Neural Computing and Applications (2021) 33:14021–14035
https://doi.org/10.1007/s00521-021-06043-1

ORIGINAL ARTICLE

Smoothing and stationarity enforcement framework for deep learning time-series forecasting

Ioannis E. Livieris[1] · Stavros Stavroyiannis[2] · Lazaros Iliadis[3] · Panagiotis Pintelas[1]

Received: 9 November 2020 / Accepted: 13 April 2021 / Published online: 5 May 2021

Abstract
Time-series analysis and forecasting problems are generally considered as some of the most challenging and complicated problems in data mining. In this work, we propose a new complete framework for enhancing deep learning time-series models, which is based on a data preprocessing methodology. The proposed framework focuses on conducting a sequence of transformations on the original low-quality time-series data for generating high-quality time-series data, "*suitable*" for efficiently training and fitting a deep learning model. These transformations are performed in two successive stages: The first stage is based on the smoothing technique for the development of a new de-noised version of the original series in which every value contains dynamic knowledge of the all previous values. The second stage of transformations is performed on the smoothed series and it is based on differencing the series in order to be stationary and be considerably easier fitted and analyzed by a deep learning model. A number of experiments were performed utilizing time-series datasets from the cryptocurrency market, energy sector and financial stock market application domains on both regression and classification problems. The comprehensive numerical experiments and statistical analysis provide empirical evidence that the proposed framework considerably improves the forecasting performance of a deep learning model.

Keywords Deep Learning · Time-series · Forecasting · Stationarity

1 Introduction

Time-series forecasting problems are considered as one of the most challenging real-world prediction problems due to the large number of unpredictable factors involved, resulting in complicated temporal dependencies [1, 2]. In general, time series are encountered in a variety of popular real-world applications ranging from sales transactions [3, 4] and commodities [5, 6] to econometrics [7, 8] and finance [9, 10]. Time series data is a series of discrete data points, obtained at successive predefined equally spaced points in time. Their characteristics and unique properties considerably distinguish them from other types of data. The major difference between time-series data and other types of data is that their features need to be invariant to transition of time.

Stationarity is an important property and issue in time series. However, many real-world time series are non-stationary, which implies that significant properties such as mean, frequency, variance and kurtosis change over time. As a result, these series possess high volatility, trend and are frequently characterized by heteroskedasticity. The significance of the stationarity property for the efficiency of a deep learning time-series model was theoretically and computationally highlighted in [11]. Additionally, to handle the high volatility and significant fluctuations in the time-series data, some researchers considered to "smooth"

✉ Ioannis E. Livieris
livieris@upatras.gr

Stavros Stavroyiannis
computmath@gmail.com

Lazaros Iliadis
liliadis@civil.duth.gr

Panagiotis Pintelas
pintelas@math.upatras.gr

[1] Department of Mathematics, University of Patras, Patras 265-00, Greece

[2] Department of Accounting and Finance, University of the Peloponnese, Antikalamos 241-00, Greece

[3] Department of Civil Engineering, Democritus University of Thrace, Xanthi 671-00, Greece

the series [12]. Summarizing, time-series data are characterized by high complexity, complicated temporal dependencies, high fluctuations in prices and usually by chaotic behavior which makes the process of analyzing and forecasting a rather difficult task. Thus, time-series analysis and forecasting constitutes an active research area, focusing on the prediction of future values and directional movements.

During the last decades, a variety of forecasting models were proposed which were based on traditional time-series forecasting methods such as autoregressive moving average (ARMA) and autoregressive integrated moving average (ARIMA) models and their variations [13, 14] or on the more sophisticated machine learning methods [15–17]. However, the former are usually not able to deal with complex time-series patterns, while the latter cannot usually depict their high volatility and chaotic nature. Therefore, the need for the development of new more elaborated approaches and methodologies was considered essential for successfully addressing time-series forecasting problems.

Recently, the significant improvements in computational capabilities and storage, assisted researchers to deal and handle large and enormous amounts of information. Furthermore, the development of sophisticated and powerful data mining algorithms such as deep learning algorithms, supported the process of identifying and analyzing complex data. These recent advances of artificial intelligence and deep learning renew the interest of scientific and industrial community for the development of expert systems based on strong prediction models as well as the development of efficient preprocessing techniques to develop high-quality data from the original low-quality data. Although deep learning models possess the ability of being able to fit highly nonlinear models, they were proved to produce inefficient forecasts and significant autocorrelation in the errors [6, 18]. Recently, Livieris et al. [11] proposed a novel methodology which ensures the "suitability" of a time series for fitting a deep learning model by performing a series transformations in order to satisfy the stationarity property while simultaneously can be applied for regression and classification time-series tasks in a unified manner.

In this work, we propose a new preprocessing framework for enhancing the performance of deep learning time-series models, which constitutes the main contribution of this research. The proposed framework aims on transforming the original low-quality time-series data to high-quality time-series data, which are "*suitable*" for efficiently fitting a deep learning forecasting model. The framework consists of two transformation processes applied on the original time series. The first transformation is used for the development of a new unique series, based on the smoothing technique in which every price contains dynamic knowledge of the all previous prices of the original value. It is worth noticing that the new smoothed series constitutes a de-noised version of the original series; thus, it is easier for fitting a deep learning model and be analyzed. To the best of our knowledge, such transformation has not been applied in the literature. The second transformation is performed on the smoothed series and it is based on differencing the series in order to enforce stationarity and stabilize its mean and variance. The stationarity property is ensured and imposed by the application of Augmented Dickey–Fuller test [2]. We conducted a number of numerical experiments utilizing data from the popular real-world application domains of cryptocurrency market, energy sector and financial stock market on both regression (price prediction) and classification (directional movement prediction) problems. The presented comprehensive analysis illustrated that the proposed framework considerably improved the forecasting performance of a deep learning model, especially in terms of classification accuracy.

The remainder of this paper is organized as follows: Section 2 presents a brief review of deep learning models for time-series forecasting. Section 3 presents the proposed framework focusing on highlighting its theoretical advantages and benefits. Section 4 presents data preparation and reports the descriptive statistics describing the basic features of each dataset. Section 5 presents the detailed experimental analysis, focusing on the evaluation of the proposed framework. Section 6 summarizes the findings of this research and discusses the numerical experiments. Finally, Sect. 7 presents our conclusions and some interesting future directions.

2 Literature review

Time-series analysis and forecasting constitutes a famous complicated problem in data mining and a challenging research area which goes from traditional historical series prediction to social media analysis. Its complexity is caused by the time-series' volatility, significant fluctuations and internal structure which is highly influenced by a number of factors. Even though some variations exist, the major focus of scientific and industrial communities is the prediction of price and directional movement of the underlying asset. In spite of the diversity of time-series application, in this research we focus on three application domains which incorporate most of global research interest, that is cryptocurrency area, energy sector and financial stock market.

Chowdhury et al. [19] attempted to apply advanced machine learning methodologies on the index and constituents of cryptocurrency, aiming at forecasting future

prices. More specifically, they focused on the prediction of the closing price of the CCI30 index as well as nine major cryptocurrencies for assisting cryptocurrency investors in trading. In their research, they utilized a variety of machine learning algorithms including gradient boosted trees, ANNs, *k*-nearest neighbor and ensemble learning methods, and their utilized data contained daily closing prices from January 2017 to January 2019. gradient boosted trees and ensemble models reported the best performance, which was competitive and sometimes better, compared to that of similar state of the art models proposed in the literature.

