Poznań University of Economics and Business
Research Papers in Economics and Finance *Vol. 8, No. 1, 2024: 91–114*
https://ref.ue.poznan.pl *https://doi.org/10.18559/ref.2024.1.1172*

Forecasting foreign exchange rate volatility using deep learning: Case of US dollar/Algerian dinar during the COVID-19 pandemic

Meryem-Nadjat Naas[1] Habib Zouaoui[2]

Abstract

This study explores the application of deep learning techniques in forecasting foreign exchange rate volatility, leveraging the capabilities of neural networks to capture complex patterns and non-linear relationships within financial data. The volatility of exchange rates is a critical factor influencing investment decisions, risk management and financial market stability. Traditional models often struggle to capture the dynamic nature of market conditions, leading to increased interest in advanced machine learning methodologies. We applied the auto regressive integrated moving average (ARIMA) and machine learning linear regression (LR) model, deep learning models, i.e. recurrent neural networks (RNN), bidirectional LSTM (BiLSTM), long short-term memory (LSTM) and gated recurrent unit (GRU). In terms of forecasting errors, Python routines were used for such a purpose. Furthermore, in order to investigate the quality of the models used, we compared the performances of these algorithms in US dollar/Algerian dinar exchange rate forecasting through the application of significance statistical tests (*R*-squared, MSE, RMSE, MAE, MAPE).The results clearly depict that contemporary techniques have been shown to produce more accurate results than conventional regression-based modelling. The machine learning linear regression (LR) model provides the maximum accuracy rate (99.83%), followed by the RNN models, with the GRU model (92.27%), BiLSTM model (87.34%), LSTM model (74.68%) and ARIMA model (32.29%).

Keywords

- deep learning
- exchange rate
- forecasting
- ARIMA
- linear regression

Article received 11 February 2024, accepted 19 April 2024.

Suggested citation: Naas, M. N., & Zouaoui, H. (2024). Forecasting foreign exchange rate volatility using deep learning: Case of US dollar/Algerian dinar during the COVID-19 pandemic. *Research Papers in Economics and Finance*, *8*(1), 91–114. https://doi.org/10.18559/ref.2024.1.1172

[1] University of Relizane, B.P:48000, Algeria, corresponding author: meryemnadjat.naas@univ-relizane.dz

[2] University of Relizane, B.P:48000, Algeria, habib.zouaoui@univ-relizane.dz

Introduction

Forecasting foreign exchange (forex) rate volatility is a complex task due to the dynamic and non-linear nature of financial markets. It has become an important subject in economics, business and finance (Bai et al., 2018). The foreign exchange market is a global financial market that is influenced by economic, political and psychological factors which are interconnected in complex ways. This complexity makes the foreign exchange market a difficult time-series prediction (Cappiello et al., 2006). At the end of 2019, the world was faced with the COVID-19 pandemic that didn't only affect public health but also the foreign exchange market, which influenced the trading behaviour (Zahrah, 2020). However, financial market prediction is a classic research problem in quantitative finance and the research area of neural networks. Financial time series often present characteristics such as volatility, non-stationarity, periodicity, non-linearity and long-term dependence (Huyghebaert & Wang, 2010). Traditional statistical models, including the multiple regression method, the autoregressive integrated moving average (ARIMA) model and the Generalized AutoRegressive Conditional Heteroskedasticity (GARCH) model Engle (2002), can accurately capture the volatility and periodicity of financial series and are widely used in the real world. The statistical models are also highly interpretable, but it is difficult for them to analyse non-stationary sequences and capture the non-linear relationships of financial time series (Zhang et al., 2019).

Deep learning (DL) is an advanced technique of machine learning (ML) based on artificial neural network (NN) algorithms. As a promising branch of artificial intelligence, DL has attracted great attention in recent years. Compared with conventional ML techniques such as a support vector machine (SVM) and k-nearest neighbours (kNN), DL possesses advantages of the unsupervised feature learning, a strong capability of generalisation and a robust training power for big data. Currently, DL has been applied comprehensively in classification and prediction tasks, computer visions, image processing and audio-visual recognition (Chai & Li, 2019). Although DL was developed in the field of computer science, its applications have penetrated diversified fields such as medicine, neuroscience, physics and astronomy, finance and banking (F&B), as well as operations management (Chai & Ngai, 2020). The existing literature lacks a good overview of DL applications in F&B fields. The aim of this research is to investigate whether or not the machine learning models offer better predictions than traditional models in terms of lower forecasting errors and higher accuracy in forecasts. More specifically, we will compare the autoregressive integrated moving average (ARIMA) and machine learning linear regression (LR) model, deep learning models, i.e. recurrent neural networks (RNN), bidirectional LSTM (BiLSTM), long short-term memory (LSTM) and gated recurrent unit (GRU).Therefore, the research question is the following: which of the two models, i.e. conventional regression-based modelling (the ARIMA model) or deep learning models, perform better when forecasting future values of the Algerian dinar (DZD) exchange rate?

Generally, our study will try to test the following hypotheses:

H1: The ARIMA model provides the USD/NZD exchange rate forecasting with higher errors and lower accuracy.
H2: The deep learning models provides the USD/NZD exchange rate forecasting with lower errors and higher accuracy.

1. Background of the study

In order to better understand the issue under analysis, we present some recent studies on our research topic. Table 1 depicts the number of articles that use various DL models in exchange rate forecasting.

Table 1. Reviewed previous studies

Author(s)/Year	Country	Period	Methodology	Main findings
Aygün & Günay Kabakçı (2021)	USA	September 2013 – October 2020	DNN, RNN, CNN, MVA, ARIMA	the RNN yields better results
Larasati & Primandari (2021)	USA	4 August 2018 – 21 January 2020	RNN, LSTM	the best accuracy of the LSTM
Robinson & Kabari (2021)	Nigeria	December 2001 – September 2019	RNN, SVM, LSTM, GRU, ARIMA	the RNN provides the best accuracy
Kaushik (2020)	India	April 1994 – December 2018	RNN, SVM, LSTM, VAR	the best accuracy of the LSTM
Yasar & Kilimci (2020)	Turkey	1 January 2018 – 31 December 2018	CNN, RNN, LSTM, FSA, ARIMA	the best accuracy of Holt-Winters and ARIMA
Chen et al. (2020)	USA	August 2011 – July 2018	ANN, SVM, RF, LSTM, ARIMA	the best accuracy of the LSTM
Siami-Namini & Siami Namin (2019)	USA	January 1985 – August 2018	ARIMA, RNN, LSTM, BiLSTM	the best accuracy of the BiLSTM
McNally et al. (2018)	USA	August 2013 – July 2016	ARIMA, RNN, LSTM	the best accuracy of the LSTM

Source: authors' analysis based on literature review.

