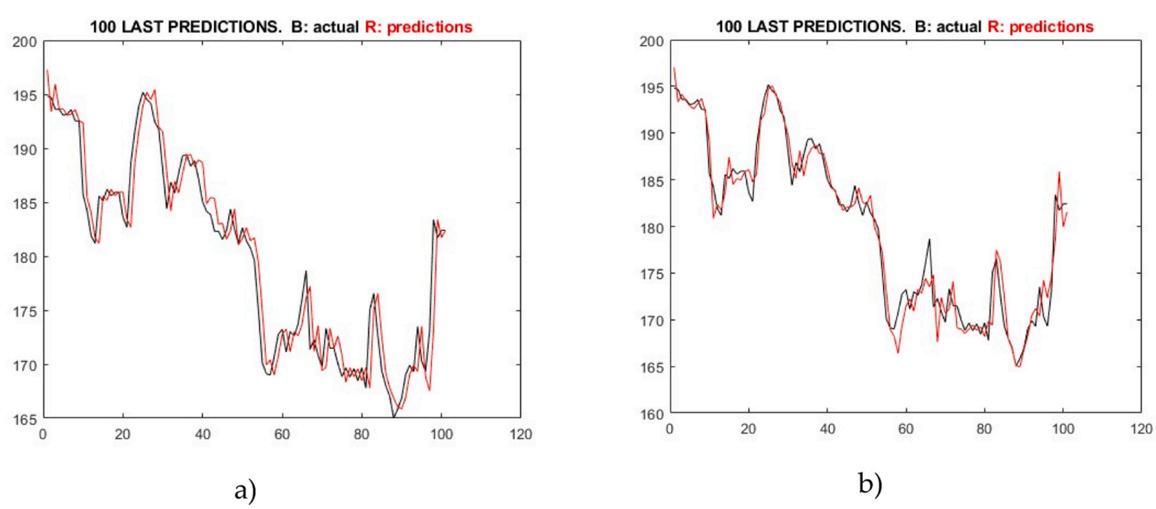


Fig. 15. 100 last predictions with MLP for the derived Apple time series. a) MLP, b) MLP+EMD.

validate the effectiveness of the neural network, the Friedman test is employed. This non-parametric test evaluates whether the null hypothesis, which posits that the mean predictions across the different methods are equivalent, can be rejected. The procedure involves the comparison of medians between groups of data. The Friedman test is specifically applied to the results produced by the neural network

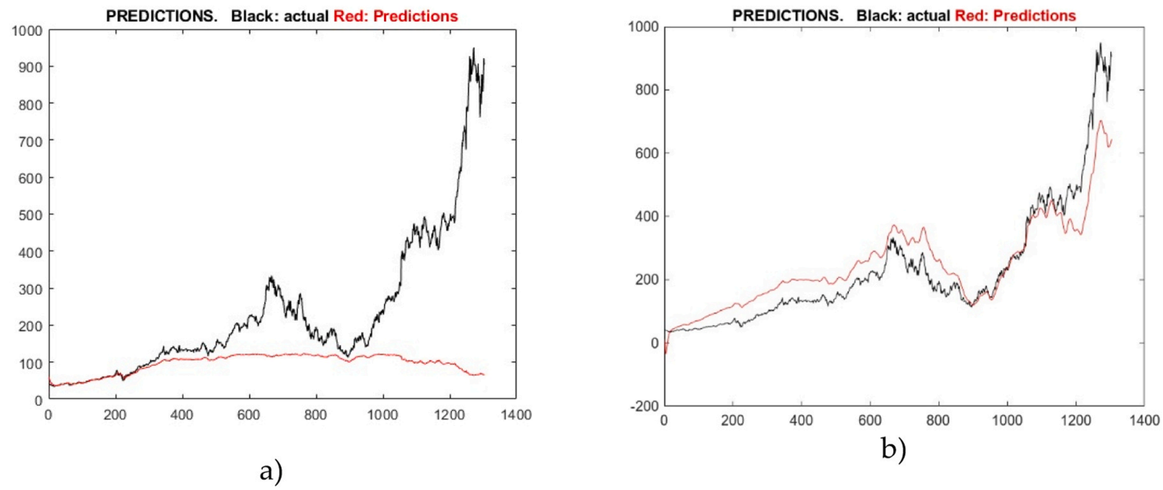


Fig. 16. Predictions with LSTM for the Nvidia time series. a) LSTM, b) LSTM+EMD.

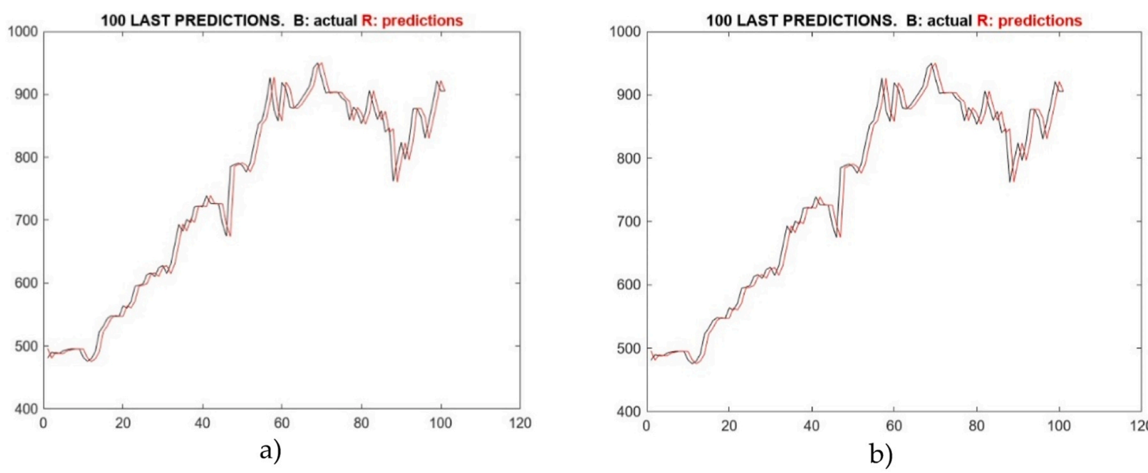


Fig. 17. 100 last predictions with LSTM for the derived Nvidia time series. a) LSTM, b) LSTM+EMD.

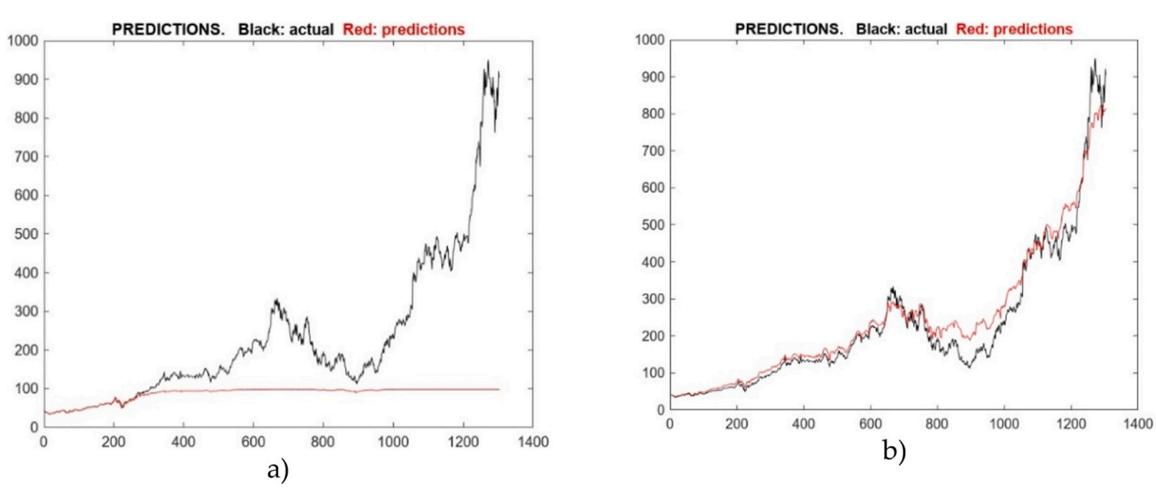
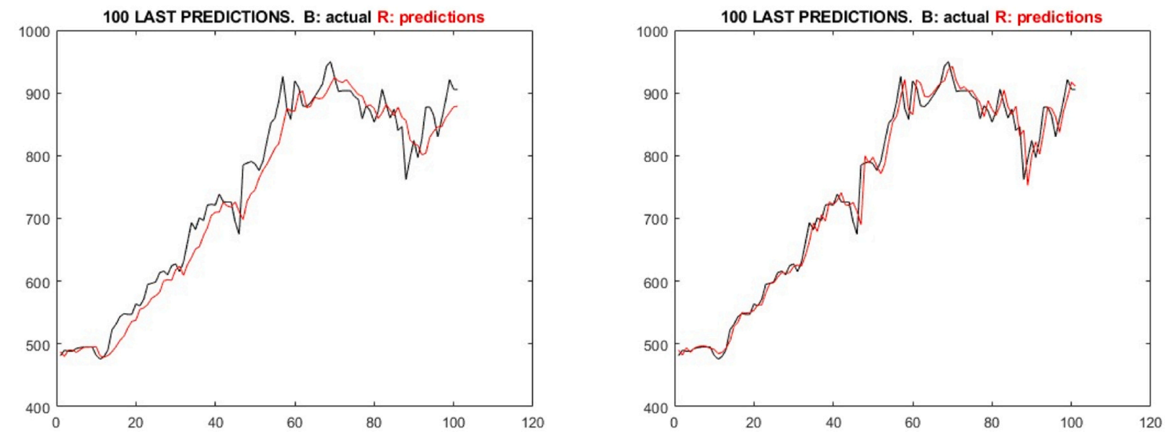