Cen and Wang [20] proposed a new neural network architecture, called CID-STNN, for forecasting energy market. The proposed model is based on a stochastic time strength neural network which is equipped with a novel learning rate, controlled by the complexity invariant distance (CID) The data used in their study regard West Texas Intermediate oil daily prices from January 31, 2005 to December 5, 2016 and Brent oil daily prices from March 02, 2007 to December 16, 2016. Additionally, the time series were decomposed into different fluctuation frequency levels using the ensemble empirical mode decomposition, in order to fit the training models. The performed empirical research was conducted by evaluating the predicting ability of the CID-STNN model against that of other state-of-the art models, which revealed the superiority of proposed model.

Xi et al. [21] proposed a novel approach for forecasting crude oil price based on online media text mining. Their primary aim was to capture the more immediate market antecedents of price fluctuations and improve portfolio management. A CNN-based neural network model was utilized for extracting significant relationships of the price change fluctuations, which were then processed by the latent Dirichlet allocation topic model for distinguishing effects from various news topics. Furthermore, the input variable combination was optimized using feature selection methods and lag-order selection. Their utilized data contained headlines from the "Crude oil news" section of the portal https://www.investing.com/ as well as historical crude oil and financial market prices over the period from September 15, 2009 to July 20, 2014. Their numerical experiments demonstrated that the proposed topic-sentiment forecasting model outperformed traditional benchmark models and that the combination of text and financial features could improve the forecasting accuracy.

Fabbri and Moro [22] focused on detecting long-term dependencies in Dow Jones Industrial Average prices (DJIA) by utilizing deep recurrent neural networks (RNNs). More specifically, they presented a complete economical profit framework based on the predictions of a long short-term memory (LSTM) network. The data used in their study contained daily prices covering a period of DJIA historical prices between January 01, 2000 and December 31, 2017, concerning open price, high price, low price and adjusted close price for every working day. Their experimental results showed that the proposed framework outperformed feed-forward neural networks, obtaining a profit of more than 5 times the initial capital. Additionally, based on their experimental analysis, they stated that their proposed framework could be extended for the investigation of possible correlations among various market indexes.

Site et al. [23] evaluated the performance of a number of deep learning and machine learning models for forecasting the fluctuations in stock market. In more detail, they evaluated the prediction performance of LSTM networks, gated recurrent unit-based networks, recurrent neural networks, support vector regression, linear regression and ridge regression. The authors used data from a long consecutive period of 10 years (January 03, 2006 to December 29, 2017) containing weekly and monthly prices from two different indexes, namely Dow Jones Industrial Average and Standard & Poor's 500. Their experimental results showed that LSTM networks exhibited satisfactory performance on both different stock exchange data, outperforming the rest prediction models.

One common weakness of all presented and discussed approaches as well as most proposed in the literature is that they focused on obtaining better performance by exploiting more sophisticated models and deep learning techniques, usually ignoring the development of a sophisticated data preprocessing procedure. It is also worth mentioning that the powerful deep learning models were proved to provide unreliable predictions, especially when dealing with time series which follow an almost random walk process [18]. In this research, we propose a different approach and present a new complete framework for the development of accurate and reliable forecasting models. The novelty of the proposed framework is that the deep learning prediction model is fitted with a transformed series which is smoother than the original series resulting in a easier and faster training. Additionally, the new series is ensured to be stationary which guarantees the development of an accurate forecasting model which exhibits reliable forecasts, in terms of presenting no-autocorrelation in the errors.

3 Smoothing and stationarity framework for time-series data

Suppose that $y_0, y_1, \ldots, y_n$ be the values of a time series. A nonlinear regression model of order m is defined by

$$y_t = f(x_t, \theta), \tag{1}$$

where $x_t = (y_{t-1}, y_{t-2}, \ldots, y_{t-m}) \in \mathbb{R}^m$ consists of m values of y_t and θ is the parameter vector. After the model

structure has been defined, function $f(\cdot)$ can be determined by a machine learning or a deep learning algorithm.

In the sequel, we present the proposed preprocessing process and techniques for time-series forecasting, used in this research. Our methodology focuses on the transformation of the original low-quality time-series data to high-quality time-series data which are suitable for efficiently fitting and training a deep learning model. Notice that each step of our framework is successful and applies on the previous one to incrementally improve the forecasting performance.

3.1 Smoothing a time series

We define a new time series $\{s_t\}$, which consists of a "*smooth*" version of $\{y_t\}$, by

$$s_t = \begin{cases} y_0 & \text{if } t = 0; \\ \alpha y_t + (1-\alpha)s_{t-1}, & \text{otherwise}, \end{cases} \tag{2}$$

where $\alpha \in (0,1)$ is the smoothing factor or smoothing coefficient and the term $(1-\alpha)$ is called dumping factor. It is worth mentioning, that the smoothed value s_t consists of the weighted average of past observations while the weights decaying exponentially as the observations get older [24, 25]. In other words, higher weights are associated with the most recent observations. More specifically, by applying direct substitution of (2) back into itself, we obtain

$$\begin{aligned} s_t =& \alpha y_t + (1-\alpha)s_{t-1} \\ =& \alpha y_t + \alpha(1-\alpha)y_{t-1} + (1-\alpha)^2 s_{t-2} \\ =& \alpha y_t + \alpha(1-\alpha)y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + (1-\alpha)^3 s_{t-3} \\ & \vdots \\ =& \alpha y_t + \alpha(1-\alpha)y_{t-1} + \ldots + \alpha(1-\alpha)^t y_1 + (1-\alpha)^{t+1} y_0, \end{aligned}$$

which implies that as time passes the calculated smoothed value s_t consists of the weighted average of a number of past values and these assigned weights are in general proportional to the term of the following geometric progression

$$\alpha, \alpha(1-\alpha), \alpha(1-\alpha)^2, \ldots,$$

Notice that the value of parameter α controls the trade-off between the influence of each point y_t and the average of the most recent past observations s_t. In more detail, a large value of parameter α denotes that s_t pays more attention to the most recent observation while a small value denotes that the observation y_t has less influence to the smoothed value s_t. Moreover, s_t approximately averages over a number of $1/\alpha$ past observations. For example, in case $a = 0.1, 0.2$ or 0.5 implies that s_t approximately averages 2, 5 or 10 observations, respectively. Figure 1 presents daily Crypto-Cryptocurrency Index (CCI30) price trend from December 1, 2019 to December 31, 2019 and the corresponding smoothed series for $a = 0.1(0.1)0.9$.