Several studies have examined the impact of COVID-19 on the foreign exchange market (e.g. Aslam et al., 2020; Umar & Gubareva, 2020). These studies applied traditional regression techniques to investigate the volatility of foreign exchange markets. However, these studies failed to consider the predictability of exchange rates during the COVID-19 and non-COVID-19 periods. Our study attempts to apply the best machine learning and deep learning algorithms to predict the foreign currency exchange rates during the COVID-19 pandemic and compare them with the rates during the normal non-COVID-19 period. It is important to predict the exchange rate accurately because it helps policymakers and businesspeople to improve the quality and quantity of appropriate management decisions and plan their finances more precisely (Fang & Bessler, 2018). Different methods are used to predict the foreign currency exchange rate (Mahmoud & Hosseini, 1994; Maya & Gómez, 2008; Windsor & Thyagaraja, 2001); most of them have been based on statistical analysis (Abedin et al., 2021).

2. Materials and methods

2.1. Autoregressive integrated moving average (ARIMA) model

The autoregressive integrated moving average (ARIMA) model is one of the time series forecasting methods which says that the current value of a variable can be explained in terms of two factors: a combination of lagged values of the same variable and a combination of a constant term plus a moving average of past error terms (Aloui & Hkiri, 2014).

As seen in **Equation 2.1**, the auto-regressive (AR) model expresses the time series x_t at time t as a linear regression of the previous p observations, that is:

$$x_t = \alpha + \sum_{i=1}^{p} \varphi_i x_{t-i} + \varepsilon_t \tag{2.1}$$

where ε_t is the white noise residual term and φ_i are real parameters.

In **Equation 2.2**, the Moving averages (MA)use dependency between residual errors to forecast values in the next period. The model helps to adjust to unpredictable events. The q^{th} order moving average model, denoted by MA (q), is defined as follows:

$$x_t = \alpha - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{2.2}$$

where α and θ_1 are real parameters.

The ARMA model combines the power of AR and MA components together. This way, an ARMA (p, q) model incorporates the p^{th} order AR and q^{th} order MA model, respectively.

We denote the AR and MA coefficients vectors by φ and θ. The α and ε_t captures the intercept and the error term at time t. The complete ARMA (p, q) model can be seen in detail in **Equation 2.3**, that is:

$$x_t = \alpha + \varphi_1 x_{t-1} - \varphi_2 x_{t-2} + \cdots + \varphi_p x_{t-p} - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} \cdots - \theta_q \varepsilon_{t-q} + \varepsilon_t \tag{2.3}$$

ARIMAis a generalisation of the ARMA model by including integrated components, which are useful when data is non-stationary. The ARIMA applies differencing on time series to remove the non-stationarity. The ARIMA (p, d, q) represents the order for AR, MA and differencing components (Udom, 2018).

To forecast using a regression model with ARIMA errors, we need to forecast both the regression part of the model and the ARIMA part of the model, and combine the results. As with ordinary regression models, in order to obtain forecasts, we first need to forecast the predictors. When the predictors are known into the future (e.g. calendar-related variables such as time, day-of-week, etc.), this is straightforward. However, when the predictors are themselves unknown, we must either model them separately or use assumed future values for each predictor.

2.2. The machine learning linear regression (LR) model

Linear regression (LR) is a fundamental machine learning algorithm used for predicting a continuous outcome variable (also called the dependent variable) based on one or more predictor variables (independent variables). The basic idea behind linear regression is to find the linear relationship between the input features and the output variable.

The linear regression model assumes that the relationship between the independent variable X and the dependent variable Y.

There are two factors (x, y) involved in linear regression analysis. The equation below shows how *y* is related to *x*, known as regression:

$$y = \beta_0 + \beta_1 x + \varepsilon_t$$

Or equivalently:

$$E(y) = \beta_0 + \beta_1 x$$

Here, ε is the error term of linear regression. The error term accounts for the variability between both x and y, β_0 represents y-intercept, β_1 represents the slope. To put the concept of linear regression in the machine learning context in order to train the model, x is represented as input training dataset, y represents the class labels present in the input dataset. The goal of the machine learning algorithm then is to find the best values for β_0 (intercept) and β_1 (coefficient) to get the best-fit regression line. The best fit implies that the difference between the actual values and predicted values should be minimum. The minimisation problem can be represented as Grinsted et al. (2004).

$$Minimise \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

$$g = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

Here, g is called a cost function, which is the root mean square of the predicted value of $y(pred_i)$ and actual $y(y_i)$, n is the total number of data points.

In summary, linear regression is a simple and widely used algorithm for predicting continuous outcomes based on the linear relationship between input features and the target variable.

2.3. Deep learning (DL) models

This section is devoted to briefly describe the basic principle of four non-linear machine learning models or deep learning models that will be used later for the exchange rate timeseries forecasting, namely RNN, LSTM, GRU, BiLSTM (Zouaoui & Naas, 2023).

2.3.1. Recurrent neural networks (RNN)

Forecasting with recurrent neural networks (RNNs) is a common application in time series analysis, where the goal is to predict future values based on past observations. RNNs are particularly well-suited for sequential data due to their ability to capture temporal dependencies. Here isa general guide on how to use RNNs for forecasting:

1. **Data Preparation**
   - **Time Series Data:** Ensure your data is in the form of a time series, with each data point associated with a timestamp.
   - **Normalisation/Standardisation:** Scale your data to a small range, typically between 0 and 1, to help the neural network converge faster.
2. **Sequence Generation**
   - **Input Sequences:** Divide your time series data into input sequences. For instance, if your time series is [1, 2, 3, 4, 5, 6, 7, 8], you might create input sequences such as [1, 2, 3], [2, 3, 4], etc.
   - **Output Labels:** Correspondingly, create output labels for each input sequence, representing the next value in the sequence. For the examples above, the output labels would be [4], [5], etc.
3. **Model Architecture**
   - **Recurrent Layers:** Use RNN layers such as the LSTM (long short-term memory) or GRU (gated recurrent unit) in your model. These layers are designed to capture sequential dependencies.
   - **Stacking Layers:** You can stack multiple recurrent layers to capture more complex patterns. However, be cautious about overfitting.
   - **Dense Layers:** Add one or more dense layers at the end of your network to make the final predictions.
4. **Training**
   - **Loss Function:** Use a suitable loss function for regression tasks, such as the Mean Squared Error (MSE) or Mean Absolute Error (MAE).
   - **Optimizer:** Common optimisers include Adam, RMSprop or SGD.
   - **Validation Data:** Split your data into training and validation sets to monitor the model's performance and avoid overfitting.
5. **Hyperparameter Tuning**
   - **Learning Rate:** Experiment with different learning rates to find the optimal one for your specific problem.
   - **Batch size:** Adjust the batch size based on your available computational resources.
   - **Epochs:** Train the model for an appropriate number of epochs, monitoring validation performance.
6. **Evaluation**
   - **Test set:** After training, use a separate test set to evaluate the model's performance on unseen data.

- **Metrics:** Use appropriate metrics such as the Mean Absolute Error (MAE), Mean Squared Error (MSE) or others, depending on your specific requirements.

7. **Post-Processing**

   Inverse transformation: If you have normalised or standardised your data, apply the inverse transformation to get the predictions in the original scale.

8. **Fine-tuning**

   Depending on the results, you might need to fine-tune hyperparameters, adjust the model architecture or gather more data for better performance.

9. **Deployment**

   Once satisfied with the model's performance, deploy it for making real-time predictions.

Remember that the effectiveness of an RNN for forecasting depends on the nature of your data and the complexity of the patterns it contains. Experimentation and iteration are crucial in finding the best model for your specific use.

2.3.2. Long Short-Term Memory (LSTM) model

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to overcome some of the limitations of traditional RNNs in capturing and learning long-term dependencies in sequential data. LSTM networks (LSTMs) were introduced by Hochreiter and Schmidhuber (1997) (Zeroual et al., 2020).

Here are some key features of LSTM:

Memory cells: LSTMs have memory cells that can store information over long periods of time. These cells can be thought of as conveyor belts, allowing information to be added or removed selectively.

Gates: LSTMs use three gates to control the flow of information into and out of the memory cell.

Forget gate: Determines what information from the previous state should be discarded.

Input gate: Determines what new information from the current input should be stored in the memory cell.

Output gate: Determines what information from the memory cell should be output as the final prediction.

Cell state: LSTMs have a separate cell state that runs across the entire chain. This allows LSTMs to carry information over long sequences without being diluted or lost.

The equations governing the flow of information in an LSTM network are more complex than those of a standard RNN, but they allow LSTMs to address the vanishing and exploding gradient problems associated with traditional RNNs.

The ability of LSTMs to capture long-range dependencies in sequential data makes them well-suited for a variety of tasks, such as natural language processing, speech recognition and time-series prediction (Ketkar & Moolayil, 2021).

2.3.3. Gated recurrent unit (GRU) model

Gated Recurrent Unit (GRU) is another type of the recurrent neural network (RNN) architecture, introduced by Cho et al. (2014). Similar to LSTMs, GRUs are designed to address the vanishing gradient problem in traditional RNNs and capture long-term dependencies in sequential data. The GRU is considered a more simplified version of an LSTM with fewer parameters. A GRU uses the hidden layers to transfer information and calls its two gates the reset gate and the update gate (Zouaoui & Naas, 2023).

The parameters of the GRU include Wr, Wz and W_h. The reset signal r_t determines if the previous hidden state should be ignored, while the update signal z_t determines if the hidden state h_t should be updated with the new hidden state $hat(h_t)$.

$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right) + b_z$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right) + b_r$$

$$\hat{h}_t = \tanh\left(W_h \cdot [r_t \cdot h_{t-1}, x_t] + b_h\right)$$

$$h_h = (1 - z_t) \cdot h_{t-1} + z_t \cdot \hat{h}_t$$

Update gate (z_t): Controls the extent to which the information from the previous time step is passed along to the current time step. It decides what information to discard from the previous state.

Reset gate (r_t): Determines how much of the past information to forget. It decides which part of the previous state is relevant for computing anew candidate state.

Candidate state ($\hat{h}_t$): Similar to the memory cell in the LSTM, it represents new information that could be added to the hidden state.

Hidden state (h_t): Represents the output of the GRU at each time step. It is a combination of the previous hidden state and the new candidate state.

2.3.4. Bidirectional LSTM (BiLSTM)

Bidirectional long short-term memory (BiLSTM) is a type of the recurrent neural network (RNN) architecture that has proven effective in various sequence-related tasks, including time series forecasting. The bidirectional aspect allows the model to consider information from both past and future time steps, which can be beneficial in capturing complex patterns and dependencies in sequential data.

The idea of the BiLSTM comes from the bidirectional RNN (Schuster & Paliwal, 1997), which processes sequence data in both forward and backward directions with two separate LSTM hidden layers. It has been proved that the bidirectional networks are substantially better than the unidirectional ones in many fields (Cui et al., 2020).

The deep-bidirectional LSTMs are an extension of the described LSTM models, in which two LSTMs are applied to the input data. In the first round, an LSTM is applied on the input sequence (i.e. forward layer). In the second round, the reverse form of the input sequence is fed into the LSTM model (i.e. backward layer). Applying the LSTM twice leads to improving learning long-term dependencies, and thus it will improve the accuracy of the model [3] (Siami-Namini & Siami Namin, 2019).

2.4. Evaluation metrics

When evaluating forecasting models, it is important to use appropriate evaluation metrics to assess their performance. The choice of metrics depends on the nature of the forecasting problem and the specific goals of the forecasting task. Table 2 presents some commonly used evaluation metrics for forecasting.

Table 2. Performance evaluation metrics

Evaluation metric	Equation
Mean Squared Error (MSE)	$MSE = \frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2$
Root Mean Squared Error (RMSE)	$RMSE = \sqrt{\sum_{t=1}^{n}\frac{(y_t - \hat{y}_t)^2}{n}}$
Mean Absolute Error (MAE)	$MAE = \sum_{t=1}^{n}\frac{\lvert y_t - \hat{y}_t \rvert}{n}$

Table 2 continued

Evaluation metric	Equation
Mean Absolute Percentage Error (MAPE)	$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left\lvert \frac{y_t - \hat{y}_t}{y_t} \right\rvert \cdot 100$
R-squared	$R^2 = 1S \frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{n}(y_t - \mu)^2}$

Source: Korstanje (2021).

