Neural Computing and Applications
https://doi.org/10.1007/s00521-020-05169-y

S.I. : EMERGING APPLICATIONS OF DEEP LEARNING AND SPIKING ANN

A novel validation framework to enhance deep learning models in time-series forecasting

Ioannis E. Livieris[1] · Stavros Stavroyiannis[2] · Emmanuel Pintelas[1] · Panagiotis Pintelas[1]

Received: 15 April 2020 / Accepted: 24 June 2020

Abstract
Time-series analysis and forecasting is generally considered as one of the most challenging problems in data mining. During the last decade, powerful deep learning methodologies have been efficiently applied for time-series forecasting; however, they cannot guarantee the development of reliable prediction models. In this work, we introduce a novel framework for supporting deep learning in enhancing accurate, efficient and reliable time-series models. The major novelty of our proposed methodology is that it ensures a time-series to be "*suitable*" for fitting a deep learning model by performing a series of transformations in order to satisfy the stationarity property. The enforcement of stationarity is performed by the application of Augmented Dickey–Fuller test and transformations based on first differences or returns, without the loss of any embedded information. The reliability of the deep learning model's predictions is guaranteed by rejecting the hypothesis of autocorrelation in the model's errors, which is demonstrated by autocorrelation function plots and Ljung–Box Q test. Our numerical experiments were performed utilizing time-series from three real-world application domains (financial market, energy sector, cryptocurrency area), which incorporate most of global research interest. The performance of all forecasting models was compared on both problems of forecasting time-series price (regression) and time-series directional movements (classification). Additionally, the reliability of the models' forecasts was evaluated by examining the existence of autocorrelation in the errors. Our numerical experiments indicate that our proposed methodology considerably improves the forecasting performance of a deep learning model, in terms of efficiency, accuracy and reliability.

Keywords Time-series · Deep learning · Spatiotemporal data · Reliability

1 Introduction

Time-series are encountered in a large variety of real-world applications, ranging from finance [10, 18] and commodities [8, 20] to healthcare [9, 29] and pollution management [4, 14]. Time-series data consist of discrete data points, obtained at successive equally spaced points in time. The main properties and characteristics of time-series data are responsible for distinguishing them from other types of data. More specifically, they frequently contain much noise, exhibit high volatility as well as extremal directional movements and possess a tendency for possible reversing these movements in the near-term future. Due to these significant characteristics, time-series forecasting is generally considered as one of the most challenging problems in data mining. As a result, the analysis of time-series data

✉ Ioannis E. Livieris
livieris@upatras.gr

Stavros Stavroyiannis
s.stavroyiannis@teipel.gr

Emmanuel Pintelas
ece6835@upnet.gr

Panagiotis Pintelas
pintelas@math.upatras.gr

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

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

Journal : Large 521 | Dispatch : 7-7-2020 | Pages : 19
Article No. : 5169 | ☐ LE | ☐ TYPESET
MS Code : NCAA-D-20-01027R1 | ☑ CP | ☑ DISK

has been an active subject of research for decades [6, 7, 23, 30].

In the literature, the problem of time-series price and movement forecasting has been comprehensively studied for decades and numerous rewarding approaches have been proposed. Traditional time-series methods such as ARIMA (Auto-Regressive Integrated Moving Average) [5, 6] and its variations as well as the more elaborated Machine Learning methods [1, 3] probably constitute the most famous and widely utilized methods for time-series prediction. Nevertheless, these methods frequently do not possess the ability to accurately model such complex data and be successfully effective, since they cannot depict the stochastic nature and high volatility of time-series.

During the last decade, the rapid advances in artificial intelligence, as well as the vigorous developments in deep learning techniques, attracted wide attention of scientific and industrial communities for the development of efficient and robust time-series forecasting models. Probably the most popular and widely utilized deep learning methodology is the development of an ANN-type network utilizing convolutional and long short-term memory (LSTM) layers, along with the classical dense layers. The former are utilized to filter out the noise of the input data [11], and the latter are tailored to efficiently capture complex temporal dependencies and sequence pattern information by exploiting their special architecture design [2]. Along this line, researchers [18–20, 31] paid special attention to exploit the advantages and benefits of both mentioned deep learning techniques, proposing forecasting models utilizing both convolutional and LSTM layers.

Recently, Pintelas at al. [24, 25] evaluated the performance of several deep learning models for price and movement forecasting of major cryptocurrencies. The major novelty in their work was the application of a series of tests for examining the prediction efficiency but mostly the reliability of the models. In other words, they examined whether the models have properly fitted the time-series data and exploited all the available mined information, during the training process. Based on their experimental analysis, the authors stated that even the powerful deep learning methodologies cannot guarantee the development of reliable forecasting models. Additionally, they concluded that new more sophisticated algorithmic methods should be considered for the development of an accurate and reliable prediction model.

In this work, we propose a novel framework for the development of efficient and reliable deep learning models which constitutes the main contribution. The novelty of our proposed methodology is that it guarantees considerable improvement in the deep learning model's forecasting performance in terms of reliability and accuracy, regarding any utilized time-series. More analytically, the proposed methodology ensures the "*suitability*" of a time-series for fitting a deep learning model by performing a proper transformation, in order to satisfy the stationarity property and therefore no autocorrelation in the model's errors. The stationarity is secured by the application of Augmented Dickey–Fuller test and transformations based on first differences or returns. It is worth mentioning that an adequate deep learning model trained with the transformed series presents no autocorrelation in the errors and a big improvement of the forecasting performance is expected, compared with the same model trained with the original non-transformed series. We conducted a detailed and comprehensive experimental analysis on time-series from three real-world application domains which incorporate most of global research interest, that is, financial stock market, energy sector and the novel cryptocurrency area. All prediction models were evaluated on both problems of forecasting time-series price (regression) and also for the prediction of time-series directional movements (classification). Furthermore, the reliability of the models' forecasts was evaluated by examining the existence of autocorrelation of the errors using the autocorrelation function plot and the Ljung–Box Q test.

The remainder of this paper is organized as follows: Sect. 2 presents a brief review of state-of-the-art deep learning-based models for time-series forecasting. Section 3 presents a comprehensive description of the problem of reliability introduced in deep learning models for time-series forecasting. Section 4 presents our proposed framework, providing special attention to its theoretical advantages and benefits. Section 6 presents our experimental methodology including the data preparation and preprocessing as well as the detailed experimental analysis, regarding the evaluation of proposed methodology. Section 7 summarizes our findings and discusses the experimental results. Finally, Sect. 8 presents the conclusions and some future directions.

2 Related work

Time-series forecasting is generally considered as one of the most challenging and significantly complex research areas. The complexity of time-series' internal structure is caused by the variety of factors which have a deep influence on the series and on the volatility of these factors [5, 6, 30]. During the last years, the significant developments in computer science as well as rapid advances in research lead to the exponential generation of temporal and sequential data [13]. Therefore, time-series infiltrated almost every task and assignment, requiring a human cognitive process. Recently, Fawaz et al. [11] provided an excellent review, presenting a comprehensive overview of

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

the application of deep learning approaches in various time-series domains. More specifically, they presented in detail the process of mining time-series data using deep learning methods for discovering new insights, and how those insights impact the process of decision making. In the rest of our research, we focus on three time-series application domains which incorporate most of global research interest, that is, financial stock market, energy sector and the novel cryptocurrency area.

Liu et al. [18] developed a CNN–LSTM framework for modeling and analyzing stock markets' quantitative selection and timing strategy. The convolutional-based framework is used for determining the selection of the quantitative stock strategy and subsequently, the LSTM-based framework is utilized for performing the quantitative timing strategy in order to improve the amounts of profits. The stock time-series data used in their research range from January 1, 2007, to December 31, 2017. Their experiments demonstrated that their proposed CNN–LSTM framework could be efficiently applied for defining a quantitative strategy and achieving better profits than the classical Benchmark index.

Fischer and Krauss [12] focused on developing an efficient forecasting model based on deep learning techniques and unveil the sources of stock profitability. More specifically, the utilized LSTM networks for predicting the directional movements for Standard & Poor's 500 (S&P500) constituent stocks. The utilized data contained prices of S&P500 constituents from Thomson Reuters from December 1989 to September 2015. The experiments reported that LSTM exhibited a Sharpe Ratio of 5.8 prior to transaction costs and daily returns of 46% per day. Additionally, LSTM networks outperformed computationally efficient classification methods such as deep neural network, random forest and logistic regression. Finally, the authors developed a rule-based decision support making system which focused on selecting winning and losing stocks, exploiting the LSTM predictions.

Zhao et al. [32] proposed a deep learning ensemble forecasting model to address the problem of forecasting oil prices. Their proposed model is based on Bootstrapping aggregation (Bagging) ensemble strategy which exploits the predictions of advanced deep learning base models, called Stacked Denoising Auto-Encoders (SDAE). The data utilized in their study contained monthly prices covering a period from January 1986 to May 2016, concerning 198 exogenous factors such as cost of crude oil imports, refiner values of crude oil products, information on rigs and development wells drilled, oil product consumption, crude oil production as well as macroeconomic and financial indicators. Their experimental analysis showed the forecasting superiority of the proposed model, which was statistically proved by three nonparametric tests.

Cen and Wang [8] aimed at forecasting the volatility behaviors of crude oil prices for increasing the prediction accuracy of oil market price. The authors considered a methodology based on a transfer learning approach in order to extend the size of training set. Their research contained daily data of West Texas Intermediate covering a time period from January 31, 2005, to December 5, 2016, and daily data of Brent oil covering a range from January 31, 2006, to October 17, 2017, concerning oil factors such as opening price, closing price, lowest price and highest price. Their proposed methodology was evaluated by comparing the performance of a classical LSTM model trained with the initial data and with the data transfer approach. Their experiments showed that their methodology improved the performance of the LSTM model, and thus, the authors stated that the prediction was able to catch most fluctuations of crude oil prices.

Nakano et al. [21] considered improving the traditional "buy-and-hold" strategy presenting a new methodology which exploits the predictions of advanced machine learning models on Bitcoin's high-frequency technical trading. More specifically, they designed ANN-based models to extract the useful trading signals from technical indicators calculated from the time-series return data at time intervals of 15 min. The utilized data in this research concerned historical returns and technical indicators, ranging from December 2017 to January 2018, during which Bitcoin suffers from substantial volatility and a significant number of negative returns. Their preliminary experimental results reported that the utilization of various technical indicators could prevent over-fitting and considerably enhance trading performance.