The main objective of the application of the exponential smoothing technique is twofold: (i) the development of a new time series in which every price contains dynamic knowledge of the all previous prices. (ii) This new smoothed time series constitutes a de-noised version of the

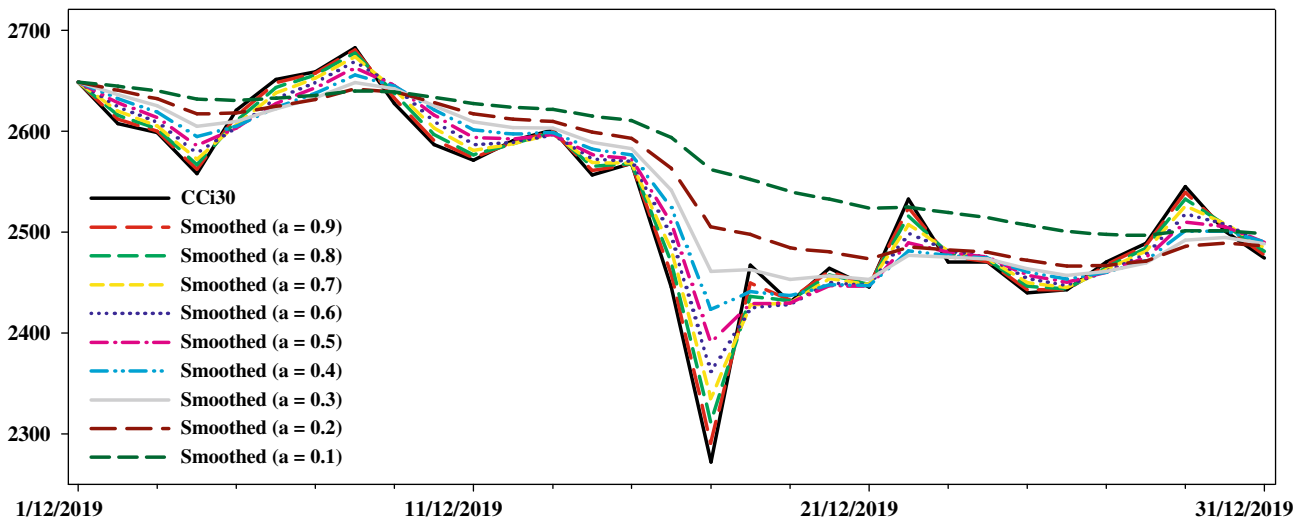

Fig. 1 Daily CCI30 price trend and the corresponding smoothed series for $a = 0.1(0.1)0.9$

original series; thus, it is easier for fitting a deep learning model.

Notice that after a deep learning model is trained with the smoothed series' any prediction $\hat{s}_t$ can be utilized for calculating the prediction for the levels of the original time series $\hat{y}_t$ by simply applying the inverse transformation, defined by

$$\hat{y}_t = \frac{1}{\alpha}(\hat{s}_t + (\alpha - 1)s_{t-1}).$$

3.2 Stationarity and model reliability

Although deep learning models constitute an efficient and widely utilized choice for forecasting time-series prices, they were proved to develop unreliable forecasts [11, 18]. The reason for this inefficiency is based on some undesired properties of time series such as noise, high volatility but mainly to the lack of stationarity of the time series. To address this problem, Livieris et al. [11] introduced a novel framework, which takes into account the nature of the time series and iteratively applies a series transformations and guarantees that the new transformed series is stationary and it is "suitable" for fitting a deep learning model. More specifically, the applied transformations are based on the first differences

$$\Delta y_t = y_t - y_{t-1}, \tag{3}$$

or the returns

$$r_t = \frac{y_t - y_{t-1}}{y_{t-1}}, \tag{4}$$

of the series. It is worth mentioning that the application of transformations (3) and (4) secures the efficiency and effectiveness of the model since its able to better capture possible nonlinearities, explain the data much better and successfully remove the autocorrelation in the errors [11].

In our proposed framework, we utilized the first differences transformation instead of the transformations based on the returns, which reported slightly better overall performance.

3.3 Proposed framework

In the sequel, we present our proposed framework which consists of two phases: *Smoothing*, and *Differencing-Training*.

In the "Smoothing phase," the data are imported and the smoothing technique is applied (Steps 1–3). In the "Training phase," the levels of the smoothed series are examined if they possess a unit root or if the series is stationary (Step 4). This is performed by the application of the Augmented Dickey–Fuller (ADF) test [2, 25]. In case, the smoothed series possess a unit root, the first differences transformation is iteratively applied, until the new developed transformed series is stationary (Steps 6–9). Subsequently, the new transformed time series is used for fitting and training the prediction model (Step 10). Notice that since the series is differenced, the errors of the forecasting

Table 1 Smoothing and stationarity framework to enhance deep learning in time-series forecasting

	/* Phase I: Smoothing */
Step 1.	Import time-series training data.
Step 2.	Select value of smoothing parameter α.
Step 3.	Apply smoothing on time-series data using (2).
	/* Phase II: Differencing-Training */
Step 4.	Apply the ADF unit root test.
Step 5.	**If** (*Time series possess a unit root*) **then** /* *Time series is not I*(0) */
Step 6.	**repeat**
Step 7.	Apply the transformation based on first differences.
Step 8.	Apply the ADF unit root test.
Step 9.	**until** (*Time series is stationary*)
Step 10.	Train the deep learning model DL using the transformed time series.
Step 11.	**else** /* *Time series is I*(0) */
Step 12.	Train the deep learning model DL using the smoothed time series.
Step 13.	Calculate the model's predictions on the smoothed training data.
Step 14.	Calculate the residuals between the smoothed training data and the model's predictions.
Step 15.	**If** (*Residuals possess autocorrelation*) **then**
Step 16.	Apply the transformation based on first differences.
Step 17.	Re-train the deep learning model DL using the transformed time series.
Step 18.	**end if**
Step 19.	**end if**

Table 2 Descriptive statistics for CCI30, Brent and DJIA datasets

Data	Minimum	Maximum	Mean	Std. Dev.	Median	Skewness	Kurtosis
CCI30							
Training set	276.35	20796.64	4316.56	3332.19	3548.02	1.90	4.35
Testing set	1938.49	4760.54	3307.14	738.22	3427.91	−0.10	−1.05
Brent							
Training set	26.01	118.90	70.69	24.55	63.57	0.53	−1.06
Testing set	14.85	70.25	57.66	12.12	61.15	−1.97	3.03
DJIA							
Training set	13328.85	27359.16	19721.59	3899.00	18041.55	0.47	−1.19
Testing set	18591.93	29551.42	27064.85	2261.92	27576.29	−1.90	3.40

model exhibit no autocorrelation [25]. In contrast, in case the series does not possess a unit root, then the levels of the smoothed series are used for training the model (Step 12). Then, the residuals are examined for exhibiting autocorrelation (Step 13). In case autocorrelation exists, the first-differences transformation is applied on the series and the model is re-trained utilizing the transformed series (Steps 15–18). Notice that if the levels of the smoothed series do not possess a unit root and the model's error exhibit no-autocorrelation there is no need to transform the series and update the training model, since over-differencing the time series could lead to a "non-invertible" process [11]. It is worth mentioning that the additional computational cost of the proposed framework is negligible (Table 1).

Finally, after the deep learning model is trained with the new smoothed transformed series, the inverse transformations are applied for calculating the prediction for the levels of the original time series $\hat{y}_t$.

4 Data

The datasets used in this research concern daily historical data from three widely utilized real-world application domains: cryptocurrency, commodity and finance.

From cryptocurrency domain, we utilized data from January 1, 2017 to March 31, 2020 of Crypto-Cryptocurrency Index (CCI30) which contains the top 30 cryptocurrencies weighted by market capitalization. The data were obtained from https://cci30.com/ website and were divided into training set consisting of daily values from January 1, 2017 to December 31, 2019 (1095 points) and a testing set consisting of daily prices from January 1, 2020 to March 31, 2020 (103 points). From commodity domain, we utilized data from January 1, 2016 to March 31, 2020 of Brent prices in USD from https://www.eia.gov/ website. The data were divided into training set consisting of daily Brent prices from January 1, 2016 to December 31, 2019 (1702 points) and a testing set consisting of daily prices from January 1, 2020 to March 31, 2020 (150 points). From finance domain, we utilized data from January 1, 2016 to March 31, 2020 of Dow Jones Industrial Average (DJIA) prices in USD from http://finance.yahoo.com website. The data were divided into training set consisting of daily prices from January 1, 2016 to December 31, 2019 (1702 points) and a testing set consisting of daily prices from January 1, 2020 to March 31, 2020 (150 points). It is worth noticing that all utilized datasets contained values that include the recent COVID-19 crisis in the beginning of 2020, which are characterized by considerable volatility and deviations from the regular behavior as well as structural breaks.