Fig. 18. Predictions with MLP for the Nvidia time series. a) MLP, b) MLP+EMD.

models, given that these models have demonstrated superior accuracy.

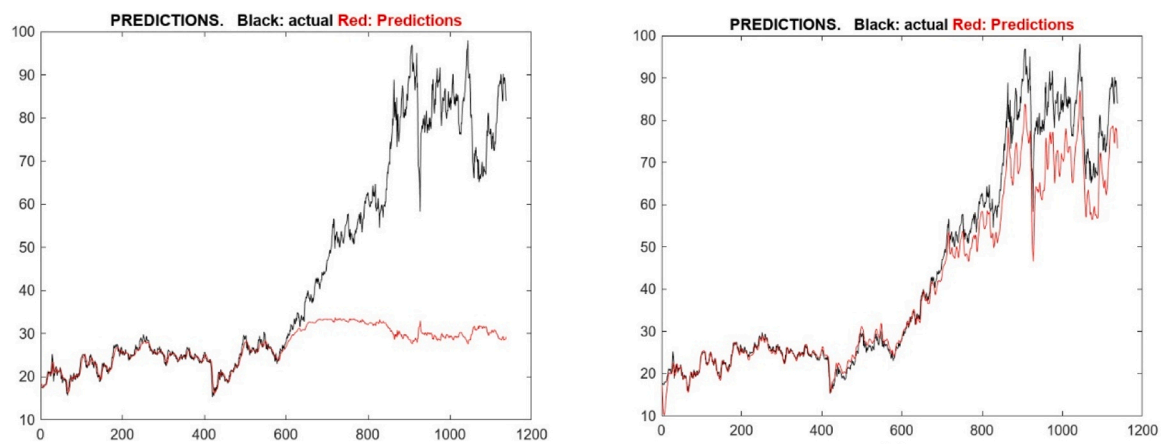
When the test results, incorporating all the models evaluated in the experiment, reveal a p-value below 0.05. This indicates that the null hypothesis can be rejected, providing enough evidence to determine that there are significant differences in the means of predictions. If significant differences are observed in the Friedman test, the Wilcoxon test is performed.

The Wilcoxon test, also known as the Wilcoxon signed-rank test, is a non-parametric statistical method used to compare two related or paired samples to determine if their medians differ significantly. This test is



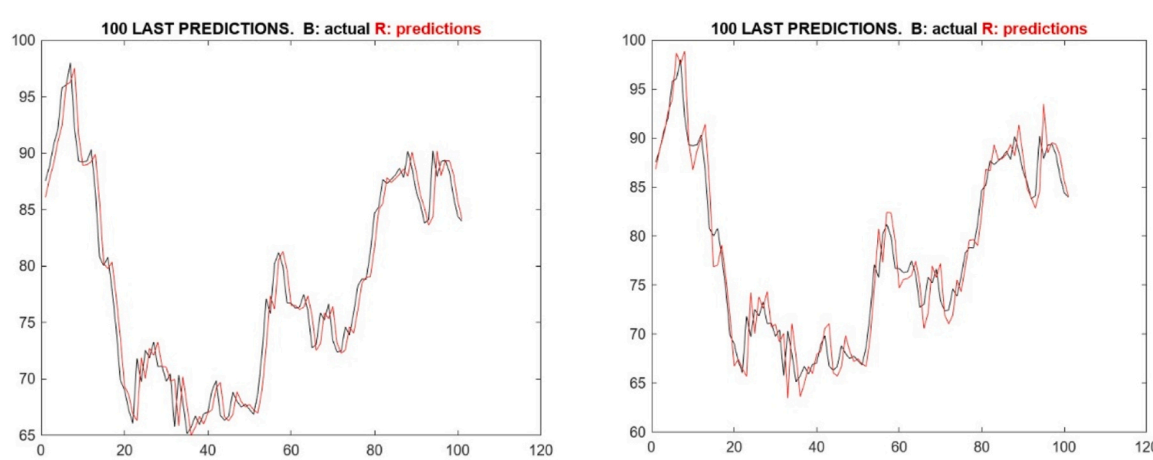
a) b)

Fig. 19. 100 last predictions with MLP for the derived Nvidia time series. a) MLP, b) MLP+EMD.



a) b)

Fig. 20. Predictions with LSTM for the CO_2 time series. a) LSTM, b) LSTM+EMD.



a) b)

Fig. 21. 100 last predictions with LSTM for the derived CO_2 time series. a) LSTM, b) LSTM+EMD.

particularly useful when the data do not meet the normality assumptions required for the paired *t*-test. It works by calculating the differences between paired observations, excluding any differences equal to zero. Absolute ranks are then assigned to the remaining differences, and the ranks are summed separately for positive and negative values. The smaller of these sums is the Wilcoxon statistic, which is compared to critical values or used to calculate a p-value. If the p-value is below a predefined significance level (usually p-value=0.05), the null

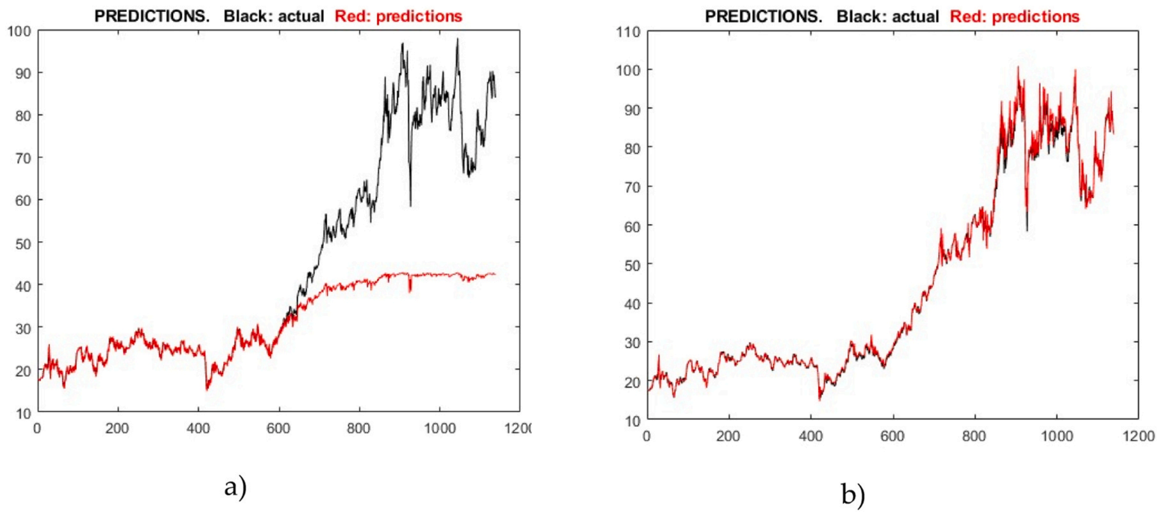


Fig. 22. Predictions with MLP for the CO_2 time series. a) MLP, b) MLP+EMD.