Ji et al. [16] studied the prediction performance on Bitcoin price of various deep learning forecasting models such as deep neural networks, convolutional neural networks, LSTM networks, deep residual networks and their combinations. In their study, they utilized Bitcoin data from 2590 days (from November 29, 2011, to December 31, 2018), containing 29 features. The authors performed a comprehensive experimental procedure, considering the problems of predicting the next's day Bitcoin price, and whether or not the next day price will increase or decrease. Their numerical experiments demonstrated that deep neural networks reported the best performance for price movement, while the LSTM models exhibited the best performance for forecasting Bitcoins' price, slightly outperforming the rest prediction models.

Pintelas et al. [24, 25] performed a comprehensive research, evaluating advanced deep learning models for forecasting the prices and directional movements of major cryptocurrencies. Furthermore, the authors conducted a detailed discussion, concerning if deep learning models can be trusted as reliable predictors and if the cryptocurrencies

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

prices follow a random walk process. Their experimental analysis presented that even the state-of-the-art deep learning models were unable to create reliable forecasting models. Moreover, the authors stated that a few hidden patterns in cryptocurrency prices may probably exist, although these prices seem to follow almost a random walk process.

Summarizing, most approaches proposed in the literature attempt to exploit deep learning techniques for extracting useful knowledge from time-series data, aiming at obtaining better performance compared to the already existing proposed models. In this research, we propose a different approach and introduce a novel framework for the development of efficient and reliable deep learning models. The novelty of the proposed methodology is that it guarantees the forecasting reliability of the model's predictions, independent of the time-series data and selected deep learning model. It is worth noticing that none of the mentioned research approaches considered to improve both the accuracy and reliability of a deep learning model by exploiting the information provided by the characteristics of the time-series as well as the error of prediction model.

3 Reliability evaluation on times-series forecasting

3.1 Significance of autocorrelation in model's forecasting reliability

Let $y_1, y_2, \ldots, y_n$ be the observations of a time-series. A nonlinear regression model of order m is defined by

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

where $x_t = (y_{t-1}, y_{t-2}, \ldots, y_{t-m}) \in \mathbb{R}^m$ consists of m values of y_t, θ is the parameter vector and ϵ_t is the white noise residual. After the model structure has been defined, function $f(\cdot)$ can be determined by sophisticated machine learning or deep learning methods.

After a prediction model has been successfully fit, it is significant to evaluate and assess how well the model is able to capture patterns. The most commonly utilized metrics for evaluating the regression performance of a forecasting model are mean absolute error (MAE) and root mean square error (RMSE).

Nevertheless, both regression evaluation metrics help determine how close the predicted values are to the actual ones, they do not evaluate whether the model properly fits the time-series data, while the residuals are usually dedicated to evaluate this. In other words, provided that function f is appropriately estimated, the prediction model's residuals

$$\hat{\epsilon}_t = y_t - \hat{y}_t, \tag{2}$$

are identically distributed and asymptotically independent, where $\hat{y}_t$ is the predicted value.

It is worth noticing that in case the assumption of no-autocorrelation in the residuals is violated in a forecasting model, implies that its predictions may be inefficient, since the model has not exploited all the available mined information during the training process. In other words, the dependence between the residuals indicates that the model has not properly fitted the time-series data and there exists significant information left over which should be taken into account.

Two significant tools for testing the existence autocorrelation of the residuals are the auto-correlation function (ACF) plot and the Ljung–Box Q test for residual autocorrelation [6]. More analytically, ACF is obtained from the linear correlation of each residual $\hat{\epsilon}_t$ to the others in different lags, $\hat{\epsilon}_{t-1}, \hat{\epsilon}_{t-2}, \ldots$ and illustrates the intensity of the temporal autocorrelation, while Ljung–Box Q test is a "portmanteau" test which assesses the null hypothesis H_0 that "*a series of residuals exhibits no autocorrelation for a fixed number of lags L*, against the alternative H_1 that "*some autocorrelation coefficient is nonzero*." More specifically, the Ljung–Box Q test statistic is defined by

$$Q = n(n+2)\sum_{k=1}^{L}\frac{\rho_k^2}{n-k}, \tag{3}$$

where ρ_k are autocorrelation coefficients at lag-k, defined by

$$\rho_k = \frac{\sum_{i=1}^{n-k}(y_i - \overline{y})(y_{i+k} - \overline{y})}{\sum_{i=1}^{n}(y_i - \overline{y})^2}, \tag{4}$$

with $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. Under H_0 the statistic Q asymptotically follows a $\chi^2_{(L)}$ distribution. The null hypothesis H_0 is rejected and state that the model exhibits autocorrelation if

$$Q > \chi^2_{1-\alpha,L} \tag{5}$$

where the critical value of the Chi-square distribution is defined for significance level α, or critical level $p = 1 - \alpha$, known as p value.

3.2 Strict stationarity and weak stationarity

Time-series exhibit a variety of properties which appear so often that are called stylized facts which include autocorrelation, long memory, fractal and multi-fractal properties. The major drawback when dealing with prices or values (levels) of such series is, from the stochastic processes point of view, that they follow a random walk process. The autocorrelation coefficients ρ_k, with $k > 1$ are statistically

Journal : Large 521 | Dispatch : 7-7-2020 | Pages : 19
Article No. : 5169 | LE | TYPESET
MS Code : NCAA-D-20-01027R1 | CP | DISK

significant for a large number of lags L and the first-order autocorrelation coefficient ρ_1 is equal to one [27]. Such series are also named *unit root time-series* or *integrated of order one* and are denoted by $I(1)$.

Under these conditions, modeling the levels of such series is inefficient because the residuals of the models display autocorrelation setting the whole structure of statistical significance under question. In order to study efficiently these series, they have to be stationary, which is a highly significant for ensuring the development of a reliable prediction model.

Suppose that $F_y(y_{t_1+\tau}, \ldots, y_{t_n+\tau})$ is the cumulative distribution function of the unconditional joint distribution of $\{y_t\}$ at times $t_{1+\tau}, \ldots, t_{n+\tau}$, then the stochastic process $\{y_t\}$ is strictly stationary if

$$F_y(y_{t_1+\tau}, \ldots, y_{t_n+\tau}) = F_y(y_{t_1}, \ldots, y_{t_n}), \tag{6}$$

for all $\tau, t_1, \ldots, t_n \in \mathbb{R}$ and $n \in \mathbb{N}$. Nevertheless, in time-series the strong form of stationarity is relaxed leading to the *weak-stationarity* or *covariance stationarity* [6]. Therefore, a stochastic process is covariance stationary if the mean is constant, the second moment is finite, and the covariance function depends only on the difference between t_1 and t_2 and needs to be indexed by only one variable, i.e.,

$$cov_{yy}(t_1, t_2) = cov_{yy}(t_1 - t_2, 0). \tag{7}$$

where cov_{yy} is the auto-covariance of series y_t. Summarizing, stationarity means that the statistical properties of a stochastic process which generates a time-series are constant over time. Stationary processes are easier to analyze, model, and investigate, and it has been a common assumption of many practices involving statistical inference, modeling and forecasting.

Having identified the problem, a solution for stationarity comes from the partial autocorrelation function, where the lag-k coefficient $\phi_{k,k}$ is given by the following formula

$$\begin{cases} \phi_{k,k} &= \dfrac{\rho_k - \sum_{j=1}^{k-1} \phi_{k-1,j}\, \rho_{k-j}}{1 - \sum_{j=1}^{k-1} \phi_{k-1,j}\, \rho_{k-j}}, \\ \phi_{k,j} &= \phi_{k-1,j} - \phi_{k,k}\, \phi_{k-1,k-j}, \end{cases} \tag{8}$$

for $k > 1$ and $\phi_{1,1} = \rho_1$ [26]. Clearly, in case the series exhibits a unit root, that is $\rho_1 = 1$, it immediate follows that the first-order partial autocorrelation coefficient $\phi_{1,1}$ will be one. The significance of the partial autocorrelation function is that if only the first coefficient is statistically significant and the rest are not, which is the usual case in most time-series of scientific interest, then this is a guide that the initial series should be differenced by using the first differences of the series, namely

$$\Delta_t = y_t - y_{t-1}. \tag{9}$$

Therefore, taking the first difference of the levels of the series results in stationarity and these series are named *integrated of order zero* and are denoted by $I(0)$.

However, when dealing with time-series there might be an overlapping of a variety of non-stationarities, including unit-roots, structural breaks, level shifts, seasonal cycles, or a changing variance. Notice that the typical transformation when the series is $I(1)$ (non-stationary) is to take the first differences of the series and transform it to a series $I(0)$ (stationary), while if the series incorporate structural breaks or a changing variance, i.e., due to crises, a nonlinear Box-Cox transformation [22] is the appropriate available option. A Box-Cox transformation is a way to transform non-normal dependent variables into a normal shape, since normality is a critical assumption for many statistical techniques. The one-parameter Box-Cox transformation is defined as

$$y_t = \begin{cases} \dfrac{y_t^{\lambda} - 1}{\lambda}, & \text{if } \lambda \neq 0; \\ \ln y_t, & \text{if } \lambda = 0. \end{cases} \tag{10}$$

where common nonzero Box-Cox transformations are for $\lambda = -3, -2, -0.5, 0, 0.5, 1$ and 2. The large majority of time-series follow the rule $\lambda = 0$; therefore, the stationarity of these series is achieved via the returns that is the first logarithmic differences,

$$r_t = \ln y_t - \ln y_{t-1} \approx \frac{y_t - y_{t-1}}{y_{t-1}}, \tag{11}$$

the last expression being the percentage change or *returns*.

4 Research methodology and proposed framework

In this section, we introduce our proposed methodology for considerably improving the performance of a deep learning model for time-series forecasting in terms of accuracy and reliability. based on the well-established econometric theory and time-series analysis with respect to stationarity and non-stationarity properties.

Revisiting the problem, when applying a machine learning or a deep learning model to time-series for forecasting, the levels of the series are not-stationary, meaning that they possess unit roots and some order of integration[1].

[1] Non-stationary time-series which can be transformed in this way are called series integrated of order d. Usually, the order of integration is either $I(0)$ or $I(1)$; it's extremely rare to see values for d that are 2 or more in real-world applications [7]. Additionally, all series in this research are $I(1)$.