3. Results and analysis

3.1. Data description

The Algerian dinar exchange rate data(‘DZD’) from “2000-12-01” to “2020-12-31”, and was collected in daily time intervals. The total number of observations amounted to 4383 ($n = 4383$). The data were obtained from www.yahoo.finance.

It shall be noted that authors have just considered historical data from the previous three years in order to remove monotonic data from the initial US/DZD years. Since 2007, the DZD/USD price has become more volatile. The Algerian dinar price time series can be observed in Figure 1, where the linear trend and non-stationarity are highlighted as the first geometrical properties.

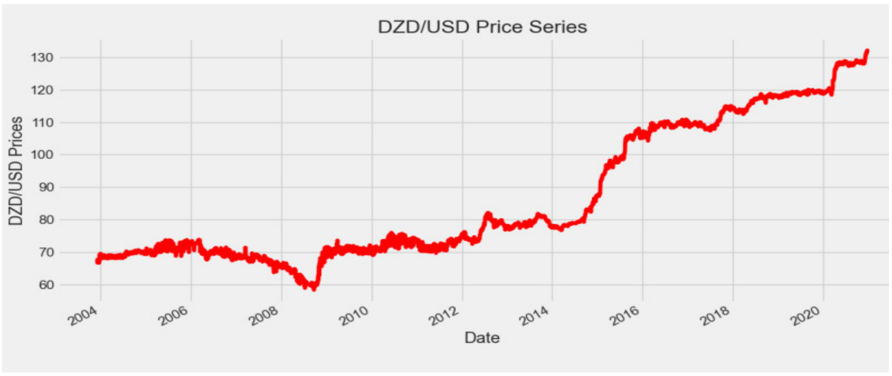

Figure 1. USD/DZD trend from 2004 to 2020

Source: based on Yahoo-finance datasets using Python code.

Despite a more favourable international context, marked by the rise in oil prices and the resumption of national and international economic activity, which was impacted by the COVID-19 pandemic in 2020, the depreciation of the Algerian dinar in 2021 accelerated against the US dollar and the euro, reflecting mainly the evolution of the American dollar against the euro. On annual average, the Algerian dinar further depreciated in 2021 by 6.1% against the US dollar and 9.3% against the euro, compared to respective depreciations of 5.9% and 7.7% in 2020 (Bank of Algeria, 2022).

The international economic context, marked by strong uncertainties and the persistence of twin deficits which still characterise the national economy, requires the deployment of all economic policy instruments. Indeed, the exchange rate which plays a major role in restoring imbalances by acting on the balance of payments, production and prices must be aligned with its equilibrium value. The year 2021 was marked by domestic inflation increasingly higher than that recorded in partner countries. This gap, combined with the loss of value of the Algerian dinar against several currencies, brought the real effective exchange rate closer to its level of balance. However, efforts must be made on the government's side to consolidate public finances to deal with disequilibrium macroeconomics.

3.2. ARIMA Model Analysis

The autoregression integrated moving average (ARIMA) takes into consideration the assumption that the past or previous data can forecast the behaviour of present and/or future data under normal conditions. The name clearly defines the aspects of the model, which is based on autoregression, integration and a moving average. It uses lag observations to determine the difference in raw observations and calculate error. The main parameters of the ARIMA model are the lag order (P), the degree of differencing (D) and the order of the moving average (Q). Before simulating the model, a scatter plot of the dataset is used to visualise the data as shown in Figure 2. The scatter plot (lag plot) shows a positive linear correlation with some outliers and randomness. The scatter plot helps to visualise the data and to determine what type of model is suitable for the data. In this case, the scatter plot is linear, so the use of an autoregressive model will be a good choice for predictions. The Dickey-Fuller test (DFT) is used to test for stationarity of the data (Figure 3). The dataset is said to be stationary when certain statistical properties are constant and have an independent covariance. This is particularly important as most statistical analysis works best on stationary data. Also, stationary datasets are easier to model, especially in time series analysis. From the DFT test, it can be seen that the dataset is not stationary (non-stationary) by the upward

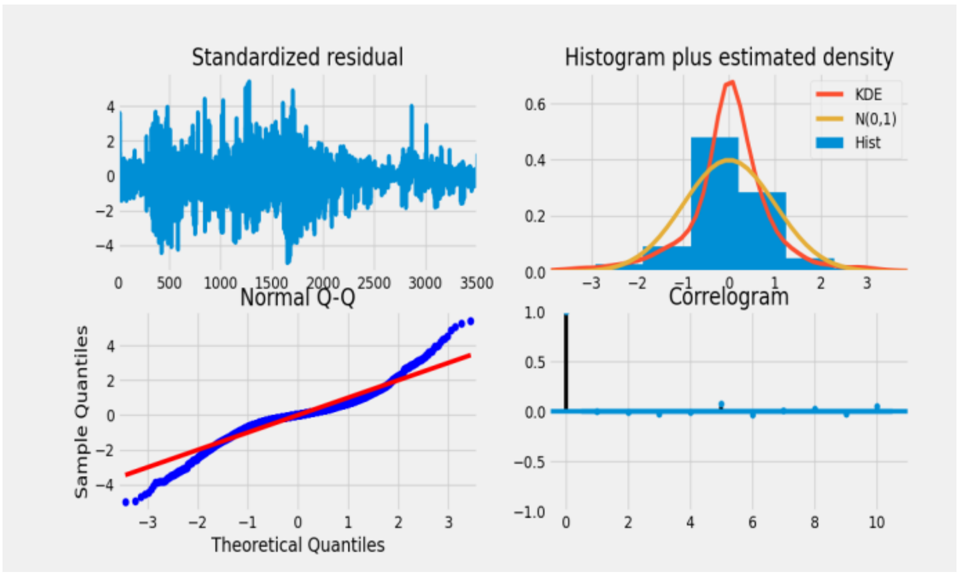

Figure 2. Auto ARIMA plot diagnostics

Source: based on Python code GitHub.