Table 2 reports the descriptive statistics for presenting the nature of the distribution, regarding the training set and the testing set of DJIA, Brent and CCI30 datasets, including the measures: minimum, maximum, mean, standard deviation (Std. dev.), median, skewness and kurtosis. Additionally, Table 3 summarizes the up and down movements in the prices as well as the corresponding percentages for all datasets.

Finally, it is worth mentioning that all time-series data contained no missing values, while the outlier prices were not removed in order not to destroy the dynamics of each series, even if these prices are the result of exceptional events.

5 Experimental methodology

In this section, we present a detailed experimental analysis, regarding the efficiency of our proposed framework which consists of three distinct phases: In the first phase, we apply our proposed framework to the CCI30, Brent and DJIA time series; in the second phase, we compare the performance of an efficient deep learning model trained with the traditional series and with the proposed smoothed series. Notice that our interest in the experimental analysis is mainly focused on the identification of the best value of parameter α; hence, we tested values of α ranging from 0 to

Table 3 The number of up and down movements of CCI30, Brent and DJIA datasets

Data	Up		Down	
CCI30				
Training set	606	55.39%	488	44.61%
Testing set	55	53.92%	47	46.08%
Brent				
Training set	850	49.97%	851	50.03%
Testing set	68	45.64%	81	54.36%
DJIA				
Training set	925	54.35%	777	45.65%
Testing set	81	54.36%	68	45.64%

1 in steps of 0.1. Finally, in the third phase, we perform a detailed statistical analysis to examine if there exists any statistical differences between the performance of the utilized prediction model trained with the traditional series and the proposed smoothed series.

5.1 Application of the proposed framework

In the sequel, we apply our proposed forecasting framework to CCI30, Brent and DJIA time-series data. Firstly, we examine the acceptance of the null hypothesis H_0: "*the series possess a unit root and are non-stationary*," using the ADF unit root test.

Table 4 reports the t-statistics and the associated p values of the ADF unit root test, performed on the level (Levels) of the CCI30, Brent and DJIA series as well as on the corresponding smoothed (Smoothed) series for every value of parameter α. Notice that each smoothed series for $\alpha = 1$ is identical with the original series. The interpretation of Table 4 reveals that the null hypothesis H_0 is accepted which implies that all series under consideration are non-stationary.

Next, according to our proposed framework, we difference all series under consideration and perform the ADF test to the transformed time series to examine if the unit root has been removed. Table 5 presents the results of the ADF unit root test for the training data of all transformed time series. Notice that (*) denotes statistical significance at the 5% critical level. As expected, all p values are practically zero which implies that the null hypothesis H_0 is rejected and all transformed series are indeed stationary. It is worth mentioning that the process of differencing the series solved the unit root problem and the series are "suitable" for fitting a deep learning model.

5.2 Numerical experiments

In this section, we evaluate the regression and classification performance of our proposed framework for forecasting time series. More specifically, we compare the performance of an efficient deep learning forecasting model trained with the first-differenced series and the Smoothed$_{FD}$ series for every selected value of parameter α. The deep learning model utilized in our study was CNN-LSTM which consists of two convolutional layers of 16 and 32 filters of size (2;) with the same padding, followed by a max pooling layer of size 2, a LSTM layer of 50 units and an output layer of one neuron. This model was selected under exhaustive experimentation (using different numbers and combinations of convolutional, LSTM and dense layers as well as different number of filters, units and neurons in these layers, respectively) and reported the best overall results for all series. The selected training algorithm was adaptive moment estimation [26] with a batch size equal to 64, utilizing mean-squared loss function. Furthermore, the CNN-LSTM model was trained with the first-differenced

Table 4 ADF unit root test of all time series under consideration

Time series	CCI30		Brent		DJIA	
	t stat	p value	t stat	p value	t stat	p value
Levels	−2.217	0.200	−1.695	0.434	−0.784	0.824
Smoothed ($\alpha = 0.9$)	−2.256	0.187	−1.712	0.425	−0.768	0.828
Smoothed ($\alpha = 0.8$)	−2.297	0.173	−1.713	0.424	−0.763	0.830
Smoothed ($\alpha = 0.7$)	−2.339	0.160	−1.715	0.423	−0.759	0.831
Smoothed ($\alpha = 0.6$)	−2.380	0.147	−1.717	0.422	−0.753	0.832
Smoothed ($\alpha = 0.5$)	−2.230	0.195	−1.719	0.421	−0.747	0.834
Smoothed ($\alpha = 0.4$)	−2.217	0.200	−1.724	0.419	−0.738	0.837
Smoothed ($\alpha = 0.3$)	−2.210	0.203	−1.731	0.415	−0.723	0.841
Smoothed ($\alpha = 0.2$)	−2.245	0.190	−1.746	0.408	−0.694	0.848
Smoothed ($\alpha = 0.1$)	−2.372	0.150	−1.788	0.387	−0.614	0.868

Table 5 ADF unit root test of all differenced time series

Time series	CCI30		Brent		DJIA	
	t stat	*p* value	*t* stat	*p* value	*t* stat	*p* value
First-differenced	−10.925	0.000*	−39.455	0.000*	−41.983	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	−10.563	0.000*	−35.698	0.000*	−38.122	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	−10.186	0.000*	−32.229	0.000*	−34.509	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	−9.791	0.000*	−28.963	0.000*	−31.085	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	−9.373	0.000*	−25.826	0.000*	−27.797	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	−9.445	0.000*	−22.739	0.000*	−24.593	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	−9.146	0.000*	−19.618	0.000*	−21.404	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	−8.661	0.000*	−16.357	0.000*	−18.137	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	−7.747	0.000*	−12.788	0.000*	−14.643	0.000*
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	−6.047	0.000*	−8.500	0.000*	−10.561	0.000*

series and the $\text{Smoothed}_{\text{FD}}$ using two different values of window size m, i.e. $m = 6$ and 12. Finally, the implementation was coded in Python 3.4 programming utilizing Keras library on a laptop (Intel(R) Core(TM) i7-6700HQ CPU 2.6 GHz and 16 GB RAM).

Regarding the utilized performance metrics in our research: mean absolute error (MAE), root mean square error (RMSE) and *R*-squared (R^2) were selected for evaluating the regression accuracy of the proposed model using all series under consideration. These metrics are respectively defined by

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2},$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \bar{y})^2}{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

where N is the number of forecasts, y_t is the actual value, $\bar{y}$ is the mean of the actual values and $\hat{y}_t$ is the predicted value. Additionally, accuracy (Acc), geometric mean (GM), sensitivity (Sen) and specificity (Spe), where utilized for evaluating the performance for the binary classification problem of directional movement (price increasement or decreasement on the following day with respect to the today's price). There metrics are defined by

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{FP}},$$

$$\text{GM} = \sqrt{\text{TP} \cdot \text{TN}},$$

$$\text{Spe} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

$$\text{Spe} = \frac{\text{TN}}{\text{TN} + \text{FP}},$$

where TP stands for the number of values which were correctly identified to be increased, TN stands for the number of values which were correctly identified to be decreased, FP (type I error) stands for the number of values which were misidentified to be increased and FN (type II error) stands for the number of values which misidentified to be decreased.