Journal : Large 521 | Dispatch : 7-7-2020 | Pages : 19
Article No. : 5169 | LE | TYPESET
MS Code : NCAA-D-20-01027R1 | CP | DISK

Notice that the identification of a unit root in a time-series can be easily performed via the Augmented Dickey–Fuller (ADF) test [6, 23]. The testing procedure is applied to the model,

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \sum_{i=1}^{k-1} \delta_i \Delta y_{t-i} + \epsilon_t, \quad (12)$$

where α is a constant, β is the coefficient of time trend, and $\gamma = (\rho_1 - 1)$ where ρ_1 denotes the first-order autocorrelation coefficient. It is worth mentioning that k is the lag order of the autoregressive process chosen so that no serial correlation exists in the residuals ϵ_t, ensuring that the test is efficient and reliable. If $\alpha = 0$ and $\beta = 0$, then we have a random walk stochastic process, while if $\alpha \neq 0$ and $\beta = 0$, we have a random walk stochastic process with drift. The unit root test is carried out testing the statistical significance under the null hypothesis $H_0 : \{\gamma = 0 \text{that is} \rho = 1\}$ against the alternative hypothesis $H_1 : \{\gamma < 0 \text{that is} \rho < 1\}$.

The solution depending on the nature of the time-series is to iteratively take the first differences (9) or the returns (11) until the series is made stationary, implying that the first-order autocorrelation coefficient ρ_1 is less than one. It is worth noticing that the series transformation based on first differences or returns implies that the autocorrelation in the residuals of the model is removed. This indicates that the prediction model is able to explain the data much better, since it captures all possible nonlinearities, ensuring the efficiency and effectiveness of the model.

Table 1 presents the pseudo-code of our proposed framework. Initially, the time-series data are imported (Step 1). Then, the ADF test is applied to examine whether the levels of the series are non-stationary, meaning that they possess a unit root (Step 2). In case the series is non-stationary the transformation based on first differences or returns is iteratively applied on the training data until the new transformed series is stationary (Steps 4–7). Subsequently, the new transformed time-series data are used for training the prediction model (Step 8).

In contrast, in case the series is stationary, then the levels of time-series are used for training the prediction model (Step 10). Subsequently, the prediction model's errors on the training data are used for further examination and testing. Notice that a model trained with a series which does not possess a unit root and has not been differenced, may exhibit autocorrelation in the training errors. In other words, although for any reasonable model the predicted values will be close to the real values, the existence of large autocorrelation coefficients characterizes the model as inefficient [28]. Therefore, the residuals of the training data are examined for autocorrelation by simply performing ACF plots and/or Ljung–Box Q test (Step 11). In case the residuals possess autocorrelation, the proposed transformation is applied on the training data and the model is re-trained using the new transformed series (Steps 13–14). Notice that if the levels of the series are stationary and the residuals on the training set exhibit no autocorrelation, then there is no need to transform the series, since this will lead to the dangerous phenomenon of over-differencing the series. In other words, over-differencing leads the whole process to be "non-invertible" and lacks an infinite-order autoregressive representation. Figure 1 presents an overview of the proposed architecture in the form of a flowchart.

Finally, it is worth mentioning that in case the model is trained with a transformed series based on first differences or returns, the reverse transformation is used in the predictions of the model to obtain the prediction for the levels of the original time-series.

5 Data

In our research, we utilized three benchmark datasets from the popular real-world application domains: finance, commodity and cryptocurrency, in order to demonstrate the efficiency of our proposed methodology.

From finance domain, we utilized data from January 1, 2013, to December 31, 2019, of Standard & Poor's 500 (S&P500) prices in USD from http://finance.yahoo.com Web site. The data were divided into training set consisting of daily Brent prices from January 1, 2015, to December 31, 2018 (4 years), and a testing set consisting of daily prices from January 1, 2018, to December 31, 2019 (1 year).

From commodity domain, we utilized daily data from January 1, 2013, to December 31, 2019, of Brent prices in USD from https://www.eia.gov/ Web site. The data were divided into training set consisting of daily Brent prices from January 1, 2015, to December 31, 2018 (4 years), and a testing set consisting of daily prices from January 1, 2018, to December 31, 2019 (1 year).

From cryptocurrency domain, we utilized daily data from January 1, 2015, to December 31, 2019, of Bitcoin (BTC) cryptocurrency in USD from https://coinmarketcap.com Web site. The data were divided into training set consisting of daily BTC prices from January 1, 2015, to June 30, 2018 (3.5 years), and a testing set consisting of daily prices from July 1, 2018, to December 31, 2019 (1/2 year).

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.

Table 2 summarizes the descriptive statistics for the training and testing set of each dataset, including the

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

Table 1 Proposed framework to enhance deep learning in time-series forecasting

Step 1. Import time-series data.
Step 2. Apply the ADF unit root test.
Step 3. **If** (*Time-series possess a unit root*) **then**
Step 4. **repeat** /* *Time-series is not I*(0) */
Step 5. Apply the transformation based on first differences (9) or returns (11). /* *non-stationary* /*
Step 6. Apply the ADF unit root test.
Step 7. **until** (*Time-series is stationary*)
Step 8. Train the forecasting model using the transformed time-series.
Step 9. **else**
Step 10. Train the forecasting model using the original time-series. /* *Time-series is I*(0) */
Step 11. Calculate the residuals on the training set. /* *stationary* /*
Step 12. **If** (*residuals possess autocorrellation*) **then**
Step 13. Apply the transformation based on first differences (9) or returns (11).
Step 14. Re-train the forecasting model using the transformed time-series.
Step 15. **end if**
Step 16.end if /* *Training process completed* */

measures: Minimum, Maximum, Mean, Standard Deviation (Std. Dev.), Median, Skewness and Kurtosis, for presenting the nature of the distribution. Additionally, Table 3 presents the increase and decrease cases and the corresponding percentages in S&P500, Brent and BTC datasets.

6 Experimental analysis

In this section, we apply our proposed methodology to the S&P500, Brent and BTC time-series to identify whether or not the training data are stationary, utilizing the ADF unit root test.

Table 4 presents the results of the ADF unit root test for the training data of all series under consideration, i.e., S&P500, Brent and BTC, performed on the level of the original series. By taking into consideration, the *t*-statistics (*t*-stat.) and the associated *p* values, we conclude that the null hypothesis H_0: "*the levels possess a unit root and are non-stationary*" is accepted for S&P500, Brent and BTC series.

In the sequel, we perform the ADF test to the transformed time-series based on both first differences (first differenced series) and returns (returns series), to examine if the unit root has been removed, according to our proposed framework.

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. Clearly, performing either the first differences or the returns transformation clearly solves the unit root problem, since all *p* values are practically zero and therefore the null hypothesis H_0 is rejected and all transformed series are indeed stationary.

Thus, both transformed series are "suitable" for fitting a deep learning model which will present no autocorrelation in the errors, and a big improvement of the forecasting performance is expected, compared with the same model trained with the original non-transformed series.

6.1 Numerical experiments

In the sequel, we present a comprehensive experimental analysis, to evaluate the efficiency and reliability of our proposed methodology. More specifically, we compare the performance of two efficient deep learning forecasting models trained with the levels of the time-series (Time-series) and with the two transformed series based on first differences (first differenced series) and returns (returns series).

Under exhaustive experimentation (utilizing different number of the CNN and LSTM layers, different number of units in the LSTM layers, different number of filter in CNN layers), the selected models were an LSTM model which consist of a LSTM layer of 50 units and an output layer of one neuron (Fig. 2) and a CNN–LSTM which consists of two convolutional layers of 16 and 32 filters of size (2,) with the same padding, followed by a LSTM layer of 50 units and an output layer of one neuron (Fig. 3). Notice that both models were trained with Adaptive Moment Estimation [17] with a batch size equal to 128, using a mean-squared loss function. The implementation code was written in Python 3.4 using Keras library [15] on a laptop (Intel(R) Core(TM) i7-6700HQ CPU 2.6GHz and 16GB

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

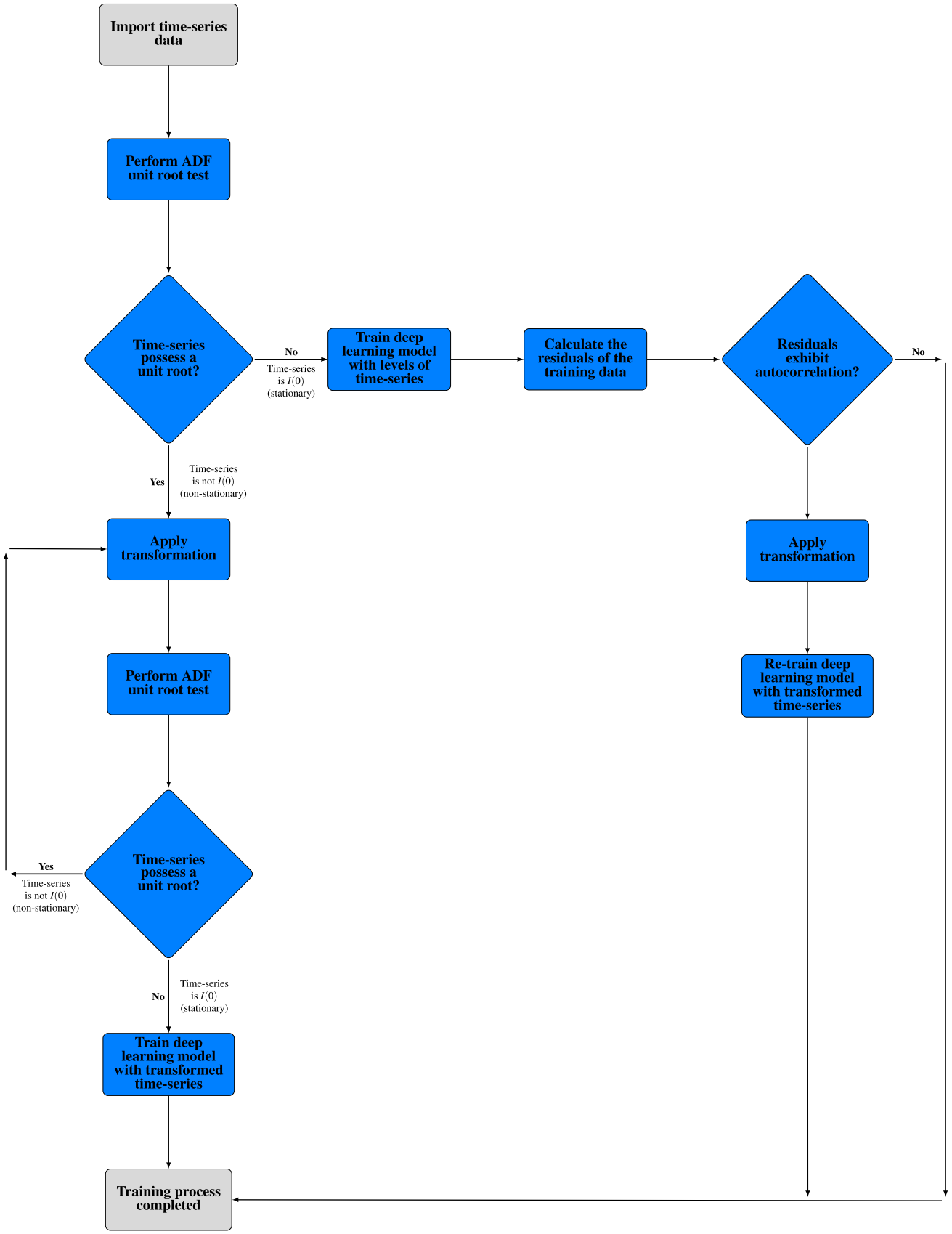


Fig. 1 Flowchart

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

RAM) running Windows 10.0 operating system. Each forecasting model was trained with the traditional time-series, the first differenced series and returns series, utilizing four different values of window size m, i.e., $m = 4, 6, 9$ and 12. Finally, in order to reduce exponential trend and homogenize the variability and stability of the patterns, the traditional time-series data were transformed utilizing a natural logarithm (ln) and the model's predicted value is used to predict the price on the following day. Moreover, we recall that in case the models were trained using the first difference or returns transformation to series, the reverse transformation is utilized and to predict the price on the following day.