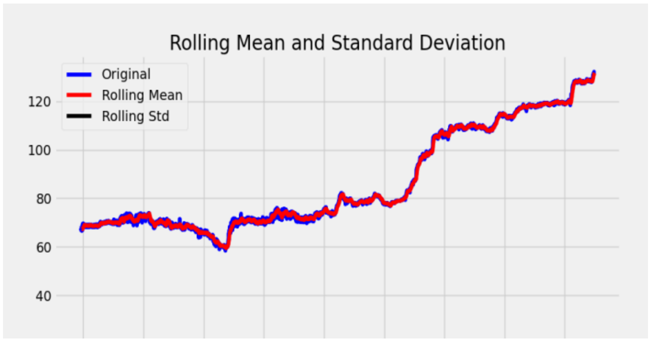

Figure 3. Dickey-Fuller test for stationarity showing upward and downward movement of rolling mean and standard deviation

Source: based on Python code GitHub.

and downward movement of the rolling mean and standard deviation as shown in Figure 3. Additionally, the static test is more than 5% of the critical value, and the p-value (0.49) is greater than 0.05. Thus, the Dickey-Fuller null hypothesis cannot be rejected. Before diving into the predictions, the dataset needs to be transformed into stationary data in order to have good results of the prediction. Non-stationary data can be converted to stationary data by applying decomposition and differencing techniques.

Decomposition is the separation of the trend and seasonality of the dataset, whereas differencing is finding the differences in the observations as shown in the code section of the ARIMA model. Once it is converted into stationary data, the auto ARIMA function is used to find the best values of p, d, and q (ARIMA hyperparameters code). The (p) parameter stands for the order of the autoregressive model, the (d) parameter stands for the order of differencing, and the (q) parameter stands for the order of the moving average. The (p) parameter uses past values in regression calculations, and the (d) parameter subtracts the previous and current values of (d). The (d) parameter is also used to convert the data from non-stationary to stationary. The (q) parameter defines the error of the model by a combination of previous errors. It determines the number of terms to be included in the model. After fitting the model, the optimal values of p, d, and q were (0, 1, 2), which can be written as Equation (3.1), where μ is the regressive parameter, $Yt - 1$ the differencing, and $\theta 1$ et − 1 the moving average.

The diagnostic plots are used to visualise the auto ARIMA model. Figure 4 shows four graphs. In the top left corner, standardised residuals fluctuate around a mean of zero. The top right graph displays the density plot which shows a normal distribution. The bottom left graph represents a correlation which shows a normal liner distribution following the red line.

Finally, the bottom right graph is the ACF (correlogram) which shows that the residuals are not autocorrelated. From the diagnosis, the auto ARIMA will fit the data perfectly. An ARIMA model is then created with the optimal parameters of p, d, and q (ARIMA model evaluation). The dataset is divided into a training set (used for training the model) and a test set (used for comparing and validating the predictions), as shown in Figure 4. It shall be noted that 80% of the data is used for the training set and 20% is used for the test dataset. The mean absolute percentage error (MAPE) is used to evaluate the model. At is the actual value and Pt is the

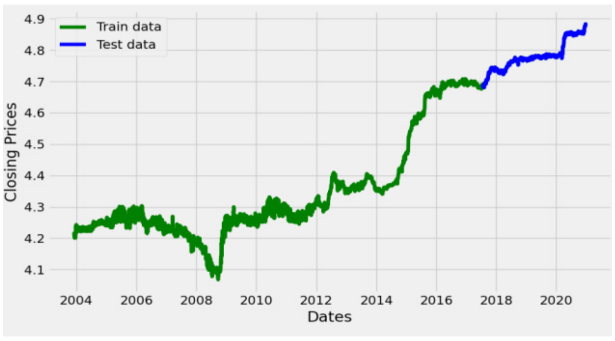

Figure 4. Training set and test dataset

Source: based on Python code GitHub.

predicted value. It is calculated by differencing the absolute of the actual value from the absolute of the predicted value divided by half the sum of the predicted and actual values. The result is summed for each fitted point (t) and divided by the number of fitted points (n). This equation evaluates the model by generating a positive and negative error while limiting the effect of outliers and bias. The prediction is done by using the predict function (which interprets the number of features it receives from the model and passes the number of features it receives to the output layer) of the future price range. The full code can be found on GitHub.

3.3. ARIMA forecasting results

To apply the ARIMA method, the time series data must be stationary. An augmented Dickey–Fuller test (ADF) is done to test whether the graph is stationary or not. Since p-value (0.588792) is > 0.5, the test has confirmed that the dataset is non-stationary. Before implementation, differencing in which seasonality and trend are eliminated to make data stationary. The original dataset and differenced dataset stationary results are presented in Figure 5 and Table 3.

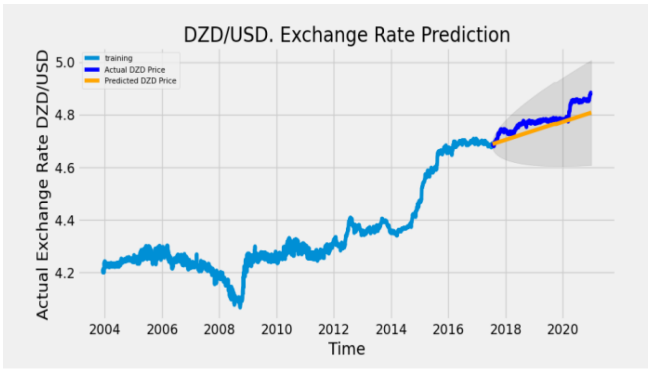

Figure 5. ARIMA (0,1,2) forecast graph

Source: based on Python code GitHub.

Our primary concern is to ensure that the residuals of our model are uncorrelated and normally distributed with zero-mean. If the seasonal ARIMA model does not satisfy these properties, it is a good indication that it can be further improved.

In this case, our model diagnostics suggest that the model residuals are normally distributed based on the following:

– In the top right plot, we see that the red KDE line follows closely the $N(0,1)$ line, where $N(0,1)$ is the standard notation for a normal distribution with a mean of

0 and standard deviation of 1. This is a good indication that the residuals are normally distributed.

- The Q-Qplot on the bottom left shows that the ordered distribution of residuals (blue dots) follows the linear trend of the samples taken from a standard normal distribution with $N(0,1)$. Again, this is a strong indication that the residuals are normally distributed.
- The residuals over time (top left plot) do not display any obvious seasonality and appear to be white noise. This is confirmed by the autocorrelation (i.e. correlogram) plot on the bottom right, which shows that the time series residuals have low correlation with lagged versions of themselves.