Furthermore, we included the performance metric area under curve (AUC) in our experimental analysis which is presented using the receiver operating characteristic (ROC) curve. Notice that ROC curve is created by plotting the true positive rate (sensitivity) against the false positive rate (specificity) at various threshold settings.

Finally, the reliability of the CNN-LSTM forecasting model was evaluated by examining the existence of autocorrelation in the errors, utilizing the autocorrelation function (ACF) plot and the Ljung-Box Q-test [2]. It is worth noticing that the reliability evaluation was originally proposed in [11, 18] to advocate the efficiency of its forecasts. Nevertheless, since the model exhibited no autocorrelation in the residuals, using any series as training data, we omitted the presentation of ACF plots and Ljung-Box Q-test.

5.2.1 CCI30 time series

Tables 6 and 7 present the performance comparison for CCI30 forecasting problem for $m = 6$ and $m = 12$, respectively. Regarding the regression performance for $m = 6$, the CNN-LSTM model reported the best RMSE score in case trained with $\text{Smoothed}_{\text{FD}}$ series with $\alpha = 0.5$ and $\alpha = 0.6$. For $m = 12$, the forecasting model trained with the $\text{Smoothed}_{\text{FD}}$ series reported slightly better RMSE score compared to that trained with first-differenced series for almost all values of parameter α. Moreover, it reported slightly better MAE score for $\alpha = 0.5, 0.6, 0.7$ and 0.9 in

Table 6 Performance comparison for CCI30 dataset ($m = 6$)

	MAE	RMSE	R^2	Acc (%)	AUC	GM	Sen	Spe
First-differenced	112.70	190.58	0.931	52.60	0.512	24.754	0.683	0.341
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	112.38	192.07	0.931	53.44	0.523	24.538	0.664	0.382
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	112.68	191.50	0.929	53.75	0.526	24.829	0.667	0.384
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	111.77	190.81	0.934	55.52	0.544	26.016	0.677	0.411
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	113.40	190.06	0.934	55.38	0.535	26.761	0.644	0.425
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	112.75	190.48	0.934	55.94	0.553	26.717	0.621	0.484
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	113.84	190.65	0.933	54.58	0.539	25.785	0.619	0.459
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	116.13	192.90	0.929	53.44	0.527	25.079	0.615	0.439
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	117.03	192.47	0.929	54.17	0.532	25.641	0.650	0.414
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	120.79	195.82	0.927	51.35	0.511	25.742	0.546	0.475

Table 7 Performance comparison for CCI30 dataset ($m = 12$)

	MAE	RMSE	R^2	Acc (%)	AUC	GM	Sen	Spe
First-differenced	121.85	197.56	0.919	50.42	0.491	25.123	0.646	0.336
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	118.51	196.86	0.919	51.88	0.508	25.060	0.642	0.373
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	122.25	200.21	0.919	52.50	0.516	25.925	0.619	0.414
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	121.81	196.20	0.924	53.98	0.512	26.381	0.606	0.418
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	120.27	195.20	0.928	54.27	0.537	26.713	0.612	0.461
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	120.96	195.70	0.927	54.06	0.538	25.923	0.575	0.500
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	122.29	195.19	0.924	52.71	0.523	25.875	0.575	0.491
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	123.66	195.67	0.924	52.92	0.523	24.773	0.594	0.452
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	123.22	196.36	0.921	54.79	0.546	25.249	0.567	0.525
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	124.81	194.35	0.922	52.60	0.525	24.871	0.542	0.507

the same situation. As regards R^2 metric, the prediction model reported the best performance in case trained with the $\text{Smoothed}_{\text{FD}}$ series for $\alpha = 0.5, 0.6, 0.7$, relative to both m values. Summarizing, we can easily conclude that CNN-LSTM model reported the best performance, trained with $\text{Smoothed}_{\text{FD}}$ series with $\alpha = 0.5, 0.6$ and 0.7. Additionally, its performance was slightly better compared to that, in case it was trained with the first-differenced series.

Regarding the classification performance, the forecasting model exhibited higher accuracy and AUC scores, in case trained with $\text{Smoothed}_{\text{FD}}$ series compared to that trained with the traditional first-differenced series. It is worth noticing that CNN-LSTM model exhibited the best classification performance when trained with $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.5, 0.6$ and 0.7, in terms of accuracy and AUC. Furthermore, it exhibited the best trade of between sensitivity and specificity as well as the highest GM score. This implies that the CNN-model trained with $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.5, 0.6$ and 0.7 are considerably less biased and performed more reliable forecasts compared to that trained with the first-differenced series, for both values of m.

5.2.2 Brent time series

Tables 8 summarizes the performance comparison for Brent forecasting problem for $m = 6$, regarding all utilized time series as training data. The CNN-LSTM model reported almost identical performance using any time series under consideration for training based on MAE and RMSE metrics. However, the forecasting model exhibited slightly better performance in case trained with the $\text{Smoothed}_{\text{FD}}$ series for $\alpha = 0.4 - 0.8$, relative to both m values. Regarding the classification performance, the forecasting model presented the best performance in case trained with $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.7$ in terms of accuracy and AUC. Additionally, CNN-LSTM presented the highest GM score in case trained with $\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$ and $\alpha = 0.7$) and the best trade-off between sensitivity and specificity, in case trained with $\text{Smoothed}_{\text{FD}}$

Table 8 Performance comparison for Brent dataset ($m = 6$)

	MAE	RMSE	R^2	Acc (%)	AUC	GM	Sen	Spe
First-differenced	1.19	1.80	0.974	54.59	0.538	38.795	0.435	0.640
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	1.19	1.80	0.974	54.53	0.537	38.383	0.428	0.645
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	1.19	1.78	0.976	54.73	0.542	38.126	0.472	0.611
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	1.20	1.79	0.976	56.22	0.556	39.499	0.479	0.633
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	1.20	1.78	0.977	55.41	0.543	38.884	0.449	0.634
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	1.20	1.79	0.976	54.93	0.543	38.408	0.466	0.630
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	1.20	1.79	0.976	55.68	0.550	37.361	0.468	0.633
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	1.23	1.82	0.975	54.26	0.533	37.149	0.410	0.655
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	1.25	1.87	0.975	51.35	0.507	35.418	0.431	0.584
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	1.26	1.91	0.975	51.42	0.506	35.302	0.402	0.610

($\alpha = 0.5$ and $\alpha = 0.7$). Summarizing, the utilization of $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.5$, 0.6 and 0.7, considerably improved the classification performance of the forecasting model while its regression performance was slightly improved.

Tables 9 summarize the performance comparison of the forecasting model for Brent forecasting problem for $m = 12$. The MAE and RMSE scores were improved in case the model was trained with $\text{Smoothed}_{\text{FD}}$ series instead of the traditional first-differenced series. Additionally, it exhibited the best regression performance for $\alpha = 0.3$. The interpretation of Table 9 presents that the classification performance of CNN-LSTM model exhibited the best performance in case trained with $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.7$ in terms of accuracy and AUC. Additionally, the best trade-off between specificity and sensitivity was reported in case the forecasting model was trained with $\text{Smoothed}_{\text{FD}}$ series with $\alpha = 0.7$ as well as the best GM score. Conclusively, we point out that with the utilization of $\text{Smoothed}_{\text{FD}}$ series ($\alpha = 0.7$) as training data, the forecasting model reported the best overall classification performance and developed more reliable forecasts.