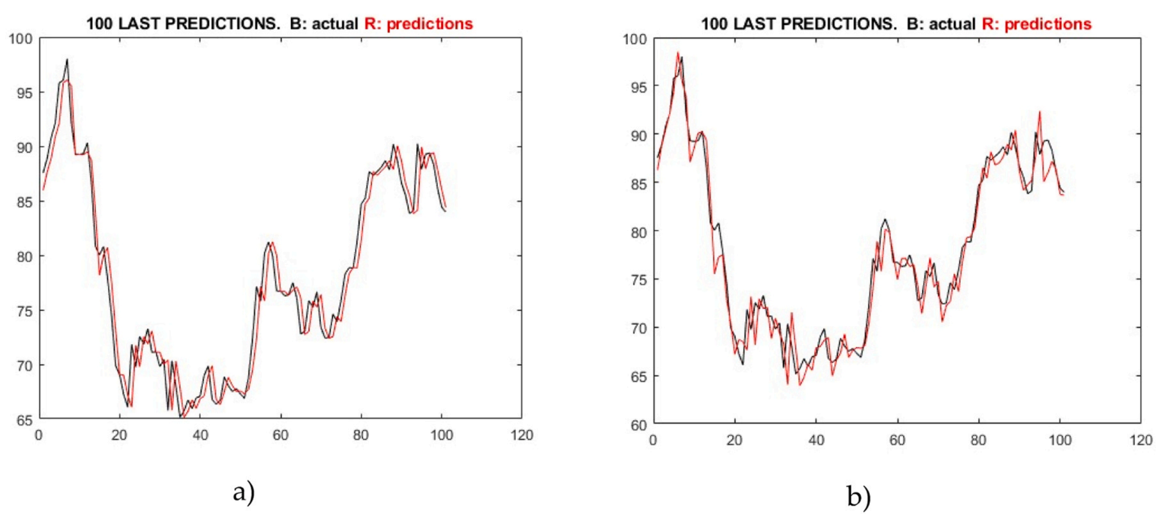


Fig. 23. 100 last predictions with MLP for the derived CO_2 time series. a) MLP, b) MLP+EMD.

hypothesis is rejected, indicating that the differences between the samples are statistically significant.

The result provided by the Friedman test with the Apple time series are shown in Table 7. It shows that at least two of the groups analyzed, taking into account the results obtained by the LSTM and MLP models for both the derived and underived series, show significant differences. After applying the Wilcoxon test (Table 8), the only group of results that allowed us to reject the null hypothesis are those derived, with an optimal result in the EMD methodology, which does not show significant differences with the original series for the two models.

The same process is carried out for the Nvidia time series, in which the Friedman test yields the results shown in Table 9.

The p-value shows that at least two of the groups show significant differences, so now, with the Wilcoxon test (Table 10), we will analyze which groups show the least significant differences with the original sample.

In this instance, the most favorable outcomes were achieved using the Wavelet methodology with the LSTM model and the EMD

Table 7
APPLE Friedman results.

Statistical Value	8153.422
***p*-Value**	0.0

Table 8
APPLE Wilcoxon results.

Compared Data	Statistical Value	*p*-Value
APPLE and EMD LSTM	422207.0	0.812
APPLE and EMD MLP	415690.0	0.473

Table 9
NVIDIA Friedman results.

Statistical Value	1412.080
***p*-Value**	1.383e−299

Table 10
NVIDIA Wilcoxon results.

Compared Data	Statistical Value	*p*-Value
NVIDIA and EMD LSTM	282378.0	7.164
NVIDIA and EMD MLP	367619.0	2.133

methodology with the MLP model, both demonstrating superior performance compared to the other approaches. The high statistical values obtained indicate that there are no significant differences with the original series, highlighting the effectiveness of these methodologies. This result underscores how differentiation significantly enhances performance, particularly in time series with a clear trend.

For the CO2 series, the result provided by the Friedman test is shown in Table 11. Once again, a large difference is shown, in this case greater than in the previous results, between at least two of the groups analyzed for both the derived and non-derived series and for the two models used, MLP and LSTM. This makes the unequal result obtained by the different methodologies with the different models analyzed more evident.

The most accurate result, with the least significant differences with respect to the original series, was obtained with the direct derivation methodology for the LSTM model and EMD for the MLP model (Table 12), which reflect the least significant differences with the original series. This result shows that depending on the characteristics of the time series, different types of preprocessing will achieve different results.

Once again, the results presented in Table 13 highlight that, within the entire set of S&P 500 time series, there are statistically significant differences in one of the two groups being analyzed. This indicates that the modifications applied during preprocessing or modeling introduce notable changes in the characteristics of the data, which may affect its comparability with the original series. To address this, the Wilcoxon signed-rank test is employed as a non-parametric statistical method to identify the pair of series that exhibits the smallest differences when compared to the original unprocessed series.

The results provided by the Wilcoxon test (Table 14) indicate that the combinations producing the least differences are those obtained using EMD for the MLP model and Wavelet for the LSTM model. These findings reinforce the notion that the integration of derivation and preprocessing techniques plays a pivotal role in improving model performance. Specifically, the pair involving Wavelet preprocessing with LSTM yielded a p-value of 0.776, indicating minimal differences when compared to the original series. On the other hand, the pair with EMD preprocessing and the MLP model achieved a p-value of 0.011, highlighting a relatively greater divergence but still showcasing notable improvements compared to results without advanced preprocessing.

Table 15 highlights the presence of significant differences among the samples of the WTI time series analyzed, as evidenced by the Friedman test results, which yield a statistical value of 144.612 and a p-value of 2.601e-27. These findings indicate a clear rejection of the null hypothesis, confirming that the performance metrics across the studied models are not identical. Consequently, it becomes essential to conduct a more detailed analysis using the Wilcoxon test to pinpoint the specific pairs of datasets or model outputs that exhibit the smallest discrepancies when compared.

The values obtained (Table 16) indicate that both methodologies achieve relatively small differences compared to the original series, further emphasizing the benefits of incorporating advanced preprocessing techniques such as Wavelet and EMD. Combining these techniques with the respective deep learning models improves the accuracy of the predictions by adding drift to the trended time series, although this is not as marked as in the previous cases.

Figs. 12–15 show the results obtained with the Apple time series. Fig. 8 shows the results with LSTM without differentiation: a) LSTM alone, and b) LSTM+EMD. Fig. 13 shows the last 100 predictions of the same models with differentiation. This reduced set of predictions was chosen to highlight the good accuracy of the models with differentiation, since these data should be the most difficult to predict. Figs. 10 and 11 repeat this distribution of results, but with MLP. The results shown in Fig. 12 reveal that the neural model, when applied without any preprocessing, exhibited a saturation effect, failing to provide predictions beyond a certain threshold. This limitation suggests that the model struggles to extrapolate values significantly higher than those seen during training, which is a well-documented challenge in forecasting steeply rising trends. However, with the inclusion of Empirical Mode Decomposition (EMD), the forecasting model demonstrated a notable improvement in capturing the overall structure of the series. Although it still could not fully reach the highest values observed in the actual data, it was able to approximate the general trend more effectively, reducing some of the discrepancies found in the baseline neural model. Additionally, the results presented in Figs. 13 and 15 highlight the impact of differentiation as a preprocessing step. By applying differentiation, the forecasting models were not only capable of capturing the underlying trend more precisely but also exhibited a significant reduction in error when predicting extreme values. This improvement suggests that differentiation plays a crucial role in stabilizing the series, allowing the model to focus on meaningful fluctuations rather than being hindered by the steep upward trajectory. The ability of differentiated models to align closely with the actual time series behavior further supports the idea that preprocessing techniques tailored to the specific characteristics of economic data can significantly enhance forecasting accuracy.