Those observations lead us to conclude that our model produces a satisfactory fit that could help us understand our time series data and forecast future values.

Although we have a satisfactory fit, some parameters of our seasonal ARIMA model could be changed to improve our model fit. For example, our grid search only considered a restricted set of parameter combinations, so we may find better models if we widen the grid search.

3.4. Forecasting performance of benchmark models

We train the neural network with the following hyperparameters (see Table 3):

- four hidden fully connected layers,
- each layer has 200 neurons,
- batch size of 64,
- 200 training epochs,
- 80–20 train-validation split,
- MSE as loss function.

Table 3. Hyperparameter of each model

Hyperparameter	LSTM, BiLSTM	GRU
Activation function	RELU	RELU
Loss function	MSE	MSE
Neurons	[200,200,200,200,200,1]	[200,200,200,200,200,1]
Learning rate	0.001	0.001
Optimiser	Adam	Adam

Note: Here [200, 200, 200, 200, 200, 1] represents the number of neurons from the first to the last network layer.

Source: authors based on Python code GitHub.

A special feature of a deep learning algorithm is that it can perform feature selection by itself and scale the data as required (Mathew et al., 2020). In this work, we have presented the actual versus predicted exchange rates with automatic scaled values generated by our proposed deep learning approach over time (see Appendices N:01). Table 4 presents the MSE, MAE, MAPE, RMSE and R-squared of all techniques.

Table 4. Comparison of the best model

Model	Parameters	MSE	MAE	MAPE	RMSE	R-squared
LR	test = 20% train = 80%	0.1602	0.2502	0.0090	0.3477	99.42
GRU	test = 20% train = 80%	1.3038	1.2338	0.0105	1.3906	92.27
BiLSTM	test = 20% train = 80%	3.4531	1.7924	0.0149	1.8583	87.34
LSTM	test = 20% train = 80%	6.9083	2.4146	0.0199	2.6284	74.68
ARIMA	order = (0,1,2) AIC= –27190.878	7.0445	0.0330	0.0068	0.0376	32.29

Source: authors based on Python code GitHub.

The experimental results revealed that the LR model is the best among the four methods. In terms of forecasting the performance measures, the MAPE is 0.009, MAE is 0.2502, RMSE is 0.3477 and R-squared is 0.9942, which is the lowest among the four forecasting models. Therefore, the LR model is superior to the other four models under analysis in terms of DZD/USD forecasting.

The preliminary results indicate promising outcomes, suggesting that deep learning models exhibit enhanced predictive capabilities in capturing the intricate volatility patterns inherent in foreign exchange markets. The findings contribute to the growing body of literature on the application of artificial intelligence in finance and offer insights into the potential improvements in the accuracy of volatility forecasting achievable through deep learning methodologies. As financial markets continue to evolve, embracing innovative technologies like deep learning becomes imperative for refining risk assessment and decision-making processes in the realm of foreign exchange trading.

Conclusion

In this study, we have focused on forecasting the USD/NZD exchange rate volatility during the period of the COVID-19 pandemic by proposing an ensemble of a deep learning approach and time series analysis using the ARIMA model.

We have applied other machine learning algorithms such as RNN and LR, as well as deep learning algorithms such as LSTM, GRU and BiLSTM.

Before the model was verified, we first analysed whether the data had periodicity, and the current time data were affected by past and future data. The data was then normalised and the criteria for evaluating the validity of the model were developed.

We confirmed the second hypothesis of study that DL models still performed well. We compared it with econometric existing prediction model in terms of the RMSE, MAE, MSE, MAPE and *R*-score.

The highly competitive prediction capacity of the proposed model during the COVID-19 period is beneficial for policymakers, entrepreneurs and foreign exchange brokers. Therefore, we can better understand the specific volatility in Algeria's foreign exchange rate during the COVID-19 pandemic. It is essential to analyse data such as exchange rate movements, central bank interventions and economic indicators like GDP growth and inflation, as these factors can provide insights into how the pandemic affected Algeria's currency market and the measures taken by authorities to stabilise it.

Appendices N:01

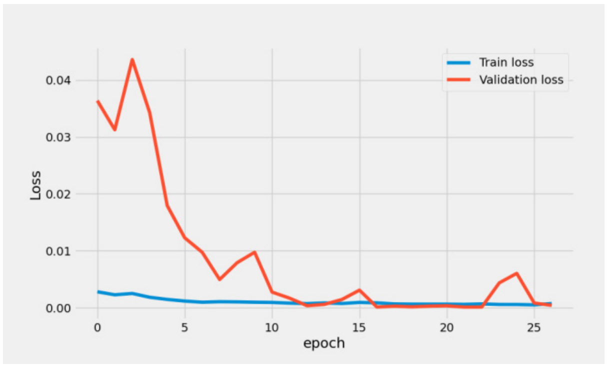

Figure 6. Train and test loss of LSTM model

Source: based on Python code GitHub.

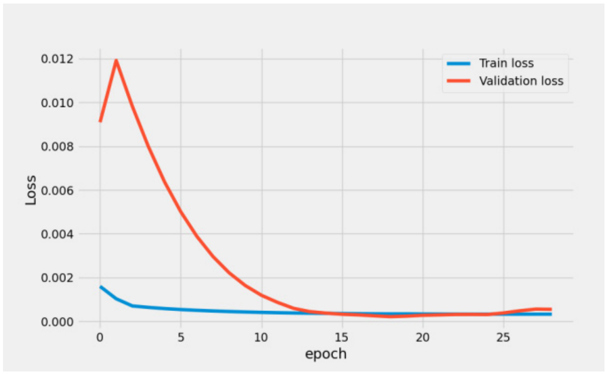

Figure 7. Train and test loss of BiLSTM model

Source: based on Python code GitHub.

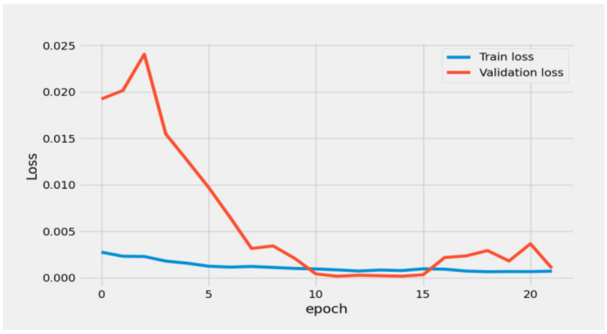

Figure 8. Train and test loss of GRU model

Source: based on Python code GitHub.