The regression performance was evaluated utilizing the metrics: mean absolute error (MAE) and root mean square error (RMSE), while for the binary classification problem of predicting whether the price would increase or decrease on the following day, four performance metrics were used: Accuracy (Acc), F_1-score (F_1), Sensitivity (Sen), Specificity (Spe), Positive Predicted Values (PPR) and Negative Predictive Values (NPV) which are defined by

$$\mathrm{Acc} = \frac{\mathrm{TP+TN}}{\mathrm{TP+FP+FN+FP}}, \tag{13}$$

$$\mathrm{F_1} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}, \tag{14}$$

$$\mathrm{Spe} = \frac{\mathrm{TP}}{\mathrm{TP+FN}}, \tag{15}$$

$$\mathrm{Spe} = \frac{\mathrm{TN}}{\mathrm{TN+FP}}, \tag{16}$$

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP+FP}}, \tag{17}$$

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN+FN}} \tag{18}$$

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

Moreover, we included area under curve (AUC) metric in our analysis which constitutes one of the most significant classification metrics and it is presented using the receiver

Table 3 The number of up and down movements of S&P 500, Brent and BTC datasets

Data	Decrease	%	Increase	%
S&P500				
Training set	815	54.01	694	45.99
Testing set	149	59.36	102	40.64
Brent				
Training set	738	47.19	826	52.81
Testing set	140	53.64	121	46.36
BTC				
Training set	900	55.87	711	44.13
Testing set	102	47.44	113	52.56

Table 4 ADF unit root test of all series under consideration

Series	S&P500	Brent	BTC
Time-series			
t stat.	− 2.8469	− 1.1599	− 2.7497
p value	0.1806	0.9170	0.2166

Table 5 ADF unit root test of all transformed times-series based on first differences and returns

Series	S&P500	Brent	BTC
First differenced series			
t stat.	− 39.146	− 38.516	− 7.3402
p value	0.0000*	0.0000*	0.0000*
Returns series			
t stat.	− 39.663	− 38.259	− 39.893
p value	0.0000*	0.0000*	0.0000*

Table 2 Descriptive statistics for S&P500, Brent and BTC prices

Data	Minimum	Maximum	Mean	SD	Median	Skewness	Kurtosis
S&P500							
Training set	1457.15	2930.75	2153.78	367.52	2087.90	0.31	− 0.75
Testing set	2447.89	3240.02	2912.09	149.32	2918.65	− 0.26	0.17
Brent							
Training set	26.01	118.90	71.46	25.84	63.27	0.43	− 1.28
Testing set	50.57	74.94	64.31	4.44	63.99	0.16	− 0.30
BTC							
Training set	178.10	19497.40	3261.48	3675.02	1152.36	1.40	1.84
Testing set	6640.52	13016.23	9221.26	1466.76	9244.97	0.27	− 0.93

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

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.

6.1.1 S&P500

Tables 6 and 7 present the performance comparison of both LSTM and CNN–LSTM forecasting models, respectively, for S&P500 dataset. The LSTM model improved its average performance, in terms of MAE and RMSE scores by 30.57%-43.88% and 18.73%-37.31%, respectively, when trained with the first differenced series, while the CNN–LSTM model considerably improved its MAE average performance by 11.35%-45.34% and its RMSE average performance by 4.82%-34.59%, in the same situation. Furthermore, the LSTM model reduced its average MAE and RMSE scores by 23.76%-49.40% and 11.51%-45.45% in case it was trained with the returns of S&P500 prices, while the CNN–LSTM model reduced its average MAE performance by 21.10%-48.98% and its RMSE average performance by 11.89%-38.96%. Summarizing, we conclude that the regression performance of both LSTM and CNN–LSTM forecasting models was considerably improved, utilizing the first differenced and returns series, instead of the traditional S&P500 time-series.

Furthermore, the classification performance of both prediction models was also improved utilizing our proposed methodology. More specifically, both LSTM and CNN–LSTM models were biased in case they were trained with the traditional time-series. In contrast, the trade-off between sensitivity and specificity as well as between positive and negative predictive values of both models was considerably increased in case the models were trained with the first differenced.

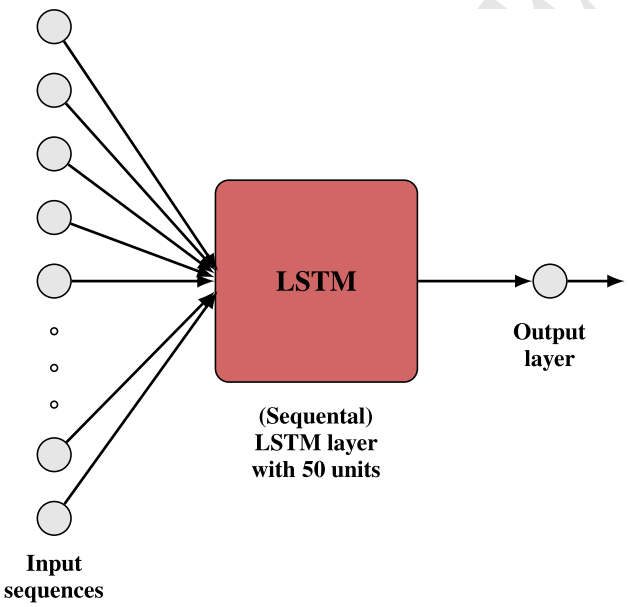


Fig. 2 LSTM forecasting model architecture

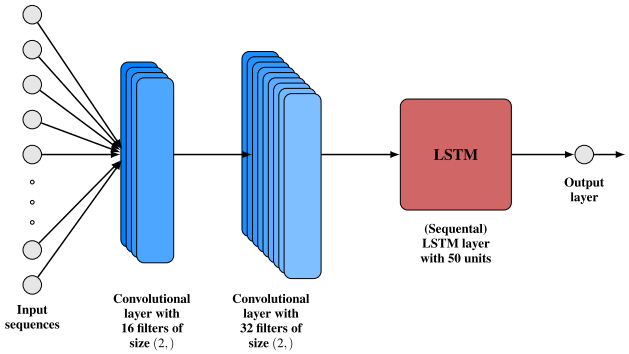


Fig. 3 CNN–LSTM forecasting model architecture

It is worth noticing that both LSTM and CNN–LSTM models exhibited the highest classification performance in case they were trained with the first differenced series and the best regression performance when they were trained with the returns series. Moreover, the LSTM model trained with the first differenced time-series reported the best classification performance for all values of window size m and the best regression performance for $m = 6, 9$ and 12. The CNN–LSTM model trained with the first differenced and returns series reported the best performance relative to classification and regression accuracy, respectively.

6.1.2 Brent

Tables 8 and 9 present the performance comparison for Brent forecasting problem of LSTM and CNN–LSTM, respectively. The LSTM and CNN–LSTM models improved their average MAE score by 6.44–29.66% and 5.88–17.52%, respectively, in case they were trained with the first differenced series, instead of the traditional series. Furthermore, their average RMSE score was reduced by 5.73–25.46% and 2.88–10.33% in the same situation. However, the regression performance of both forecasting models worsens, in case they were trained using the returns series.

Regarding the classification performance, both LSTM and CNN–LSTM models were biased in case they were trained with the traditional series. On the other hand, LSTM considerably improved its classification performance utilizing either first differenced or returns series as training data in terms of trade-off between sensitivity and specificity as well as the trade-off between positive and negative predicted values. Additionally, the CNN–LSTM significantly improved its classification performance using first differenced series as training data, in terms of both accuracy and trade-off between sensitivity and specificity.

Journal : Large 521 | Dispatch : 7-7-2020 | Pages : 19
Article No. : 5169 | LE | TYPESET
MS Code : NCAA-D-20-01027R1 | CP | DISK

It is also worth mentioning that both LSTM and CNN–LSTM forecasting models improved their F_1-score, in case they were trained with the transformed series.

The performance of the forecasting models, relative to the value of the window size m, both models improved their performance as the value of window size increases. Moreover, it is worth mentioning that the best overall performance was reported by CNN–LSTM trained with the first differenced series with $m = 12$.

6.1.3 Bitcoin

Tables 10 and 11 present the performance comparison of LSTM and CNN–LSTM, respectively, relative to BTC dataset. Similar conclusions can be drawn with the previous benchmarks. More specifically, the LSTM model improved its MAE and RMSE average performance by 24.29–31.90% and 19.44–23.38%, respectively, when trained with the first differenced series instead of the traditional series, while the CNN–LSTM model improved its MAE average performance by 14.34–38.12% and its RMSE average performance by 6.0–29.1%, in the same situation. Moreover, the MAE and RMSE average performance of the LSTM model was reduced by 21.14–36.59% and 15.99–27.49% in case it was trained with the returns of Bitcoin's prices, while the CNN–LSTM model improved its MAE average performance by 8.57–39.57% and its RMSE average performance by 2.56–29.52%, in the same situation. Summarizing, we can easily conclude that the regression performance of both forecasting models was considerably improved, utilizing our proposed methodology for data preparation.

Regarding the classification performance, our proposed methodology increased the accuracy of both prediction models. More analytically, the interpretation of Tables 10 and 11 reveals that LSTM and CNN–LSTM models were biased when trained with the traditional time-series. In contrast, the trade-off between sensitivity and specificity as well as the trade-off between positive and negative predictive values of both forecasting models was considerably improved, in case they were trained with the first differenced series or the returns series. Finally, AUC and F_1-score of both models were improved in case they were trained with the transformed series instead of the traditional time-series.