The same distribution of graphical results is observed for the Nvidia time series in Figs. 16–19, following a similar pattern to the previously analyzed cases. Once again, these results highlight the inherent limitations of neural models when dealing with rapidly increasing trends in time series forecasting. Specifically, the models without any preprocessing struggle to capture the full extent of the upward movement, resulting in predictions that plateau beyond a certain threshold. This

Table 11
CO2 Friedman results.

Statistical Value	103.249
***p*-Value**	9.237e−19

Table 12
CO2 Wilcoxon results.

Compared Data	Statistical Value	*p*-Value
CO2 and Direct LSTM	289711.0	0.051
CO2 and EMD MLP	300086.0	0.030

Table 13
S&P Friedman results.

Statistical Value	495.482
***p*-Value**	6.554e−102

Table 14
S&P Wilcoxon results.

Compared Data	Statistical Value	*p*-Value
S&P and Wavelet LSTM	422219.0	0.776
S&P and EMD MLP	391592.0	0.011

Table 15
WTI Friedman results.

Statistical Value	144.612
***p*-Value**	2.601e−27

Table 16
S&P Wilcoxon results.

Compared Data	Statistical Value	*p*-Value
WTI and WAVELET LSTM	1617012.0	0.320
WTI and EMD MLP	1596882.0	0.126

saturation effect suggests that the model lacks the capacity to extrapolate beyond the patterns seen during training, leading to systematic underestimations in the later stages of the series.

However, when Empirical Mode Decomposition (EMD) is incorporated into the prediction model, a notable improvement is observed. The decomposition process enables the model to break down the complex structure of the time series into more manageable components, allowing it to generate forecasts that more closely follow the actual trend. While the predictions remain somewhat constrained in terms of accuracy, especially in capturing extreme values, they still exhibit a better approximation of the overall trajectory compared to the baseline neural model.

Furthermore, the inclusion of differentiation in the forecasting model proves to be a crucial factor in enhancing prediction accuracy. By applying differentiation, the model gains the ability to better capture both the local variations and the general evolution of the time series, mitigating issues related to trend saturation. As a result, the forecasting model can produce more reliable predictions, aligning more effectively with the real data and demonstrating the importance of appropriate preprocessing techniques in improving neural network performance for financial time series forecasting.

The results obtained with the CO_2 time series are shown in Figs. 20–23. They are similar to those obtained with the other time series, although here a good result was obtained with MLP+EMD, as good as that obtained when differentiation was included in the forecasting model. A less accurate result was also obtained with LSTM+EMD, although it was slightly better than the others. These two results prove the beneficial effect of including EMD in the prediction model, since it has allowed the two neural models to improve their accuracy in all the cases studied, although only with the CO_2 time series, up to a significant level. These good results could be obtained because moderately high values were included in the training dataset of CO_2, what could help the models with EMD be able to adequately capture the rising behavior of the series. The neural network alone and the models with decomposition into trend and fluctuations were not able to capture it.

The inclusion of differentiation allows all the forecasting models to improve their accuracy to really good levels, overcoming the problems that the forecasting models tested had with the rising trend of the forecasted time series.

For the S&P time series results, (Figs. 24–27) present the outcomes obtained through the combination of different methodologies with the various models under study. A clear superior performance is observed for the Wavelet methodology when applied to the LSTM model. Without any differentiation applied to the series, the models struggle to adequately capture the time series behavior, even though this particular series does not exhibit as pronounced an upward trend as others analyzed in the study.

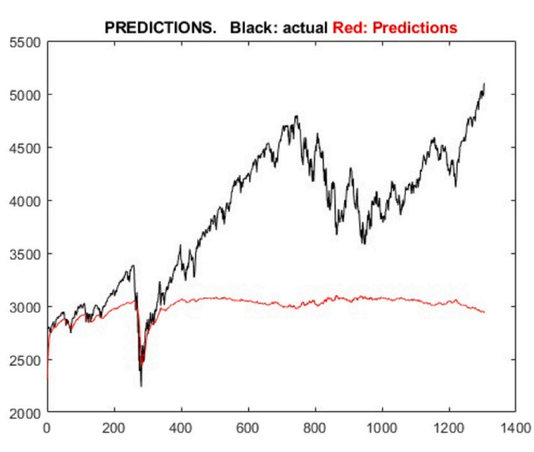
Specifically, when no type of derivation is applied, Fig. 24 a) highlights how the LSTM model fails to align its predictions closely with the original data, producing a graph with noticeable deviations. However, the application of the Wavelet methodology significantly enhances the model's performance, as evidenced by a much tighter graph that effectively follows the original data trend, thereby considerably reducing the prediction error.

Similarly, for the MLP model, the lack of preprocessing leads to results that are not fully adjusted, but a remarkable improvement is observed when the EMD methodology is applied as a preprocessing step. This combination allows for a substantial reduction in the error, resulting in better and more reliable predictions. It is particularly noteworthy that for this time series, the application of differentiation consistently improved the accuracy across all methodologies and models. Even in the case of MLP+EMD, which already delivered very good accuracy without differentiation, the application of differentiation further enhanced the performance, underscoring its efficacy in refining model predictions and reducing errors.

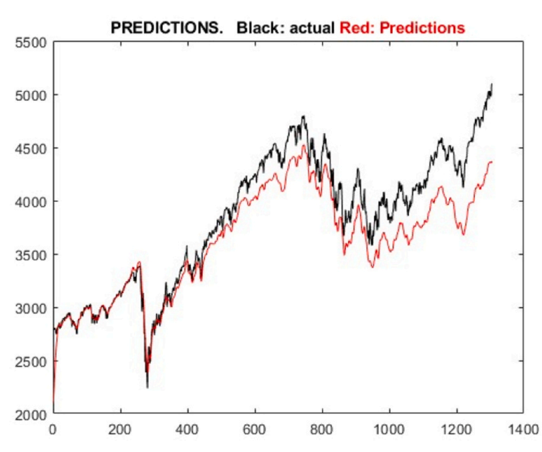
Finally, the WTI time series is modelled, addressing a dataset that differs from the previously analyzed series due to the absence of a clearly marked upward trend (Figs. 28–31). In contrast, the WTI series exhibits frequent fluctuations and irregular variations, which presents a greater challenge for DL models. These complexities make the task of accurate prediction more demanding, particularly when compared to time series with more consistent trends.

Despite these inherent challenges, the use of derivation and preprocessing techniques proves essential to improve model performance, although the improvement is not as pronounced as in the other analyzed time series. The application of the Wavelet transform methodology to LSTM models and EMD to MLP models provides the best predictive accuracy among the tested approaches. These preprocessing methodologies help mitigate the difficulties posed by the irregularities of the WTI series. Specifically, the Wavelet transform facilitates the LSTM model's ability to isolate relevant features and minimize the influence of noise. This specific approach improves the predictive capability of the model. On the other hand, the EMD methodology effectively extracts intrinsic mode functions, allowing the MLP model to capture the underlying processes driving oil price variations over time.