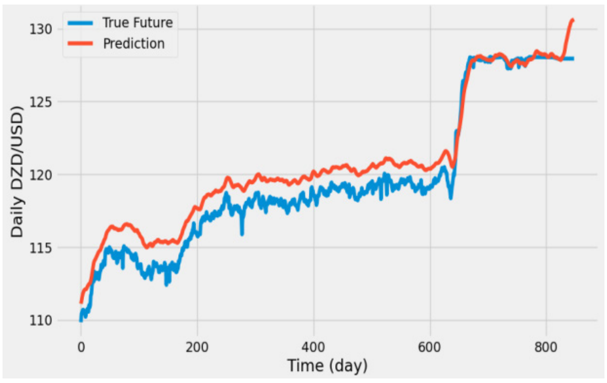

Figure 9. GRU forecast graph

Source: based on Python code GitHub.

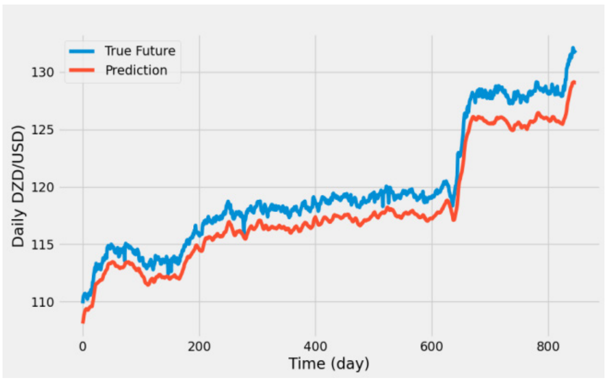

Figure 10. BiLSTM forecast graph

Source: based on Python code GitHub.

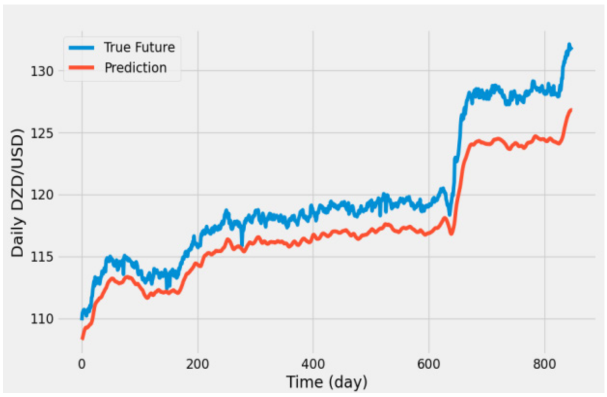

Figure 11. LSTM forecast graph

Source: based on Python code GitHub.

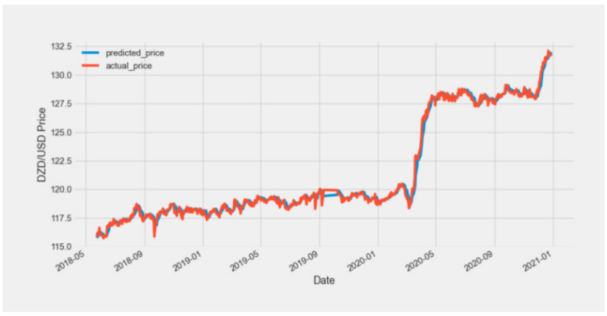

Figure 12. Linear regression forecast graph

Source: based on Python code GitHub.

Table 5. ARIMA model results

```
                              ARIMA Model Results
==============================================================================
Dep. Variable:                D.Close   No. Observations:                 3511
Model:                 ARIMA(0, 1, 2)   Log Likelihood               11623.501
Method:                       css-mle   S.D. of innovations              0.009
Date:                Tue, 30 Mar 2021   AIC                         -23239.002
Time:                        09:19:57   BIC                         -23214.348
Sample:                             1   HQIC                        -23230.205

=================================================================================
                    coef    std err          z      P>|z|      [0.025      0.975]
---------------------------------------------------------------------------------
const             0.0001   5.77e-05      2.356      0.018    2.29e-05       0.000
ma.L1.D.Close    -0.5234      0.017    -31.232      0.000      -0.556      -0.491
ma.L2.D.Close    -0.0895      0.016     -5.514      0.000      -0.121      -0.058
                                    Roots
=============================================================================
                  Real          Imaginary           Modulus         Frequency
-----------------------------------------------------------------------------
MA.1            1.5171           +0.0000j            1.5171            0.0000
MA.2           -7.3684           +0.0000j            7.3684            0.5000
-----------------------------------------------------------------------------
```

Source: based on Python code GitHub.

References

Abedin, M. Z., Moon, M. H., & Hassan, M.K. et al. (2021). Deep learning-based exchange rate prediction during the COVID-19 pandemic. *Annals of Operations Research*. https://doi.org/10.1007/s10479-021-04420-6

Aloui, C., & Hkiri, B. (2014). Co-movements of GCC emerging stock markets: New evidence from wave let coherence analysis. *Economic Modelling*, *36*, 421–431. https://doi.org/10.1016/j.econmod.2013.09.043

Aslam, F., Aziz, S., Nguyen, D. K., Mughal, K. S., & Khan, M. (2020). On the efficiency of foreign exchange markets in times of the COVID-19 pandemic. *Technological Forecasting and Social Change*, *161*, 120261. https://doi.org/10.1016/j.techfore.2020.120261

Aygün, B., & GünayKabakçı, E. (2021). Comparison of statistical and machine learning algorithms for forecasting daily bitcoin returns. *European Journal of Science and Technology*, *21*, 444–454. https://doi.org/10.31590/ejosat.822153

Bai, S., Cui, W., & Zhang, L. (2018). The Granger causality analysis of stocks based on clustering. *Cluster Computing*, *22*(12), 14311–14316. https://doi.org/10.1007/s10586-018-2290-0

Bank of Algeria (2022, December). *Raport annuel 2021. Evolution, Economique et Monetaire*. https://www.bank-of-algeria.dz/wp-content/uploads/2023/02/rapport-ba-2021fr-1.pdf

Cappiello, L., Engle, R. F., & Sheppard, K. (2006). Asymmetric dynamics in the correlations of global equity and bond returns. *Journal of Financial Econometrics*, *4*(4), 537–572.