5.2.3 Dow Jones industrial average time series

Tables 10 and 11 present the performance evaluation for DJIA forecasting problem for $m = 6$ and $m = 12$, respectively. Regarding the regression performance for $m = 6$, the CNN-LSTM model improved its MAE, RMSE and R^2 scores in case trained with $\text{Smoothed}_{\text{FD}}$ series instead of first-differenced series, for all values of parameter α. Additionally, for $m = 12$, the forecasting model trained with $\text{Smoothed}_{\text{FD}}$ series with $\alpha = 0.5 - 0.8$ reported slightly better MAE, RMSE and R^2 scores compared to that trained with first-differenced series. It is worth noticing that the CNN-LSTM model reported the best overall regression performance, trained with $\text{Smoothed}_{\text{FD}}$ series with $\alpha = 0.5, 0.6$ and 0.7; however, its performance was slightly better compared to that in case trained with the traditional first-differenced series.

As regards the classification performance, the forecasting model exhibited higher accuracy, AUC and GM scores, in case trained with $\text{Smoothed}_{\text{FD}}$ series compared to that trained with the traditional first-differenced series. More specifically, CNN-LSTM model exhibited the best

Table 9 Performance comparison for Brent dataset ($m = 12$)

	MAE	RMSE	R^2	Acc (%)	AUC	GM	Sen	Spe
First-differenced	1.36	2.03	0.971	53.04	0.526	38.470	0.475	0.578
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	1.33	1.97	0.972	51.76	0.516	38.806	0.500	0.533
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	1.31	1.93	0.974	51.89	0.518	38.983	0.502	0.534
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	1.30	1.92	0.974	53.61	0.529	40.122	0.479	0.579
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	1.30	1.89	0.974	50.74	0.505	38.340	0.469	0.540
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	1.28	1.86	0.974	51.42	0.509	39.116	0.444	0.574
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	1.26	1.84	0.973	51.05	0.533	36.368	0.438	0.628
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	1.25	1.82	0.975	51.62	0.509	36.496	0.424	0.595
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	1.24	1.85	0.973	52.43	0.511	35.964	0.347	0.675
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	1.25	1.87	0.973	51.96	0.519	36.496	0.513	0.525

Table 10 Performance comparison for DJIA dataset ($m = 6$)

	MAE	RMSE	R^2	Acc (%)	AUC	GM	Sen	Spe
First-differenced	320.24	594.22	0.928	48.18	0.472	33.495	0.580	0.363
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	317.68	589.16	0.930	48.58	0.479	32.292	0.549	0.409
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	316.30	585.59	0.931	48.58	0.481	32.057	0.532	0.430
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	318.67	588.26	0.930	48.11	0.475	35.161	0.543	0.406
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	318.56	586.56	0.931	48.72	0.482	35.825	0.533	0.431
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	317.16	591.56	0.930	51.55	0.503	36.847	0.552	0.419
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	316.83	588.37	0.931	50.74	0.498	34.674	0.594	0.403
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	316.85	589.08	0.930	51.49	0.513	33.413	0.533	0.493
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	315.21	588.47	0.931	51.62	0.504	33.790	0.631	0.378
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	315.78	587.02	0.930	50.20	0.494	33.753	0.575	0.354

Table 11 Performance comparison for DJIA dataset ($m = 12$)

	MAE	RMSE	R^2	Acc (%)	AUC	GM	Sen	Spe
First-differenced	328.08	602.64	0.927	47.23	0.463	34.583	0.551	0.348
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.9$)	331.26	613.32	0.924	48.38	0.471	35.821	0.535	0.379
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.8$)	326.80	600.03	0.928	49.53	0.486	35.191	0.540	0.379
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	323.17	595.67	0.929	48.85	0.484	37.211	0.516	0.463
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	326.95	600.71	0.929	48.92	0.486	36.510	0.528	0.430
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	322.70	598.46	0.929	49.19	0.491	36.874	0.533	0.460
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.4$)	326.72	610.53	0.926	48.65	0.483	36.221	0.474	0.512
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.3$)	330.66	615.92	0.924	51.76	0.514	36.221	0.512	0.482
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.2$)	321.74	595.43	0.922	52.57	0.526	34.797	0.532	0.466
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.1$)	334.46	647.65	0.925	49.39	0.488	34.452	0.498	0.510

classification performance in case trained with $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.1 - 0.5$, in terms of accuracy and AUC. Moreover, the model reported higher GM score in case trained with $\text{Smoothed}_{\text{FD}}$ with $\alpha = 0.4 - 0.7$ compared to that trained with the first-differenced series. Finally, it is worth noticing that the forecasting model reported the best trade of between sensitivity and specificity as well as the best GM in case $\alpha = 0.5, 0.6$ and 0.7. This implies that the proposed preprocessing framework considerably improved the classification performance of the CNN-model for $\alpha = 0.5, 0.6$ and 0.7 since it was less biased and performed more reliable forecasts, for both values of m.

5.3 Statistical analysis

Next, we attempt to provide statistical evidences about the efficiency of our proposed preprocessing framework. More specifically, for rejecting the hypothesis H_0 that the forecasting model CNN-LSTM performed equally well for a given level using the first-differenced series or the $\text{Smoothed}_{\text{FD}}$ series as training data, we used the nonparametric Friedman Aligned Ranking (FAR) [27] test. Furthermore, for examining if the differences in the performance of the utilized prediction model are statistically significant, we applied the post hoc Finner test [28] with significance level $\alpha = 5\%$. Therefore, we perform a nonparametric multiple comparison, regarding the regression and classification performance of CNN-LSTM trained with the first-differenced series and the $\text{Smoothed}_{\text{FD}}$ series. Notice that three versions of the $\text{Smoothed}_{\text{FD}}$ series where selected i.e. $\alpha = 0.5, 0.6$ and 0.7, which reported the best overall regression and classification performance.

Tables 12, 13, 14, 15, 16 and 17 report the statistical analysis, performed by nonparametric multiple comparison, relative to MAE, RMSE, R^2, accuracy, AUC and GM performance metrics. Additionally, to measure the difference in the performance regarding the balance of sensitivity and specificity for each series, we used a new metric defined by the product of sensitivity and specificity, i.e. Sen$\times$Spe. CNN-LSTM performed equally well using any time series as training data regarding MAE metric; while the performance of the forecasting model was improved as

Table 12 FAR test and Finner post hoc test based on MAE metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	9.917	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	10	0.9837	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	12.417	0.6883	Accepted
First differenced	17.667	0.1632	Accepted

Table 13 FAR test and Finner post hoc test based on RMSE metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	9.5	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	10.583	0.8820	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	10.75	0.8820	Accepted
First differenced	19.167	0.0497	Rejected

Table 14 FAR test and Finner post hoc test based on R^2 metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	7.5	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	10.167	0.551661	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	10.833	0.551661	Accepted
First differenced	21.5	0.001814	Rejected

Table 15 FAR test and Finner post hoc test based on accuracy metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	9	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	9.5	0.9025	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	13	0.4481	Accepted
First differenced	18.5	0.0487	Rejected

confirmed by the statistical test, regarding RMSE and R^2 metrics. In contrast, the classification performance of CNN-LSTM was considerably improved in case it was trained with the $\text{Smoothed}_{\text{FD}}$ series instead of the first-differenced series as confirmed statistically by the FAR and Finner tests. It is worth mentioning that CNN-LSTM exhibited the best performance in case it was trained with $\text{Smoothed}_{\text{FD}}$ series ($\alpha = 0.5$), regarding the accuracy and AUC performance metrics (Table 18).