Differencing of the time series further improves the accuracy of the LSTM model in all cases, highlighting the importance of capturing dynamic changes in the data. For the MLP model, differencing also contributes to improved performance in most cases, particularly when combined with simpler preprocessing techniques such as TF. However,

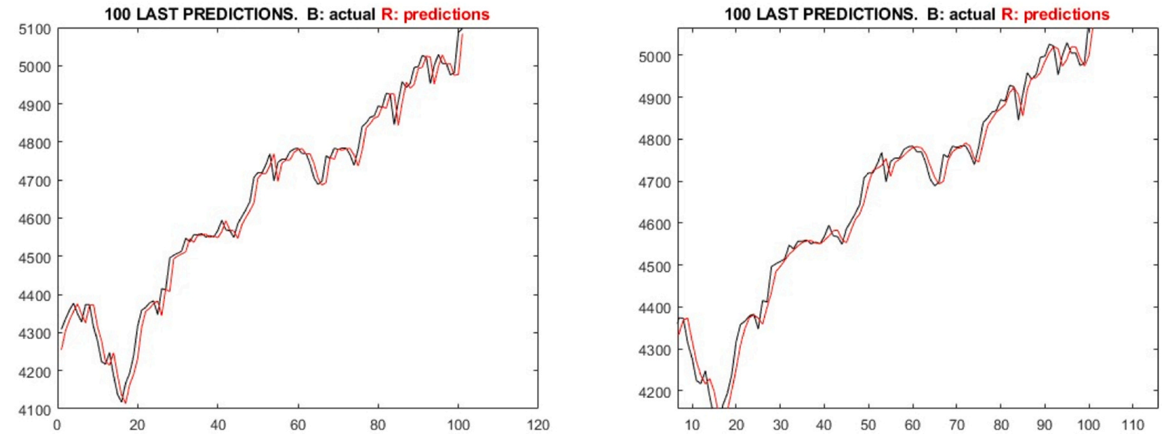


a)



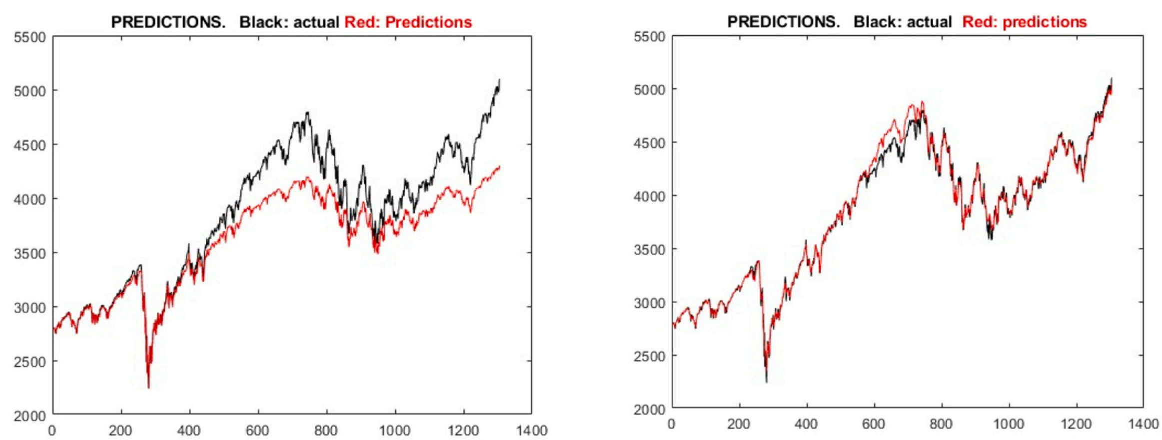
b)

Fig. 24. Predictions with LSTM for the S&P time series. a) LSTM, b) LSTM+Wavelet.



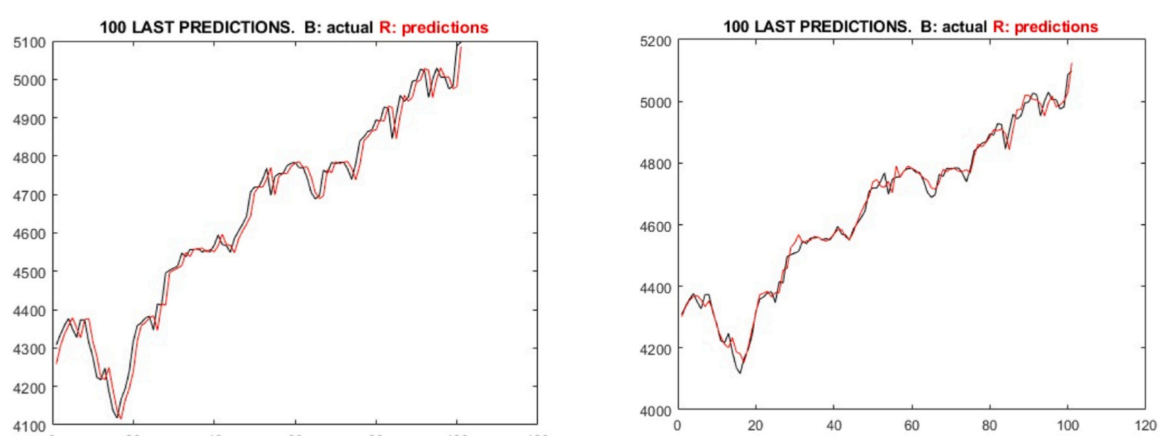
a) b)

Fig. 25. 100 last predictions with LSTM for the derived S&P time series. a) LSTM, b) LSTM+Wavelet.



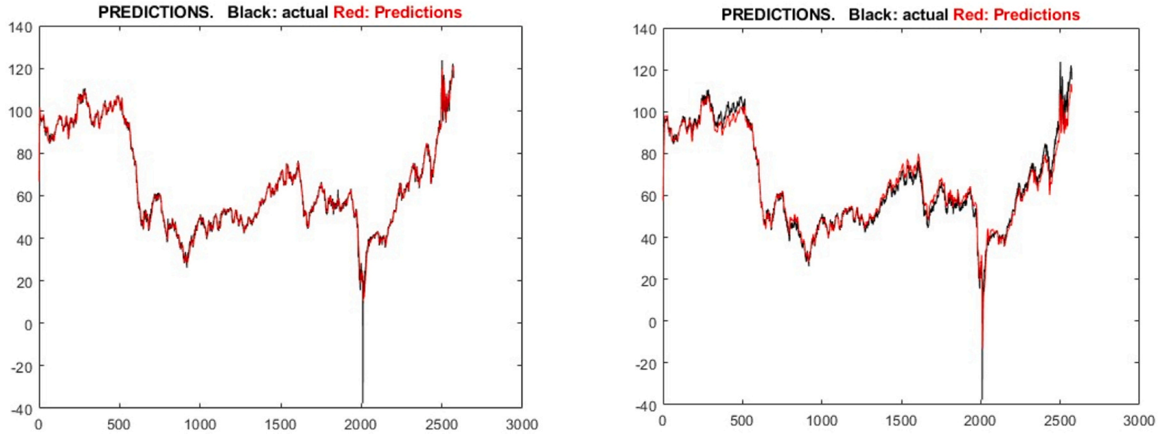
a) b)

Fig. 26. Predictions with MLP for the S&P time series. a) MLP, b) MLP+EMD.



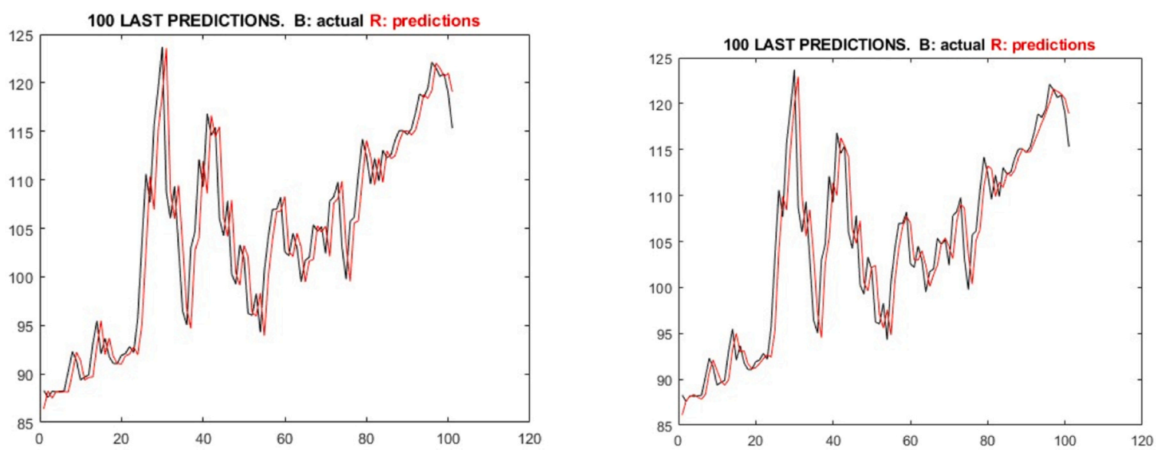
a) b)

Fig. 27. 100 last predictions with MLP for the derived S&P time series. a) MLP, b) MLP+EMD.