The LSTM model trained with the first differenced series exhibited the best classification performance for all values of window size m and the best regression performance for $m = 6, 9$ and 12. Moreover, the CNN–LSTM model exhibited the lowest (best) MAE and RMSE scores, in case it was trained with the returns series, while it presented the best overall classification performance, in case it was trained with the first differenced series. Finally, it is worth mentioning that both LSTM and CNN–LSTM models exhibited the best regression performance when they were trained with the returns series and the highest classification performance, in case they were trained with the first differenced series.

6.2 Reliability evaluation of the forecasts

In the sequel, we evaluate the reliability of all forecasting models by examining the existence of autocorrelation in the residuals utilizing the Auto-Correlation Function (ACF) plot and the Ljung–Box Q test for residual autocorrelation [6]. In other words, we examine whether each trained model has properly fitted the time-series by examining whether the residuals are identically distributed and

Table 6 Performance comparison of the LSTM model for S&P500 dataset

Series	Horizon	MAE	RMSE	Acc (%)	AUC	F_1	Sen	Spe	PPV	NPV
Time-series	4	33.31	42.80	50.42	0.532	0.606	0.595	0.470	0.651	0.405
First differenced series		18.69	26.83	47.87	0.467	0.546	0.530	0.404	0.565	0.410
Returns series		16.85	23.35	46.59	0.435	0.570	0.597	0.274	0.546	0.347
Time-series	6	34.49	42.35	47.79	0.514	0.425	0.324	0.703	0.655	0.415
First differenced series		20.10	29.38	48.11	0.466	0.554	0.545	0.388	0.566	0.417
Returns series		19.31	31.56	49.00	0.460	0.590	0.617	0.304	0.565	0.352
Time-series	9	29.84	39.02	47.39	0.520	0.385	0.277	0.762	0.631	0.409
First differenced series		20.58	29.27	50.36	0.497	0.559	0.531	0.463	0.593	0.451
Returns series		18.94	29.40	50.07	0.486	0.574	0.565	0.406	0.583	0.409
Time-series	12	27.14	38.04	47.59	0.509	0.471	0.385	0.634	0.606	0.413
First differenced series		18.84	25.43	49.40	0.489	0.547	0.514	0.465	0.585	0.435
Returns series		20.69	30.12	47.12	0.466	0.524	0.493	0.439	0.562	0.402

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

Table 7 Performance comparison of the CNN–LSTM model for S&P500 dataset

Series	Horizon	MAE	RMSE	Acc (%)	AUC	F_1	Sen	Spe	PPV	NPV
Time-series	4	33.40	38.70	48.59	0.501	0.395	0.421	0.581	0.601	0.272
First differenced series		18.26	25.31	50.20	0.478	0.551	0.529	0.426	0.575	0.411
Returns series		17.04	23.62	46.02	0.426	0.568	0.601	0.251	0.540	0.330
Time-series	6	23.40	29.67	51.67	0.535	0.494	0.437	0.634	0.649	0.437
First differenced series		20.75	28.24	51.41	0.516	0.554	0.507	0.525	0.610	0.451
Returns series		18.47	26.14	48.35	0.466	0.557	0.550	0.383	0.566	0.398
Time-series	9	35.46	42.18	46.45	0.509	0.356	0.275	0.743	0.623	0.412
First differenced series		22.78	31.94	53.41	0.458	0.513	0.480	0.436	0.554	0.403
Returns series		20.61	29.82	52.13	0.508	0.561	0.527	0.488	0.601	0.414
Time-series	12	33.96	40.96	45.38	0.489	0.389	0.302	0.677	0.577	0.398
First differenced series		23.97	33.03	52.08	0.459	0.519	0.487	0.465	0.557	0.395
Returns series		23.97	33.03	51.20	0.459	0.519	0.487	0.432	0.557	0.391

Table 8 Performance comparison of the LSTM model for Brent dataset

Series	Horizon	MAE	RMSE	Acc (%)	AUC	F_1	Sen	Spe	PPV	NPV
Time-series	4	1.40	1.81	53.05	0.543	0.442	0.370	0.717	0.618	0.496
First differenced series		0.99	1.35	48.42	0.485	0.499	0.479	0.490	0.522	0.467
Returns series		1.10	1.92	51.24	0.506	0.506	0.478	0.533	0.541	0.480
Time-series	6	1.33	1.75	53.75	0.545	0.497	0.439	0.652	0.608	0.501
First differenced series		1.25	1.86	50.19	0.504	0.503	0.472	0.537	0.541	0.497
Returns series		1.31	2.18	50.27	0.504	0.509	0.484	0.525	0.541	0.488
Time-series	9	1.62	2.03	52.36	0.530	0.462	0.449	0.610	0.584	0.508
First differenced series		1.37	1.86	51.43	0.514	0.534	0.519	0.508	0.550	0.498
Returns series		1.21	1.84	50.19	0.500	0.528	0.525	0.475	0.535	0.486
Time-series	12	1.68	2.16	53.67	0.536	0.536	0.544	0.528	0.588	0.494
First differenced series		1.45	1.94	51.66	0.515	0.543	0.537	0.493	0.551	0.499
Returns series		1.28	2.03	51.66	0.513	0.552	0.557	0.470	0.550	0.498

asymptotically independent. We recall that the Ljung–Box Q test is a "portmanteau" test which assesses the null hypothesis H_0 that "*a series of residuals exhibits no autocorrelation for a fixed number of lags L*," against the alternative hypothesis H_1 that "*some autocorrelation coefficient is nonzero*."

Tables 12 and 13 present the information of the statistical analysis performed by Ljung–Box Q test for $L = 10$ of LSTM and CNN–LSTM, respectively. Clearly, the null hypothesis H_0 of no autocorrelation in the residuals is accepted, in case the models were trained with the first differenced or returns series, relative to all benchmarks and window sizes. On the other hand, both prediction models reject the H_0 in case they were trained with the traditional time-series.

For completeness, we also present the ACF plots of LSTM and CNN–LSTM for S&P500, Brent and BTC datasets in order to illustrate the intensity of the temporal autocorrelation. In each ACF plot, the confident limits are denoted with blue dashed lines and are constructed assuming that the residuals follow a Gaussian probability distribution. Notice that the ACF plot of each model for S&P500, Brent and BTC datasets was calculated for $m = 9$, $m = 9$ and $m = 6$, respectively, for which the models exhibited the best performance.

Figures 4, 5 and 6 present the ACF plots of LSTM model for S&P500, Brent and BTC datasets, respectively. The ACF plots of the forecasting model trained with the traditional time-series violate the assumption of no autocorrelation in the residuals. More specifically, the significant spikes that occurred in several lags suggest the

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

Table 9 Performance comparison of the CNN–LSTM model for Brent dataset

Series	Horizon	MAE	RMSE	Acc (%)	AUC	F_1	Sen	Spe	PPV	NPV
Time-series	4	1.13	1.51	53.01	0.535	0.494	0.464	0.607	0.583	0.500
First differenced series		0.98	1.35	50.77	0.508	0.522	0.502	0.514	0.545	0.491
Returns series		1.17	1.61	49.10	0.468	0.507	0.508	0.428	0.508	0.457
Time-series	6	1.35	1.74	53.17	0.548	0.421	0.330	0.765	0.632	0.496
First differenced series		1.17	1.64	51.47	0.521	0.542	0.509	0.521	0.550	0.505
Returns series		1.67	2.39	52.16	0.510	0.546	0.490	0.558	0.564	0.495
Time-series	9	1.54	1.94	52.97	0.544	0.414	0.353	0.734	0.612	0.500
First differenced series		1.27	1.80	53.74	0.548	0.541	0.521	0.535	0.565	0.511
Returns series		2.07	2.88	51.35	0.506	0.549	0.511	0.517	0.550	0.497
Time-series	12	1.44	1.90	53.82	0.537	0.551	0.558	0.515	0.587	0.500
First differenced series		1.35	1.85	54.57	0.545	0.564	0.549	0.542	0.581	0.515
Returns series		1.87	2.63	54.44	0.538	0.565	0.518	0.575	0.585	0.497

Table 10 Performance comparison of the LSTM model for BTC dataset

Series	Horizon	MAE	RMSE	Acc (%)	AUC	F_1	Sen	Spe	PPV	NPV
Time-series	4	404.57	551.17	51.17	0.500	0.462	0.693	0.294	0.501	0.524
First differenced series		275.52	428.92	52.54	0.524	0.499	0.501	0.547	0.500	0.549
Returns series		256.53	399.66	51.08	0.514	0.529	0.580	0.448	0.499	0.542
Time-series	6	436.77	576.23	50.85	0.501	0.446	0.653	0.348	0.509	0.521
First differenced series		297.88	459.85	53.66	0.538	0.532	0.558	0.517	0.521	0.565
Returns series		302.62	469.09	50.01	0.491	0.480	0.497	0.484	0.486	0.515
Time-series	9	447.30	617.72	49.48	0.498	0.475	0.703	0.259	0.509	0.526
First differenced series		338.67	497.64	53.15	0.529	0.491	0.488	0.571	0.526	0.555
Returns series		352.73	518.96	50.86	0.510	0.505	0.533	0.487	0.509	0.529
Time-series	12	483.96	672.34	50.19	0.497	0.477	0.702	0.396	0.508	0.510
First differenced series		348.62	515.16	51.08	0.514	0.521	0.569	0.458	0.509	0.541
Returns series		363.98	546.14	50.70	0.510	0.518	0.565	0.455	0.506	0.535

model's predictions may be inefficient. In contrast, all ACF plots of the LSTM trained with the first differences and returns of prices reveal that there is no autocorrelation in the residuals which suggests the reliability of the model and advocates the efficiency of its forecasts.

Figures 7, 8 and 9 show the ACF plots of CNN–LSTM model for S&P500, Brent and BTC datasets, respectively. Clearly, all ACF plots of CNN–LSTM model trained with the first differenced and returns series illustrate that there exists no autocorrelation in the residuals. This implies that the model is reliable, with respect to the efficiency of its forecasts. On the other hand, the significant spikes presented in Figs. 7a, 8a and 9a reveal the CNN–LSTM trained with the traditional time-series has not properly fitted the training data and exhibited unreliable predictions.

7 Discussion

In this section, we perform a discussion relative to the theoretical and experimental contribution of our research.

We presented a detailed theoretical background regarding the problem of time-series forecasting and the reliability of the forecasts of a prediction model. Since most time-series datasets are extremely noisy and chaotic by nature, the development of a reliable deep learning prediction models is considered a significantly challenging task. Moreover, the achievement of high accuracy or low RMSE score cannot be considered as a reliable metric since a model may just accidentally perform well on a specific time period, while on a new different period, it may exhibit a totally different and probably poor prediction performance.