Table 16 FAR test and Finner post hoc test based on AUC metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	7.917	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	11	0.4501	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	12.25	0.3998	Accepted
First differenced	18.833	0.0223	Rejected

Table 17 FAR test and Finner post hoc test based on GM metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	8.833	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	10.167	0.8465	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	10.333	0.8465	Accepted
First differenced	20.667	0.0112	Rejected

Table 18 FAR test and Finner post hoc test based on Sen×Spe metric

Series	Friedman	Finner post hoc test	
	Ranking	p value	H_0
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.5$)	7.667	–	–
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.7$)	12.833	0.5676	Accepted
$\text{Smoothed}_{\text{FD}}$ ($\alpha = 0.6$)	10	0.2920	Accepted
First differenced	19.5	0.0112	Rejected

6 Discussion

In this section, we conduct a comprehensive discussion regarding the theoretical and experimental contribution of this research.

6.1 Discussion of the proposed methodology

Most real-world time-series data are considered as chaotic and noisy by nature and are often characterized by high volatility, high fluctuations in prices and large variations in the variance. This implies that the process of developing an accurate and reliable deep learning forecasting model is a challenging problem. Generally, deep learning networks constitute powerful time-series prediction models in terms of prediction accuracy, but are usually unstable and sometimes unreliable in sense that even small variations in their training data as well as noisy features could considerably affect their performance.

In this research, we proposed a complete theoretical framework regarding the problem of time-series

forecasting price and movement as well as the reliability of the prediction model's forecasts. The motivation of our approach rises from the necessity of transforming the original low-quality time-series data to high-quality time-series data, suitable for efficiently training and fitting a deep learning model. More specifically, the transformation based on the smoothing technique develops a new time series in which every price contains dynamic knowledge of the all previous prices and constitutes a de-noised version of the original series. As a result, this new smoothed time series can be easily fitted by a deep learning model. The advantage of the exponential smoothing over other smoothing techniques based on moving average is that every price contains information from all previous values and not from a subset of them, and provides significant importance to the most recent values. Additionally, the process of differencing the series makes the series stationary and transforms it to be suitable for training a deep learning model, which will provide reliable and stable prediction performances.

It is worth mentioning that our proposed framework performs an efficient preprocessing process, through a series of transformations based on the smoothing and the differencing techniques, for exploiting the internal representation of any times-series data. Therefore, it possesses the attractive property that it can be easily applied to a wide area of real-world time-series forecasting problems without the imposition of any additional modifications and extra constraints.

6.2 Discussion of the experimental results

In this work, we conducted a series of experiments using three time series from the application domains of cryptocurrency market, energy sector and financial stock market. To evaluate the efficiency and robustness of the proposed framework, we selected a state-of-the-art deep learning model, called CNN-LSTM. The performance of the utilized deep learning model trained with the differenced series and with the smoothed differenced series was evaluated on both forecasting time-series price (regression) and directional movement (classification) problems.

The presented numerical experiments demonstrated that the proposed framework can considerably improve and enhance the overall performance of the deep learning model. More specifically, the regression performance of CNN-LSTM trained with the smoothed differenced series was improved but not significantly, presenting lower MAE and RMSE scores and higher R^2 score. In contrast, its classification performance was considerably improved, which resulted that the smoothing technique property handled the noisy original data. Additionally, the performance metrics GM, Sen and Spe revealed that our framework developed deep learning models which were less biased on average than the ones trained with the first differenced series. These findings were confirmed statistically by the Friedman Aligned Ranks nonparametric test and the Finner post hoc test.

Nevertheless, the limitation of this work is that it is still not clear which is the optimal value of parameter α. In our preliminary experimental results, we observed that a value within the interval [0.5, 0.7] reported the best overall performance. However, we cannot draw safe conclusions and more experiments are needed, since this is not a general case. Therefore, its determination is considered a rather difficult task. At this point, it is worth mentioning that due to the sensitivity of various hyper-parameters as well as the high complexity of the deep learning models, it is possible that their prediction ability could be further improved by performing additional optimized configuration (see [29–31] and the references therein).

However, although we provide thorough experimental results revealing that the proposed methodology can enhance the performance of a deep learning model, there is no clear indication why trend prediction efforts, might not necessarily translate into profits. In other words, even though the presented experimental results are promising, we have no evidence that the proposed framework can actually develop a prediction model to assist cryptocurrency investors for making proper investment decisions based on our model predictions in order to achieve profitable investment returns. This is mainly based on the fact that time series from application domains such as cryptocurrency area, energy sector and financial market are highly affected by time evolution and external changes; therefore, an efficient prediction model may be temporally accurate but not in "depth of time" [11].

However, this research is dedicated to present a complete framework which enhances the forecasting performance and reliability of a deep learning model and less on the design and implementation of profitable trading system. It is worth noticing that the development of such software system would require the control of a number of aspects in addition to the implementation of a decision support system. Therefore, a possible improvement of our prediction framework could be the incorporation of trading simulations in order to identify potential profitable investment returns. This is to be certainly investigated in our future research.

Finally, it is worth mentioning that in certain times of global instability, outliers are frequently presented, especially in time series from the domains of cryptocurrency, commodity and finance. In this research, the outlier prices in all used series data were not removed in order not to destroy the dynamics of each series, even if these prices are

the result of exceptional events. Our proposed framework develops a smoothed series which constitutes a denoised version of the original series in which every value contains dynamic knowledge of all previous values. One issue which we should thoroughly investigate in the future is the possibility of enhancing our framework by the adoption of robust filtering techniques [32–35] in order to "deal" with outliers or other rare signals which could indicate forecasting instability.

7 Conclusions and future research

In this work, we proposed a new complete and novel framework which is based on a preprocessing process for enhancing deep learning time-series models. The novelty of the proposed framework focuses on getting high-quality time-series data, suitable for efficiently training and fitting a deep learning model, utilizing a transformation of the original low-quality time-series data. The transformation based on the smoothing technique which develops a new de-noised version of the original series in which every price contains dynamic knowledge of the all previous values. Moreover, the new series is differenced in order to be stationary and therefore suitable for training a deep learning model. A series of numerical experiments were conducted using three time-series datasets from the cryptocurrency market, energy sector and financial stock market application domains on both regression and classification problems. The detailed and comprehensive analysis demonstrated that the proposed framework considerably improved the forecasting performance of a deep learning model, especially in terms of classification accuracy.

Therefore, it can be utilized as a reference for developing efficient and reliable deep learning forecasting models Finally, a considerable advantage of the proposed methodology is that it can be easily utilized to cover a wider scientific area of time-series applications without any special requirements as well as adopted with any time-series forecasting model [36–39]. In our future work, we intend to evaluate how the proposed framework can enhance the performance of several forecasting models.

Nevertheless, one issue which we should thoroughly investigate in our future research is to evaluate other smoothing techniques based on moving average and double exponential smoothing. Another interesting direction for future research for the development of robust time-series models could be the incorporation of ensemble methodologies for combining deep learning models trained with different smoothed series relative to the value of parameter a together with base learners based on sophisticated architectures [40].