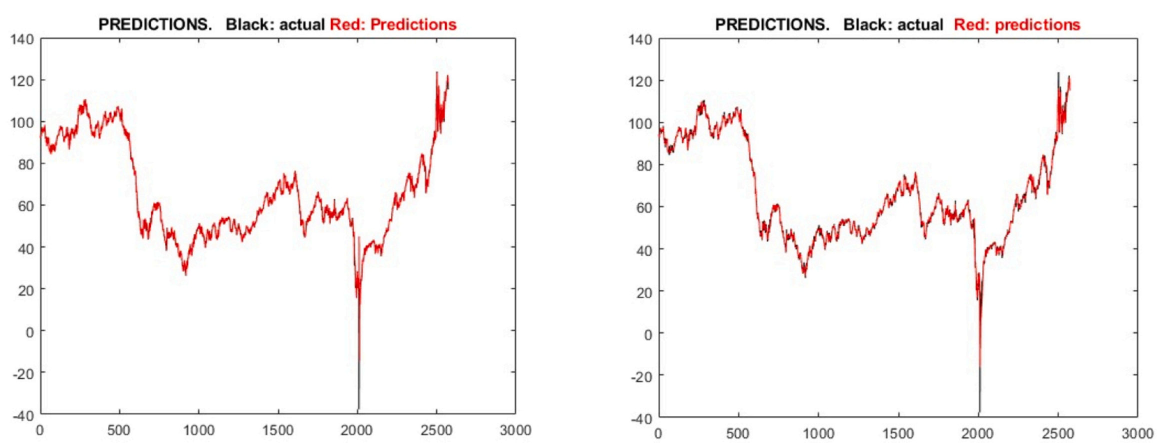
a) b)

Fig. 28. Predictions with LSTM for the WTI time series. a) LSTM, b) LSTM+Wavelet.



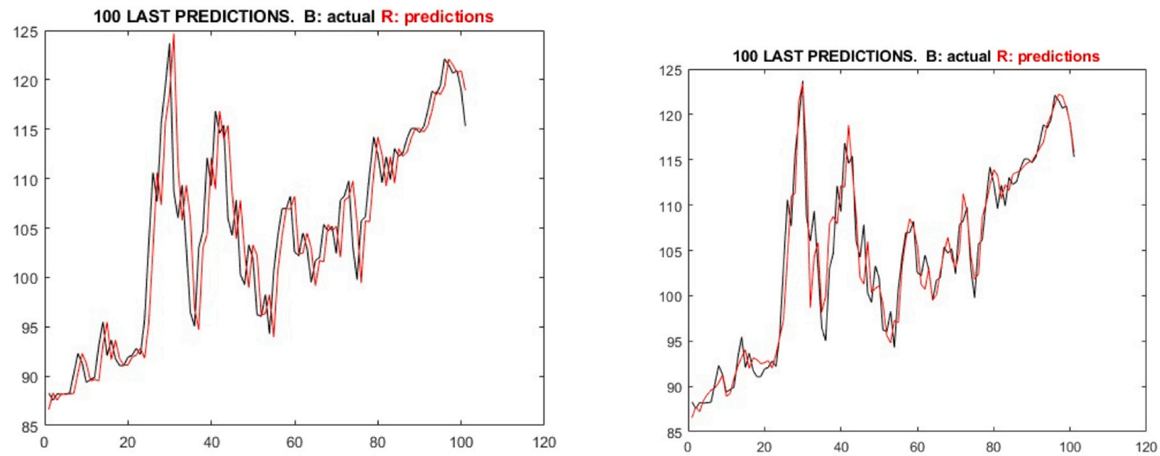
a) b)

Fig. 29. 100 last predictions with LSTM for the derived WTI time series. a) LSTM, b) LSTM+Wavelet.



a) b)

Fig. 30. Predictions with MLP for the WTI time series. a) MLP, b) MLP+EMD.



a) b)

Fig. 31. 100 last predictions with MLP for the derived WTI time series. a) MLP, b) MLP+EMD.

in cases where Wavelet filtering or EMD decomposition is applied, differencing only reaches accuracy levels comparable to those obtained without it. This plateau in performance suggests that the MLP+Wavelet and MLP+EMD combinations may have reached their maximum predictive capability, where additional preprocessing tools such as differencing no longer provide significant improvements.

6. Discussion and limitations

There is an extensive literature on the preprocessing of time series [67,68], since correct preprocessing prior to prediction leads to better error metrics and tighter results [69]. However, it is not easy to find studies on how to solve accuracy problems with time series that have a steeply rising trend.

The results obtained in this study demonstrate that two widely used neural models for time series forecasting, which have previously delivered accurate and reliable predictions, were unable to achieve satisfactory results when forecasting time series with a steeply rising trend. Even after applying some of the most widely utilized preprocessing techniques, such as EMD and decomposition into trend and fluctuations with Moving Average and Wavelet [18,70], the performance remained unsatisfactory. However, after differentiating the time series, a substantial improvement of more than 96 % in the MAPE metric was observed across all the series and neural networks tested. This significant enhancement in the MAPE score was consistently supported by improvements in other error indices, reinforcing the effectiveness of time series differentiation. These findings underscore the importance of differentiation as a key preprocessing step for improving forecasting accuracy in time series with strong upward trends, surpassing the capabilities of more traditional methods.

Despite the promising results obtained in this study, several limitations must be acknowledged. First, the performance of neural network models is highly sensitive to the selection of hyperparameters, such as the number of layers, the number of neurons per layer, and the choice of activation functions. Small variations in these parameters can lead to significant differences in predictive accuracy, requiring extensive tuning and computational resources.

Second, the quality and availability of data play a crucial role in the effectiveness of the proposed forecasting models. Economic time series often contain noise, missing values, or structural breaks, which can negatively impact model performance. Although preprocessing techniques were employed to mitigate these issues, they cannot completely eliminate the inherent challenges associated with economic data.

Third, while neural networks excel at capturing nonlinear patterns, they struggle to account for exogenous factors and sudden market disruptions, such as financial crises, policy changes, or external economic shocks. This limitation reduces the model's robustness when faced with unexpected events that significantly alter the trajectory of the time series.

7. Conclusions

The complexity inherent in economic time series requires the development of simple and effective techniques that allow for predictions with a high degree of fit and precision. Although preprocessing techniques, along with data normalization, improve error metrics, they are insufficient when the time series to be forecast exhibits a steeply rising trend, making additional techniques necessary. In this work, the proposed time series differentiation offers a simple and effective solution to smooth the time series (or the trend sub-series when decomposed into trend and fluctuations), enabling the corresponding forecasting model to overcome the challenges of forecasting time series with a strong upward trend.

The results obtained with the proposed methodology show that a single preprocessing step, such as time series differentiation, is sufficient to improve the error metrics of forecasting models, both when using simple forecasting tools and when complemented with additional preprocessing techniques, such as EMD or decomposition into trend and fluctuations using Moving Average or Wavelet. These results suggest that the structure presented in this paper (differentiation + preprocessing + neural network) could be replicated and tested with other preprocessing models and neural structures. It would be valuable to conduct further tests with time series exhibiting similar behavior to the ones studied in this work, in order to assess the performance of the methodology in diverse contexts, ensuring that the results are as general as possible.

Furthermore, modifying the neural structures to enable multistep predictions would open new opportunities for improvement. This adaptation could involve verifying the reliability of predictions across different time horizons, which would provide more robust and useful forecasting models. In this regard, the ability to extend predictions beyond a single time step could be crucial for practical applications, such as long-term economic and financial decision-making.

Funding

This work was supported in part by the project TED2021–131671B-I00 funded by MCIN/AEI /10.13039/501100011033 and by the

European Union NextGenerationEU/ PRTR.

CRediT authorship contribution statement

Jaramillo-Morán Miguel A.: Writing – review & editing, Writing – original draft, Methodology, Investigation, Data curation. **Lazcano Ana:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Conceptualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

In this section, you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Data availability

Data will be made available on request.

References

[1] G.P. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, Eur. J. Oper. Res 160 (2005) 501–514, https://doi.org/10.1016/j.ejor.2003.08.037.