Chai, J., & Li, A. (2019). Deep learning in natural language processing: A state-of-the-art survey. In: *International Conference on Machine Learning and Cybernetics* (ICMLC). (pp. 1–6). IEE. https://doi.org/10.1109/ICMLC48188.2019.8949185

Chen, W., Xu, H., Jia, L., & Gao, Y. (2020). Machine learning model for Bitcoin exchange rate prediction using economic and technology determinants. *International Journal of Forecasting*, *37*(1), 28–43. https://doi.org/10.1016/j.ijforecast.2020.02.008

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv*:1406.1078. https://doi.org/10.48550/arXiv.1406.1078

Cui, Z., Ke, R., Pu, Z., & Wang, Y. (2020). Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values. *Transportation Research Part C: Emerging Technologies*, *118*, 102674. https://doi.org/10.1016/j.trc.2020.102674

Engle, R. (2002). Dynamic conditional correlation: A simple class of multivariate generalized autoregressive conditional heteroskedasticity models. *Journal of Business & Economic Statistics*, *20*(3), 339–350.

Fang, L., & Bessler, D. (2018). Is it China that leads the Asian stock market contagion in 2015? *Applied Economics Letters*, *25*(11), 752–757. https://doi.org/10.1080/13504851.2017.1363854

Grinsted, A., Moore, J., & Jevrejeva, S. (2004). Application of the cross wavelet transform and wavelet coherence to geophysical time series. *Nonlinear Processes in Geophysics*, *11*(5/6), 561–566. https://doi.org/10.5194/npg-11-561-2004

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *8*(9), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Huyghebaert, N., & Wang, L. (2010). The co-movement of stock markets in East Asia: Did the 1997–1998 Asian financial crisis really strengthen stock market integration? *China Economic Review*, *21*(1), 98–112.

Kaushik, M. (2020). Forecasting foreign exchange rate: A multivariate comparative analysis between traditional econometric, contemporary machine learning & deep learning techniques. *arXiv*:2002.10247. https://doi.org/10.48550/arXiv.2002.10247

Ketkar, N., & Moolayil, J. (2021). *Deep learning with Python: Learn best practices of deep learning models with PyTorch* (2nd ed.). Apress.

Korstanje, J. (2021). *Advanced forecasting with Python: With state-of-the-art-models including LSTMs, Facebook's Prophet, and Amazon's DeepAR*. MaisonsAlfort.

Larasati, K. D., & Primandari, A. H. (2021). Forecasting Bitcoin price based on Blockchain information using long-short term method. *Parameter: Journal of Statistics*, *1*(1), 1–6. https://doi.org/10.22487/27765660.2021.v1.i1.15389

Mahmoud, E., & Hosseini, H. (1994). A comparison of forecasting techniques for predicting exchange rates. *Journal of Transnational Management Development*, *1*(1), 93–110. https://doi.org/10.1300/J130v01n01_07

Mathew, A., Amudha, P., & Sivakumari, S. (2021). Deep learning techniques: An overview. In: A. Hassanien, R. Bhatnagar, A. Darwish (Eds.), *Advanced machine learning technologies and applications. Proceedings of AMLTA 2020* (pp. 599–608). Springer. https://doi.org/10.1007/978-981-15-3383-9_54

Maya, C., & Gomez, K. (2008). What exactly is "bad news" in foreign exchange markets? Evidence from Latin American markets. *Cuadernos de Economía*, *45*(132), 161–183.

McNally, S., Roche, J., & Caton, S. (2018). Predicting the price of Bitcoin using machine learning. In: *26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)* (pp. 339–343). Cambridge, UK. https://doi.org/10.1109/PDP2018.2018.00060

Robinson, M., & Kabari, L. G. (2021). Predicting foreign exchange using digital signal processing. *British Journal of Computer, Networking and Information Technology*, *4*(2), 1–11. https://doi.org/10.52589/BJCNIT-SQWFNRND

Schuster, M., & Paliwal, K. K., (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, *45*(11), 2673–2681. https://doi.org/10.1109/78.650093

Siami-Namini, S., & Siami Namin, A. (2019). Forecasting Economics and Financial Time Series: ARIMA vs. LSTM. *arXiv*:1803.06386. https://doi.org/10.48550/arXiv.1803.06386

Udom, E. X. (2018). Estimating and forecasting Bitcoin daily returns using ARIMA-GARCH models. *International Journal of Science and Research*, *8*(10), 376–382.

Umar, Z., & Gubareva, M. (2020). A time – frequency analysis of the impact of the COVID-19 induced panic on the volatility of currency and cryptocurrency markets. *Journal of Behavioral and Experimental Finance*, *28*, 100404. https://doi.org/10.1016/j.jbef.2020.100404

Windsor, C., & Thyagaraja, A. (2001). The prediction of periods of high volatility in exchange markets. *The European Physical Journal B-Condensed Matter and Complex Systems*, *20*(4), 581–584. https://doi.org/10.1007/PL00011111

Yasar, H., & Kilimci, Z. H. (2020). US dollar/Turkish lira exchange rate forecasting model based on deep learning methodologies and time series analysis. *Symmetry*, *12*(9), 1553. https://doi.org/10.3390/sym12091553

Zahrah, H. H., Sa'adah, S., & Rismala, R. (2020). The foreign exchange rate prediction using long-short term memory: A case study in COVID-19 pandemic. *Journal on Information and Communication Technology*, *6*(2), 94–105. https://doi.org/10.21108/IJOICT.2020.62.538

Zeroual, A., Harrou, F., Dairi, A., & Sun, Y. (2020). Deep learning methods for forecasting COVID-19 time-Series data: A Comparative study. *Chaos, Solitons & Fractals*, *140*, 1–12. https://doi.org/10.1016/j.chaos.2020.110121

Zhang, X., Liang, X., Zhiyuli, A., Zhang, S., Xu, R., & Wu, B. (2019). AT-LSTM: An Attention-based LSTM Model for Financial Time Series Prediction. In: *IOP Conf. Series: Materials Science and Engineering*, 569, 052037, 1–7. https://doi.org/10.1088/1757-899X/569/5/052037

Zouaoui, H., & Naas, M .N. (2023). Option pricing using deep learning approach based on LSTM-GRU neural networks: Case of London stock exchange. *Data Science in Finance and Economics*, *3*(3), 267–284. https://doi.org/10.3934/DSFE.2023016