These could further improve both the regression and classification performance and develop more accurate and reliable forecasts. Furthermore, since our experiments are quite encouraging, a promising idea to enhance our propose framework with elegant and advanced techniques for detecting communities of connectivity in a time series [41].

Declarations

Conflict of interest The author declared no potential conflicts of interest with respect to the research, authorship and/or publication of this article.

References

1. Weigend AS (2018) Time series prediction: forecasting the future and understanding the past. Routledge, London
2. Brockwell Peter J, Brockwell Peter J, Davis Richard A, Davis Richard A (2016) Introduction to time series and forecasting. Springer, Berlin
3. Berry Lindsay R, Paul H, Mike W (2019) Probabilistic forecasting of heterogeneous consumer transaction-sales time series. Int J Forecast 36:552
4. Nunnari Giuseppe, Nunnari Valeria (2017) Forecasting monthly sales retail time series: a case study. In: 2017 IEEE 19th conference on business informatics (CBI), vol 1, pp 1–6. IEEE
5. Rosales Enrique Benavides (2017) Time-series and cross-sectional momentum and contrarian strategies within the commodity futures markets. Cogent Econ Financ 5(1):1339772
6. Livieris IE, Pintelas E, Kiriakidou N, Stavroyiannis S (2020) An advanced deep learning model for short-term forecasting U.S. natural gas price and movement. In: IFIP international conference on artificial intelligence applications and innovations. Springer, Berlin, pp 165–176
7. Walter E (2008) Applied econometric time series. Wiley, New York
8. Pesaran Hashem M (2015) Time series and panel data econometrics. Oxford University Press, Oxford
9. Box George EP, Jenkins Gwilym M, Reinsel Gregory C, Ljung Greta M (2015) Time series analysis: forecasting and control. Wiley, New York
10. Tsay Ruey S (2005) Analysis of financial time series, vol 543. Wiley, New York
11. Livieris Ioannis E, Stavros S, Emmanuel P, Panagiotis P (2020) A novel validation framework to enhance deep learning models in time-series forecasting. Neural Comput Appl 32:17149
12. Van Dick D, Timo T, Hans FP (2002) Smooth transition autoregressive models: a survey of recent developments. Econ Rev 21(1):1–47
13. Hassler Uwe (2016) Autoregressive moving average processes (ARMA). In: Stochastic Processes and calculus. Springer, Berlin, pp 45–75
14. Neusser K (2016) Autoregressive moving-average models. In: Time Series Econometrics. Springer, Berlin, pp 25–44
15. Chirag D, Fan Z, Junjing Y, Eang LS, Wei SK (2017) A review on time series forecasting techniques for building energy consumption. Renew Sustain Energy Rev 74:902–924
16. Trevor F, Christophe M (2016) An empirical comparison of classification algorithms for mortgage default prediction:

evidence from a distressed mortgage market. Eur J Oper Res 249(2):427–439
17. Livieris IE, Kotsilieris T, Stavroyiannis S, Pintelas P (2019) Forecasting stock price index movement using a constrained deep neural network training algorithm. Intell Dec Technol, (Preprint):1–11
18. Pintelas E, Livieris IE, Stavroyiannis S, Kotsilieris T, Pintelas P (2020) Investigating the problem of cryptocurrency price prediction—a deep learning approach. In: IFIP International conference on artificial intelligence applications and innovations. Springer, Berlin, pp 99–110
19. Chowdhury R, Rahman MA, Rahman MS, Mahdy MRC (2019) Predicting and forecasting the price of constituents and index of cryptocurrency using machine learning. arXiv preprint arXiv:1905.08444
20. Zhongpei C, Jun W (2018) Forecasting neural network model with novel CID learning rate and EEMD algorithms on energy market. Neurocomputing 317:168–178
21. Xuerong L, Wei S, Shouyang W (2019) Text-based crude oil price forecasting: A deep learning approach. Int J Forecast 35(4):1548–1560
22. Fabbri M, Moro G (2018) Dow Jones trading with deep learning: The unreasonable effectiveness of recurrent neural networks. In: Proceedings of the 7th international conference on data science, technology and applications, pp 142–153
23. Site A, Birant D, Işık Z Stock market forecasting using machine learning models. In: 2019 Innovations in intelligent systems and applications conference (ASYU), IEEE, pp 1–6
24. Hyndman RJ, Athanasopoulos G (2018) Forecasting: principles and practice
25. Montgomery Douglas C, Jennings Cheryl L, Murat Kulahci (2015) Introduction to time series analysis and forecasting. Wiley, New York
26. Kingma DP, Adam JB (2015) A method for stochastic optimization. In: 2015 International conference on learning representations
27. Hodges Joseph L, Lehmann Erich L (1962) Rank methods for combination of independent experiments in analysis of variance. Ann Math Stat 33(2):482–497
28. Helmut F (1993) On a monotonicity problem in step-down multiple test procedures. J Am Stat Assoc 88(423):920–923
29. Bergstra J, Bardenet R, Bengio Y, Kégl B (2011)Algorithms for hyper-parameter optimization. In: 25th annual conference on neural information processing systems (NIPS 2011), vol 24. Neural Information Processing Systems Foundation
30. James B, Yoshua B (2012) Random search for hyper-parameter optimization. J Machine Learn Res 13:2
31. Vladimir S, Novak N (2016) A nature inspired parameter tuning approach to cascade control for hydraulically driven parallel robot platform. J Optim Theory Appl 168(1):332–347
32. Ziran C, Baoyong Z, Vladimir S, Yijun Z, Zhengqiang Z (2020) Event-based fuzzy control for ts fuzzy networked systems with various data missing. Neurocomputing 417:322–332
33. Xuefei D, Shuping H, Vladimir S (2020) Robust fault detection filter design for a class of discrete-time conic-type non-linear markov jump systems with jump fault signals. IET Control Theory Appl 14(14):1912–1919
34. Vladimir S, Dragan P (2020) Robust identification for fault detection in the presence of non-gaussian noises: application to hydraulic servo drives. Nonlinear Dyn 100:2299–2313
35. Hongfeng T, Peng W, Yiyang C, Vladimir S, Huizhong Y (2020) An unsupervised fault diagnosis method for rolling bearing using STFT and generative neural networks. J Frank Inst 357(11):7286–7307
36. Izonin I, Tkachenko R, Vitynskyi P, Zub K, Tkachenko P, Dronyuk I (2020) Stacking-based GRNN-SGTM ensemble model for prediction tasks. In: 2020 International conference on decision aid sciences and application (DASA), pp 326–330. IEEE
37. Panagiotis M, Mattheou K, Alex K (2010) Forecasting ARMA models: a comparative study of information criteria focusing on MDIC. J Stat Comput Simulat 80(1):61–73
38. Roman T, Ivan I, Natalia K, Ivanna D, Khrystyna Z (2020) An approach towards increasing prediction accuracy for the recovery of missing IoT data based on the GRNN-SGTM ensemble. Sensors 20(9):2625
39. Vitynskyi P, Tkachenko R, Izonin I, Kutucu H (2018) Hybridization of the SGTM neural-like structure through inputs polynomial extension. In: 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), pp 386–391. IEEE
40. Tay Y, Tuan LA, Hui SC (2018) Multi-cast attention networks. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pp 2299–2308
41. Konstantinos D, Dimitrios T, Lykourgos M (2020) Modeling and forecasting the COVID-19 temporal spread in Greece: an exploratory approach based on complex network defined splines. Int J Environ Res Public Health 17:4693