[2] J. Lago, F. De Ridder, B. De Schutter, Forecasting spot electricity prices: deep learning approaches and empirical comparison of traditional algorithms, Appl. Energy 221 (2018) 386–405, https://doi.org/10.1016/j.apenergy.2018.02.069.

[3] E. Viviani, L. Di Persio, M. Ehrhardt, Energy markets forecasting. from inferential statistics to machine learning: the German case, Energies 14 (2021) 364, https://doi.org/10.3390/en14020364.

[4] C. Ubal, G. Di-Giorgi, J.E. Contreras-Reyes, R. Salas, Predicting the long-term dependencies in time series using recurrent artificial neural networks, Mach. Learn Knowl. Extr. 5 (2023) 1340–1358, https://doi.org/10.3390/make5040068.

[5] G. Bak, Y. Bae, Predicting the amount of electric power transaction using deep learning methods, Energies 13 (2020) 6649, https://doi.org/10.3390/en13246649.

[6] J. Szoplik, Forecasting of natural gas consumption with artificial neural networks, Energy 85 (2015) 208–220, https://doi.org/10.1016/j.energy.2015.03.084.

[7] A.H. Moghaddam, M.H. Moghaddam, M. Esfandyari, Stock market index prediction using artificial neural network, J. Econ. Financ. Adm. Sci. 21 (2016) 89–93, https://doi.org/10.1016/j.jefas.2016.07.002.

[8] M. Göçken, M. Özçalıcı, A. Boru, A.T. Dosdoğru, Integrating metaheuristics and Artificial Neural Networks for improved stock price prediction, Expert Syst. Appl. 44 (2016) 320–331, https://doi.org/10.1016/j.eswa.2015.09.029.

[9] D. Keles, J. Scelle, F. Paraschiv, W. Fichtner, Extended forecast methods for day-ahead electricity spot prices applying artificial neural networks, Appl. Energy 162 (2016) 218–230, https://doi.org/10.1016/j.apenergy.2015.09.087.

[10] X. Fan, S. Li, L. Tian, Chaotic characteristic identification for carbon price and an multi-layer perceptron network prediction model, Expert Syst. Appl. 42 (2015) 3945–3952.

[11] M. Han, L. Ding, X. Zhao, W. Kang, Forecasting carbon prices in the Shenzhen market, China: the role of mixed-frequency factors, Energy 171 (2019) 69–76, https://doi.org/10.1016/j.energy.2019.01.009.

[12] T. Ciechulski, S. Osowski, High precision LSTM model for short-time load forecasting in power systems, Energies 14 (2021) 2983, https://doi.org/10.3390/en14112983.

[13] B.-S. Kwon, R.-J. Park, K.-B. Song, Short-term load forecasting based on deep neural networks using LSTM layer, J. Electr. Eng. Technol. 15 (2020) 1501–1509, https://doi.org/10.1007/s42835-020-00424-7.

[14] J.H. Stock, M.W. Watson, Variable trends in economic time series, J. Econ. Perspect. 2 (1988) 147–174, https://doi.org/10.1257/jep.2.3.147.

[15] J.H. Zhao, Z. Dong, Z. Xu, Effective Feature Preprocessing for Time Series Forecasting, in: X. Li, O.R. Zaïane, Z. Li (Eds.), Adv. Data Min. Appl, Springer, Berlin, Heidelberg, 2006, pp. 769–781, https://doi.org/10.1007/11811305_84.

[16] A. Famili, W.-M. Shen, R. Weber, E. Simoudis, Data preprocessing and intelligent data analysis, Intell. Data Anal. 1 (1997) 3–23, https://doi.org/10.3233/IDA-1997-1102.

[17] J.A. Cortés-Ibáñez, S. González, J.J. Valle-Alonso, J. Luengo, S. García, F. Herrera, Preprocessing methodology for time series: an industrial world application case study, Inf. Sci. 514 (2020) 385–401, https://doi.org/10.1016/j.ins.2019.11.027.

[18] S. Fong, K. Lan, P. Sun, S. Mohammed, J. Fiaidhi, S. Mohammed, A time-series pre-processing methodology for biosignal classification using statistical feature extraction, Proc. 10th IASTED Int. Conf. Biomed. Eng. Biomed. 13 (2013) 207–214.

[19] M.A. Jaramillo-Morán, D. Fernández-Martínez, A. García-García, D. Carmona-Fernández, Improving artificial intelligence forecasting models performance with data preprocessing: European union allowance prices case study, Energies 14 (2021) 7845, https://doi.org/10.3390/en14237845.

[20] Y. Jin, H. Guo, J. Wang, A. Song, A hybrid system based on LSTM for short-term power load forecasting, Energies 13 (2020) 6241, https://doi.org/10.3390/en13236241.

[21] H. Zheng, J. Yuan, L. Chen, Short-term load forecasting using EMD-LSTM neural networks with a Xgboost algorithm for feature importance evaluation, Energies 10 (2017) 1168, https://doi.org/10.3390/en10081168.

[22] N. Li, J. Dong, L. Liu, H. Li, J. Yan, A novel EMD and causal convolutional network integrated with Transformer for ultra short-term wind power forecasting, Int J. Electr. Power Energy Syst. 154 (2023) 109470, https://doi.org/10.1016/j.ijepes.2023.109470.

[23] H. Han, H. Neira-Molina, A. Khan, M. Fang, H.A. Mahmoud, E.M. Awwad, et al., Advanced series decomposition with a gated recurrent unit and graph convolutional neural network for non-stationary data patterns, J. Cloud Comput. 13 (2024) 20, https://doi.org/10.1186/s13677-023-00560-1.

[24] X. Guo, Q. Zhao, D. Zheng, Y. Ning, Y. Gao, A short-term load forecasting model of multi-scale CNN-LSTM hybrid neural network considering the real-time electricity price, Energy Rep. 6 (2020) 1046–1053, https://doi.org/10.1016/j.egyr.2020.11.078.

[25] A. Rosato, R. Araneo, A. Andreotti, F. Succetti, M. Panella, 2-D convolutional deep neural network for the multivariate prediction of photovoltaic time series, Energies 14 (2021) 2392, https://doi.org/10.3390/en14092392.

[26] W. Yang, J. Shi, S. Li, Z. Song, Z. Zhang, Z. Chen, A combined deep learning load forecasting model of single household resident user considering multi-time scale electricity consumption behavior, Appl. Energy 307 (2022) 118197, https://doi.org/10.1016/j.apenergy.2021.118197.

[27] L. Ji, Y. Zou, K. He, B. Zhu, Carbon futures price forecasting based with ARIMA-CNN-LSTM model, Procedia Comput. Sci. 162 (2019) 33–38, https://doi.org/10.1016/j.procs.2019.11.254.

[28] T. Lin, T. Guo, K. Aberer, Hybrid neural networks for learning the trend in time series, Proc. Twenty-Sixth Int Jt Conf. Artif. Intell. (2017), https://doi.org/10.24963/ijcai.2017/316.

[29] M. Qi, G.P. Zhang, Trend time–series modeling and forecasting with neural networks, IEEE Trans. Neural Netw. 19 (2008) 808–816, https://doi.org/10.1109/TNN.2007.912308.

[30] C. Paoli, C. Voyant, M. Muselli, M.-L. Nivet, Forecasting of preprocessed daily solar radiation time series using neural networks, Sol. Energy 84 (2010) 2146–2160, https://doi.org/10.1016/j.solener.2010.08.011.

[31] S. Khedhiri, Comparison of SARFIMA and LSTM methods to model and to forecast Canadian temperature, Reg. Stat. 12 (2022) 177–194.

[32] A. Lazcano, M.A. Jaramillo-Morán, J.E. Sandubete, Back to basics: the power of the multilayer perceptron in financial time series forecasting, Mathematics 12 (2024) 1920, https://doi.org/10.3390/math12121920.

[33] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (1989) 359–366, https://doi.org/10.1016/0893-6080(89)90020-8.