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

Table 11 Performance comparison of the CNN–LSTM model for BTC dataset

Series	Horizon	MAE	RMSE	Acc (%)	AUC	F_1	Sen	Spe	PPV	NPV
Time-series	4	315.41	448.46	49.58	0.489	0.512	0.693	0.277	0.488	0.516
First differenced series		270.19	421.57	53.33	0.535	0.533	0.562	0.507	0.527	0.563
Returns series		288.38	436.99	46.81	0.471	0.487	0.534	0.409	0.479	0.493
Time-series	6	366.81	514.74	52.21	0.513	0.476	0.489	0.552	0.518	0.539
First differenced series		269.45	415.27	53.99	0.524	0.511	0.525	0.523	0.519	0.549
Returns series		256.97	398.49	53.94	0.540	0.527	0.544	0.536	0.524	0.566
Time-series	9	399.61	534.20	51.41	0.499	0.531	0.584	0.383	0.500	0.536
First differenced series		258.88	396.16	51.24	0.514	0.515	0.546	0.482	0.507	0.541
Returns series		253.61	393.25	50.00	0.501	0.494	0.517	0.485	0.505	0.527
Time-series	12	417.20	553.32	51.40	0.510	0.510	0.592	0.399	0.511	0.523
First differenced series		258.15	392.60	50.70	0.507	0.488	0.509	0.483	0.505	0.538
Returns series		252.12	389.97	50.52	0.506	0.499	0.520	0.492	0.503	0.531

In this research, we demonstrated theoretically whether time-series data are "*suitable*" for fitting a deep learning model, which constitutes the main contribution of our research. In other words, we introduced a novel framework which can efficiently identify if a time-series is suitable for developing and training a deep learning model, which will perform reliable and stable prediction performances, independent of the characteristics of the series in any time period.

By the term "*suitable*," we mean that the time-series data has successfully passed our proposed theoretical criteria and it can be used for training a prediction model. In contrast, if the series fails satisfying the requested criteria, then it is considered as "*unsuitable*" and every attempt for building a reliable prediction model will be probably in vain. Therefore, we provide a "starting point" for any attempt on developing any prediction framework for any time-series forecasting problem. This starting point is indeed the critical point in which every attempt and investment for building a model will result in a stable and reliable predictor or it will be totally wasted out in the case that the utilized starting dataset was unsuitable. By identifying the suitability of any time-series data, a "green light" for the machine learning developer is provided, in order to invest computational effort for building a forecasting framework.

Furthermore, we established a novel and complete framework which provides a solution for any "*unsuitable*" identified time-series by performing a transformation based on first differences or returns and transform these series to "*suitable*." Although these two techniques were well known as a rule of thumb for transformation and preprocessing for time-series data, it was not proved why, when and how these formulae work and if they actually can be successfully applied. Most approaches were relying on a "trial and error" logic something not appropriate and viable especially on cases when costly and time-consuming real-world projects aim to build accurate and reliable forecasting models. In this work, we proved that these formulae actually filtered these "*unsuitable*" data,

Table 12 Ljung–Box Q test for 10 lags with significance level $\alpha = 5\%$ (LSTM)

Series	Horizon	p value	H_0	p value	H_0	p value	H_0
Time-series	4	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.170	Accepted	0.692	Accepted	0.764	Accepted
Returns series		0.071	Accepted	0.370	Accepted	0.228	Accepted
Time-series	6	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.295	Accepted	0.271	Accepted	0.052	Accepted
Returns series		0.486	Accepted	0.733	Accepted	0.383	Accepted
Time-series	9	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.286	Accepted	0.143	Accepted	0.235	Accepted
Returns series		0.244	Accepted	0.113	Accepted	0.110	Accepted
Time-series	12	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.325	Accepted	0.790	Accepted	0.391	Accepted
Returns series		0.423	Accepted	0.172	Accepted	0.400	Accepted

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

Table 13 Ljung–Box Q test for 10 lags with significance level $\alpha = 5\%$ (CNN–LSTM)

Series	Horizon	p value	Null H_0	p value	Null H_0	p value	Null H_0
Time-series	4	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.102	Accepted	0.622	Accepted	0.834	Accepted
Returns series		0.137	Accepted	0.344	Accepted	0.254	Accepted
Time-series	6	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.657	Accepted	0.416	Accepted	0.160	Accepted
Returns series		0.180	Accepted	0.168	Accepted	0.064	Accepted
Time-series	9	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.105	Accepted	0.083	Accepted	0.560	Accepted
Returns series		0.619	Accepted	0.382	Accepted	0.435	Accepted
Time-series	12	0.000	Rejected	0.000	Rejected	0.000	Rejected
First differenced series		0.059	Accepted	0.060	Accepted	0.629	Accepted
Returns series		0.056	Accepted	0.869	Accepted	0.217	Accepted

eliminating this costly and time consuming "trial and error" approach.

It is worth mentioning that an attractive property of our proposed framework is that it can be easily extended to cover the wider scientific area of time-series forecasting applications without the requirement of any extra modifications or additional constraints. In more detail, the proposed framework performs an efficient preprocessing step in order to exploit the internal representation of the times-series, through the utilization of statistic and econometric test. Conclusively, we point out that our experimental analysis indicated that although deep learning models constitute a widely accepted and efficient choice for time-series forecasting, our proposed framework provides a significant boost in increasing the forecasting performance.

Nevertheless, an extensive research is under consideration to identify which of these two methodologies can be a priori efficiently applied depending on the characteristics of each time-series in order to obtain better prediction results. A possible approach could be the application of a sophisticated preprocessing framework based on the intrinsic time-series specific properties such as stationarity, heteroskedasticity, seasonal cycles and changing variance, for performing that a priori identification and the proper time-series transformation methodology.

8 Conclusions and future research

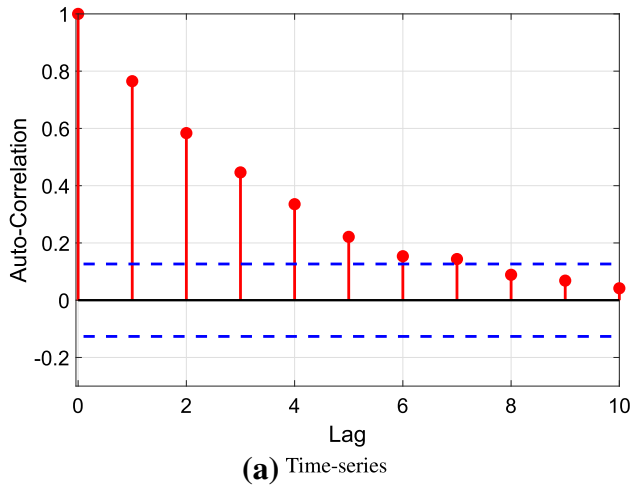
Time-series forecasting and analysis is generally considered as one of the most challenging problems in data mining. In the literature, most time-series forecasting approaches attempt to exploit machine learning and deep learning algorithms, aiming at obtaining better performance compared to the already existing or proposed models. Nevertheless, they cannot guarantee to develop reliable forecasting models.

In this work, we propose a different approach and introduce a novel methodology for the development of efficient and reliable deep learning prediction models. The major novelty of our proposed framework is that it guarantees the forecasting reliability of the deep learning model's predictions, independent of the used time-series data. This is achieved by applying a series of transformations, which ensure that a time-series satisfies the stationarity property and it is suitable for fitting a deep learning model. In addition to the theoretical advantages of the proposed framework, we provided empirical evidence about its efficiency and robustness. More specifically, we performed a series of numerical experiments using time-series from three application domains, which attracted most of research interest, namely financial stock market, energy sector and cryptocurrency area. All compared models where evaluated on both forecasting time-series price (regression) and time-series directional movements (classification) as well as on the reliability of their forecasts by examining the existence of autocorrelation of the errors. Our comprehensive experimental analysis illustrated that our proposed methodology considerably improved the forecasting performance of a deep learning model, in terms of accuracy and reliability.

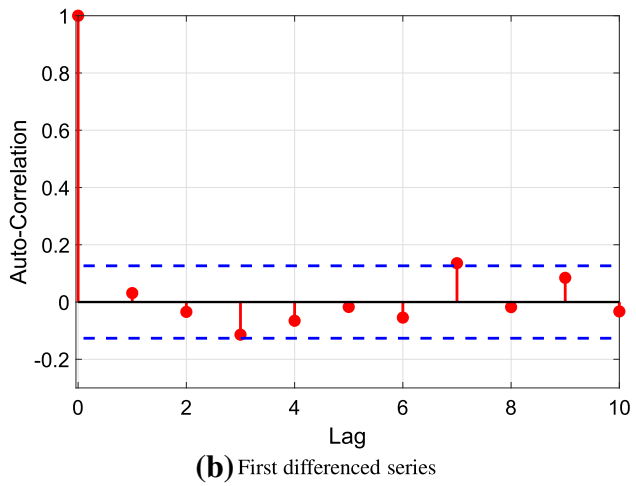
By taking into consideration that our proposed framework can be easily exploit any deep learning model, a prediction model exhibiting even better forecasting ability could be developed through the exploitation of deep learning techniques together with regularization methodologies or through additional optimized configuration of the utilized models.

It is worth mentioning that the introduced framework can be easily extended to cover the wider scientific area of time series forecasting applications such as weather forecasting, earthquake prediction, heartbeat rate and so on, without the requirement of any extra modifications or additional constraints. Furthermore, one issue which we

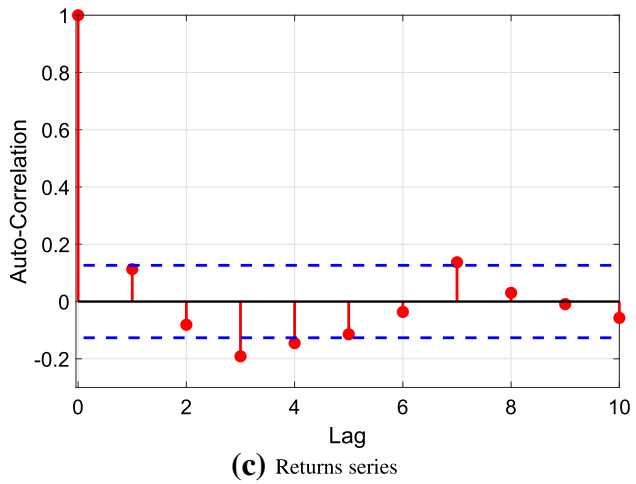
Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK



(a) Time-series

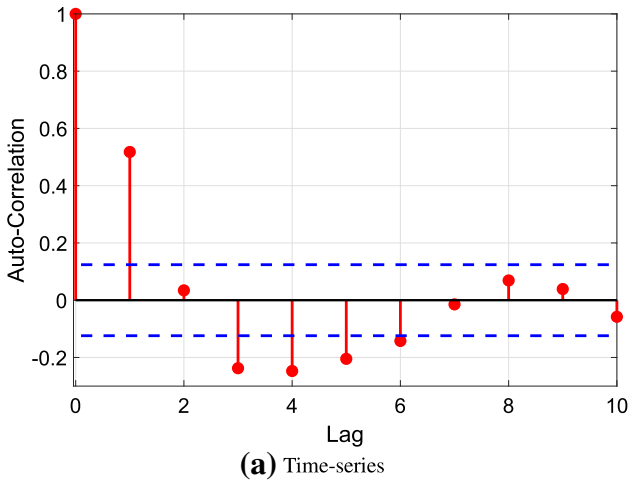


(b) First differenced series

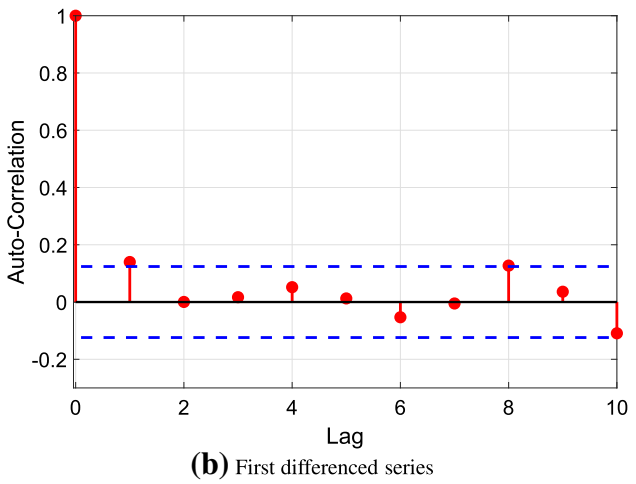


(c) Returns series

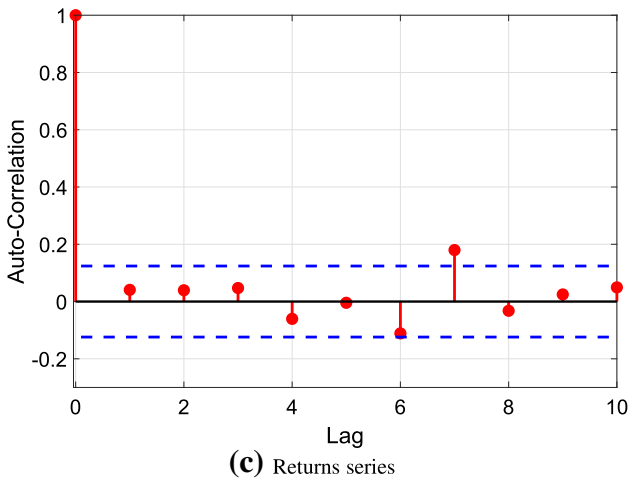
Fig. 4 Autocorrelation of residuals for S&P500 dataset of LSTM model



(a) Time-series



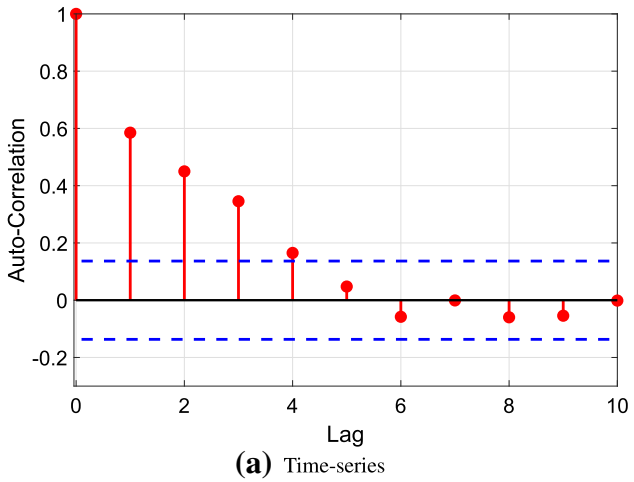
(b) First differenced series



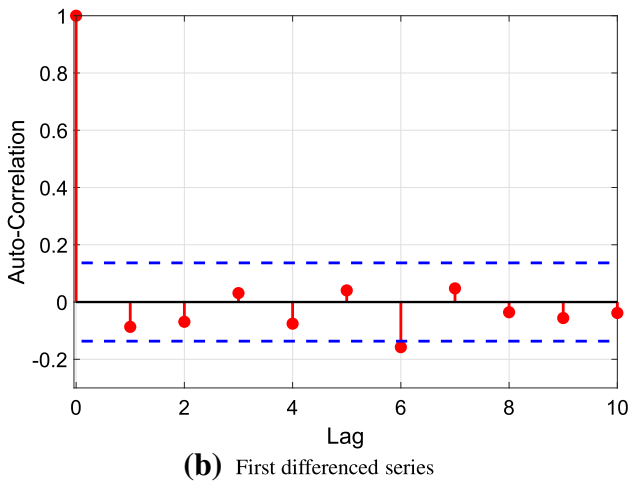
(c) Returns series

Fig. 5 Autocorrelation of residuals for Brent dataset of LSTM model

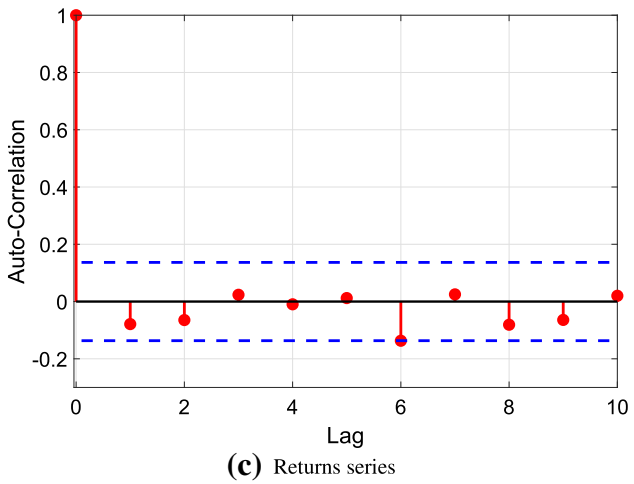
Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK



(a) Time-series

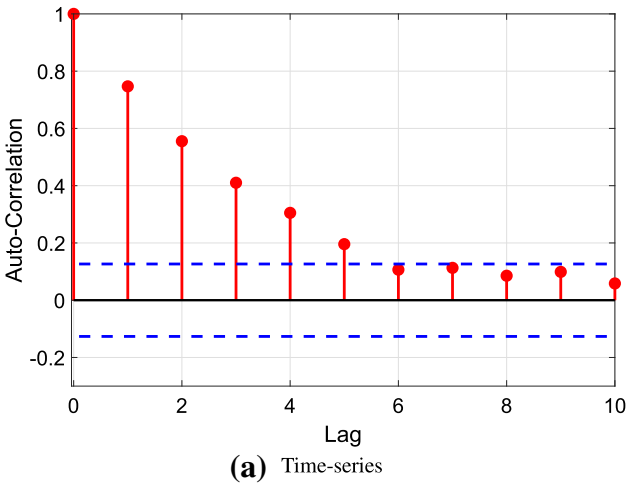


(b) First differenced series

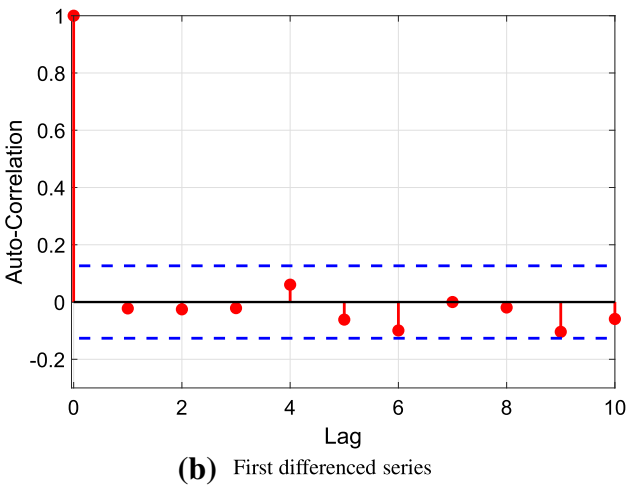


(c) Returns series

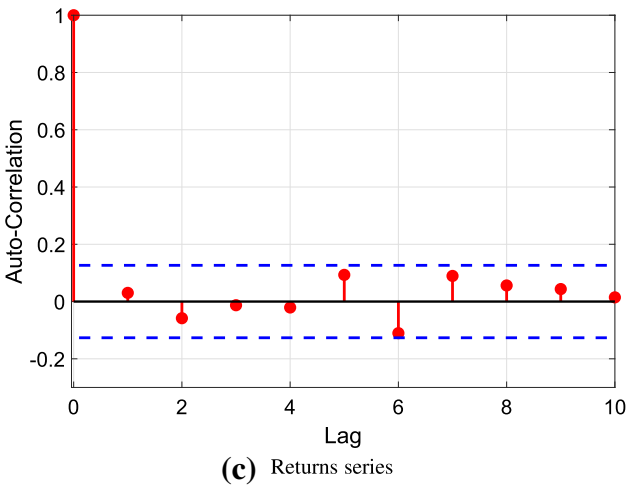
Fig. 6 Autocorrelation of residuals for BTC dataset of LSTM model