[34] P.H. Borghi, O. Zakordonets, J.P. Teixeira, A COVID-19 time series forecasting model based on MLP ANN, Procedia Comput. Sci. 181 (2021) 940–947, https://doi.org/10.1016/j.procs.2021.01.250.

[35] Zhang T., Zhang Y., Cao W., Bian J., Yi X., Zheng S., et al. Less Is More: Fast Multivariate Time Series Forecasting with Light Sampling-oriented MLP Structures 2022. https://doi.org/10.48550/arXiv.2207.01186.

[36] Madhusudhanan K., Jawed S., Schmidt-Thieme L. Hyperparameter Tuning MLPs for Probabilistic Time Series Forecasting 2024. https://doi.org/10.48550/arXiv.2403.04477.

[37] L. Wanchen, Analysis on the weight initialization problem in fully-connected multi-layer perceptron neural network, Int. Conf. Artif. Intell. Comput. Eng. ICAICE 2020 (2020) 150–153, https://doi.org/10.1109/ICAICE51518.2020.00035.

[38] Zhang A., Lipton Z.C., Li M., Smola A.J. Dive into Deep Learning nd:996.

[39] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, Neural Comput. 9 (1997) 1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735.

[40] F.A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: continual prediction with LSTM, Neural Comput. 12 (2000) 2451–2471, https://doi.org/10.1162/089976600300015015.

[41] Y. Ren, X. Qiu, P.N. Suganthan, Empirical mode decomposition based adaboost-backpropagation neural network method for wind speed forecasting, 2014 IEEE Symp. Comput. Intell. Ensemble Learn. CIEL (2014) 1–6, https://doi.org/10.1109/CIEL.2014.7015741.

[42] N.E. Huang, Z. Wu, S.R. Long, Hilbert-Huang transform, Scholarpedia 3 (2008) 2544, https://doi.org/10.4249/scholarpedia.2544.

[43] D.R. Osborn, Moving average detrending and the analysis of business cycles, Oxf. Bull. Econ. amp; Stat. | EBSCOhost 57 (1995) 547, https://doi.org/10.1111/j.1468-0084.1995.tb00039.x.

[44] S. Arianos, A. Carbone, Detrending moving average algorithm: a closed-form approximation of the scaling law, Phys. Stat. Mech. Appl. 382 (2007) 9–15, https://doi.org/10.1016/j.physa.2007.02.074.

[45] E. Gonzalez-Romera, M.A. Jaramillo-Moran, D. Carmona-Fernandez, Monthly electric energy demand forecasting based on trend extraction, IEEE Trans. Power Syst. 21 (2006) 1946–1953, https://doi.org/10.1109/TPWRS.2006.883666.

[46] S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Trans. Pattern Anal. Mach. Intell. 11 (1989) 674–693, https://doi.org/10.1109/34.192463.
[47] S.R.M. Penedo, M.L. Netto, J.F. Justo, Designing digital filter banks using wavelets, EURASIP J. Adv. Signal Process 2019 (2019) 33, https://doi.org/10.1186/s13634-019-0632-6.
[48] L. Zhang, P.N. Suganthan, A survey of randomized algorithms for training neural networks, Inf. Sci. 364–365 (2016) 146–255, https://doi.org/10.1016/j.ins.2016.01.039.
[49] A. Botchkarev, Performance metrics (Error Measures) in machine learning regression, forecasting and prognostics: properties and typology, Inter. J. Inf. Knowl. Manag 14 (2019) 045–076, https://doi.org/10.28945/4184.
[50] A. Olawoyin, Y. Chen, Predicting the future with artificial neural network, Procedia Comput. Sci. 140 (2018) 383–392, https://doi.org/10.1016/j.procs.2018.10.300.
[51] J. Li, Assessing the accuracy of predictive models for numerical data: not r nor r2, why not? Then what? PLOS ONE 12 (2017) e0183250 https://doi.org/10.1371/journal.pone.0183250.
[52] A. Botchkarev, A new typology design of performance metrics to measure errors in machine learning regression algorithms, Inter. J. Inf. Knowl. Manag 14 (2019) 045–076.
[53] S. Nakagawa, H. Schielzeth, A general and simple method for obtaining R2 from generalized linear mixed-effects models, Methods Ecol. Evol. 4 (2013) 133–142, https://doi.org/10.1111/j.2041-210x.2012.00261.x.
[54] D. Chicco, M.J. Warrens, G. Jurman, The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation, PeerJ Comput. Sci. 7 (2021) e623, https://doi.org/10.7717/peerj-cs.623.
[55] M.Z. Naser, A.H. Alavi, Error metrics and performance fitness indicators for artificial intelligence and machine learning in engineering and sciences, Arch. Struct. Constr. 3 (2023) 499–517, https://doi.org/10.1007/s44150-021-00015-8.
[56] Y.O. Ouma, R. Cheruyot, A.N. Wachera, Rainfall and runoff time-series trend analysis using LSTM recurrent neural network and wavelet neural network with satellite-based meteorological data: case study of Nzoia hydrologic basin, Complex Intell. Syst. 8 (2022) 213–236, https://doi.org/10.1007/s40747-021-00365-2.
[57] G.P. Zhang, D.M. Kline, Quarterly time-series forecasting with neural networks, IEEE Trans. Neural Netw. 18 (2007) 1800–1814, https://doi.org/10.1109/TNN.2007.896859.
[58] R. Sun, Optim. Deep Learn.: Theory Algorithms (2019), https://doi.org/10.48550/arXiv.1912.08957.
[59] C.M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, Berlin, Germany, 2006.
[60] S. Hanifi, A. Cammarono, H. Zare-Behtash, Advanced hyperparameter optimization of deep learning models for wind power prediction, Renew. Energy 221 (2024) 119700, https://doi.org/10.1016/j.renene.2023.119700.
[61] J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, S.-H. Deng, Hyperparameter optimization for machine learning models based on bayesian optimizationb, J. Electron Sci. Technol. 17 (2019) 26–40, https://doi.org/10.11989/JEST.1674-862X.80904120.
[62] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, et al., Generative adversarial networks, Commun. ACM 63 (2020) 139–144, https://doi.org/10.1145/3422622.
[63] J.V. Beck, K.J. Arnold, Parameter Estimation in Engineering and Science, James Beck, 1977.
[64] L.N. Smith, Cyclical Learning Rates for Training Neural Networks, 2017 IEEE Winter Conf. Appl. Comput. Vis. WACV (2017) 464–472, https://doi.org/10.1109/WACV.2017.58.
[65] M. Pirani, P. Thakkar, P. Jivrani, M.H. Bohara, D. Garg, A comparative analysis of ARIMA, GRU, LSTM and BiLSTM on financial time series forecasting, 2022 IEEE Int. Conf. Distrib. Comput. Electr. Circuits Electron. ICDCECE (2022) 1–6, https://doi.org/10.1109/ICDCECE53908.2022.9793213.
[66] M. Yang, J. Wang, Adaptability of financial time series prediction based on BiLSTM, Procedia Comput. Sci. 199 (2022) 18–25, https://doi.org/10.1016/j.procs.2022.01.003.
[67] A. Stepchenko, J. Chizhov, L. Aleksejeva, J. Tolujew, Nonlinear, non-stationary and seasonal time series forecasting using different methods coupled with data preprocessing, Procedia Comput. Sci. 104 (2017) 578–585, https://doi.org/10.1016/j.procs.2017.01.175.
[68] Züfle M., Kounev S. A Framework for Time Series Preprocessing and History-based Forecasting Method Recommendation, 2020, p. 141–144. https://doi.org/10.15439/2020F101.
[69] I. Pisa, I. Santin, J.L. Vicario, A. Morell, R. Vilanova, Data Preprocessing for ANN-based Industrial Time-Series Forecasting with Imbalanced Data. 2019 27th Eur. Signal Process. Conf. EUSIPCO, A Coruna, IEEE, Spain, 2019, pp. 1–5, https://doi.org/10.23919/EUSIPCO.2019.8902682.
[70] September M.A.K., Passino F.S., Goldmann L., Hinel A. Extended Deep Adaptive Input Normalization for Preprocessing Time Series Data for Neural Networks 2024. https://doi.org/10.48550/arXiv.2310.14720.