(a) Time-series



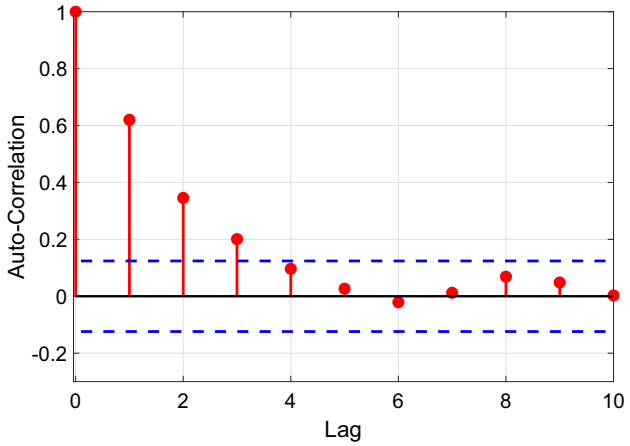
(b) First differenced series



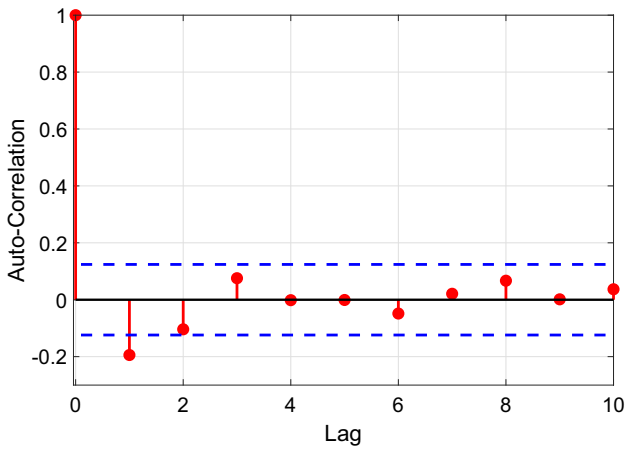
(c) Returns series

Fig. 7 Autocorrelation of residuals for S&P500 dataset of CNN–LSTM model

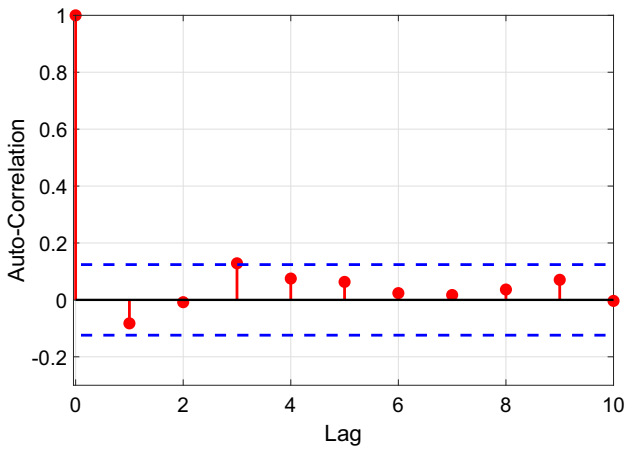
Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK



(a) Time-series

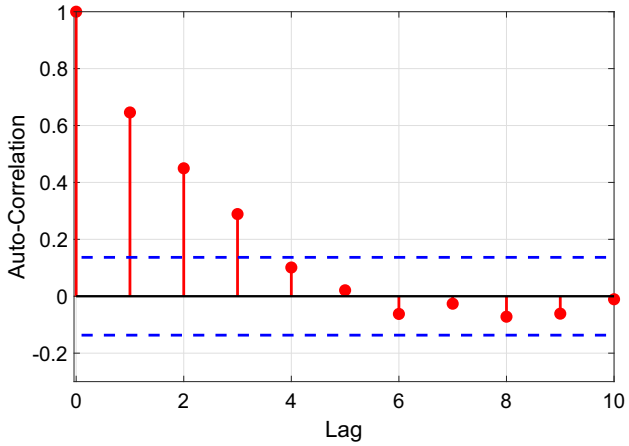


(b) First differenced series

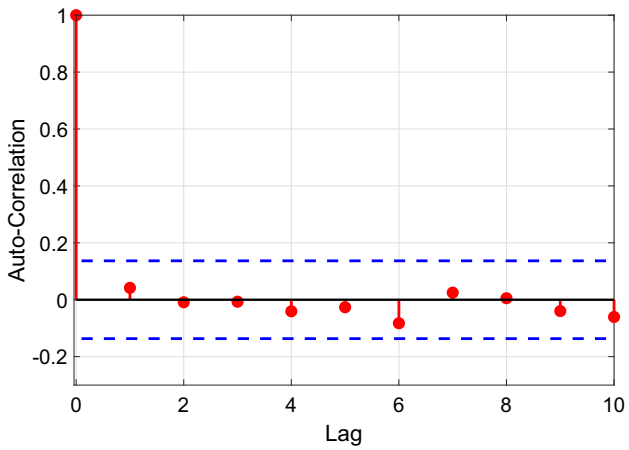


(c) Returns series

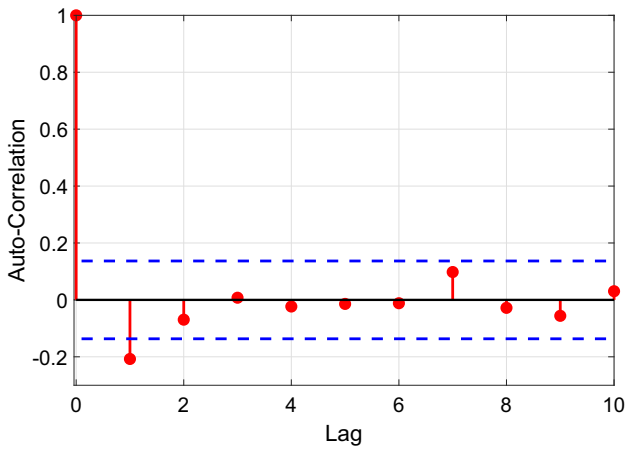
Fig. 8 Autocorrelation of residuals for Brent dataset of CNN–LSTM model



(a) Time-series



(b) First differenced series



(c) Returns series

Fig. 9 Autocorrelation of residuals for BTC dataset of CNN–LSTM model

Journal : Large 521 Dispatch : 7-7-2020 Pages : 19
Article No. : 5169 LE TYPESET
MS Code : NCAA-D-20-01027R1 CP DISK

have not thoroughly investigated is the possibility of some minor information loss due to non-stationarity in the time-series and the imposition of the proposed transformations. This is to be included and fully investigated in our future research. Furthermore, we intend to verify that the proposed framework works with any kind of regression algorithm.

Another direction for future research is to enhance our experimental framework with new performance metrics based on profits and returns. Finally, an interesting idea is the application of our proposed framework for the prediction of anomaly detection in order to "catch" outliers or other rare signals, which could indicate forecasting instability.

Compliance with ethical standards

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

References

1. Ahmed NK, Atiya AF, Gayar NE, El-Shishiny H (2010) An empirical comparison of machine learning models for time series forecasting. Econom Rev 29(5–6):594–621
2. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
3. Bontempi G, Taieb SB, Le Borgne Y (2012) Machine learning strategies for time series forecasting. In: European business intelligence summer school. Springer, Berlin, pp 62–77
4. Bougoudis I, Demertzis K, Iliadis L (2016) HISYCOL a hybrid computational intelligence system for combined machine learning: the case of air pollution modeling in Athens. Neural Comput Appl 27(5):1191–1206
5. Box GEP, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis: forecasting and control. Wiley, London
6. Brockwell PJ, Davis RA (2016) Introduction to time series and forecasting. Springer, Berlin
7. Burke S, Hunter J (2005) Modelling non-stationary economic time series: a multivariate approach. Springer, Berlin
8. Cen Z, Wang J (2019) Crude oil price prediction model with long short term memory deep learning based on prior knowledge data transfer. Energy 169:160–171
9. Chambon S, Galtier MN, Arnal PJ, Wainrib G, Gramfort A (2018) A deep learning architecture for temporal sleep stage classification using multivariate and multimodal time series. IEEE Trans Neural Syst Rehabil Eng 26(4):758–769
10. Donate JP, Li X, Sánchez GG, de Miguel AS (2013) Time series forecasting by evolving artificial neural networks with genetic algorithms, differential evolution and estimation of distribution algorithm. Neural Comput Appl 22(1):11–20
11. Fawaz HI, Forestier G, Weber J, Idoumghar L, Muller P (2019) Deep learning for time series classification: a review. Data Min Knowl Disc 33(4):917–963
12. Fischer T, Krauss C (2018) Deep learning with long short-term memory networks for financial market predictions. Eur J Oper Res 270(2):654–669
13. Fu T (2011) A review on time series data mining. Eng Appl Artif Intell 24(1):164–181
14. Gocheva-Ilieva SG, Voynikova DS, Stoimenova MP, Ivanov AV, Iliev IP (2019) Regression trees modeling of time series for air pollution analysis and forecasting. Neural Comput Appl 31(12):9023–9039
15. Gulli A, Pal S (2017) Deep learning with Keras. Packt Publishing Ltd
16. Ji S, Kim J, Im H (2019) A comparative study of Bitcoin price prediction using deep learning. Mathematics 7(10):898
17. Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: 2015 International conference on learning representations
18. Liu S, Zhang C, Ma J (2017) CNN-LSTM neural network model for quantitative strategy analysis in stock markets. In: International conference on neural information processing. Springer, Berlin, pp 198–206
19. 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: 16th International conference on artificial intelligence applications and innovations
20. Livieris IE, Pintelas E, Pintelas P (2020) A CNN–LSTM model for gold price time series forecasting. In: Neural computing and applications
21. Nakano M, Takahashi A, Takahashi S (2018) Bitcoin technical trading with artificial neural network. Physica A 510:587–609
22. Osborne J (2010) Improving your data transformations: applying the Box-Cox transformation. Pract Assess Res Eval 15(1):12
23. Pal A, Prakash PKS (2017) Practical time series analysis: master time series data processing, visualization, and modeling using python. Packt Publishing Ltd
24. Pintelas E, Livieris IE, Stavroyiannis S, Kotsilieris T, Pintelas P (2020) Fundamental research questions and proposals on predicting cryptocurrency prices using DNNs. Technical Report TR20-01, University of Patras, https://nemertes.lis.upatras.gr/jspui/bitstream/10889/13296/1/TR01-20.pdf
25. Pintelas E, Livieris IE, Stavroyiannis S, Kotsilieris T, Pintelas P (2020) Investigating the problem of cryptocurrency price prediction: a deep learning approach. In: 16th International conference on artificial intelligence applications and innovations
26. Shaman P (2010) Generalized Levinson–Durbin sequences, binomial coefficients and autoregressive estimation. J Multivar Anal 101(5):1263–1273
27. Stavroyiannis S (2019) Can Bitcoin diversify significantly a portfolio? Int J Econ Bus Res 18(4):399–411
28. Tanaka K (2017) Time series analysis: nonstationary and noninvertible distribution theory, vol 4. Wiley, London
29. Urtnasan E, Park J, Lee K (2018) Automatic detection of sleep-disordered breathing events using recurrent neural networks from an electrocardiogram signal. In: Neural computing and applications, pp 1–10
30. Weigend AS (2018) Time series prediction: forecasting the future and understanding the past. Routledge, London
31. Xingjian SHI, Chen Z, Wang H, Yeung D, Wong W, Woo W (2015) Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Advances in neural information processing systems, pp 802–810
32. Zhao Y, Li J, Yu L (2017) A deep learning ensemble approach for crude oil price forecasting. Energy Econ 66:9–16

Journal : Large 521 | Dispatch : 7-7-2020 | Pages : 19
Article No. : 5169 | ☐ LE | ☐ TYPESET
MS Code : NCAA-D-20-01027R1 | ☑ CP | ☑ DISK