applied sciences

MDPI

Article

Recurrent Neural Networks and ARIMA Models for Euro/Dollar Exchange Rate Forecasting

Pedro Escudero [1,*], Willian Alcocer [2] and Jenny Paredes [2]

[1] SISAu Research Group, Facultad de Ingeniería y Tecnologías de la Información y la Comunicación, Universidad Tecnológica Indoamérica, Ambato 180103, Ecuador
[2] Escuela de Ingeniería en Estadística Informática, Facultad de Ciencias, Escuela Superior Politécnica de Chimborazo, Riobamba 060155, Ecuador; willianalcocer@espoch.edu.ec (W.A.); jeparedes@espoch.edu.ec (J.P.)
* Correspondence: pescudero2@indoamerica.edu.ec

Abstract: Analyzing the future behaviors of currency pairs represents a priority for governments, financial institutions, and investors, who use this type of analysis to understand the economic situation of a country and determine when to sell and buy goods or services from a particular location. Several models are used to forecast this type of time series with reasonable accuracy. However, due to the random behavior of these time series, achieving good forecasting performance represents a significant challenge. In this paper, we compare forecasting models to evaluate their accuracy in the short term using data on the EUR/USD exchange rate. For this purpose, we used three methods: Autoregressive Integrated Moving Average (ARIMA), Recurrent Neural Network (RNN) of the Elman type, and Long Short-Term Memory (LSTM). The analyzed period spanned from 2 January 1998, to 31 December 2019, and was divided into training and validation datasets. We performed forecasting calculations to predict windows with six different forecasting horizons. We found that the window of one month with 22 observations better matched the validation dataset in the short term compared to the other windows. Theil's U coefficients calculated for this window were 0.04743, 0.002625, and 0.001808 for the ARIMA, Elman, and LSTM networks, respectively. LSTM provided the best forecast in the short term, while Elman provided the best forecast in the long term.

Keywords: recurrent neuronal networks; ARIMA; random series modelling; EUR/USD exchange rate

Citation: Escudero, P.; Alcocer, W.; Paredes, J. Recurrent Neural Networks and ARIMA Models for Euro/Dollar Exchange Rate Forecasting. *Appl. Sci.* **2021**, *11*, 5658. https://doi.org/10.3390/app11125658

Academic Editors: Fco. Javier Gimeno-Blanes, Cristina Soguero-Ruiz, Margarita Rodríguez-Ibáñez, José Luis Rojo-Álvarez and Giancarlo Mauri

Received: 13 March 2021
Accepted: 2 June 2021
Published: 18 June 2021

1. Introduction

Different stock market forecasting techniques have been developed to predict values since the birth of the foreign exchange market (FOREX) in the 1970s [1–3]. Some of these techniques are used to identify future movements and include fundamental analysis, technical analysis, and mixed analysis, such as using statistical methods to model prices' behaviors and generate future predictions [4,5]. Individual models, such as ARIMA, have been employed to forecast time series due to their popularity as classic prediction methods [6]. Since the beginning of the 1990s, economic and financial data studies have been carried out by applying artificial neural networks (ANN) as estimation and forecasting methods in non-linear functions with great success [7,8]. For instance, a class of neural networks was designed mainly based on the use of Liapunov's stability theory for learning laws. These networks are known as differential or dynamic neural networks (DNN) [8,9], whose applications were shown to be successful for forecasting the DAX and S&P 500 stock indices [7]. Following the same logic of variable weight analysis, a neural network (NN) system with twelve economic variables was used to analyze the significance of these variables in peso–dollar forecasting [10]. In the last decade, RNNs of the LSTM type have been widely used for forecasting sequential data [11–14]. The mechanism by which such networks store long- and short-term information makes them powerful when performing historical data forecasting. This type of RNN has been used for currency-pair forecasting, action trading on the New York Stock Exchange, recognition, environmental predictions,

electricity demands, etc. [9]. The accuracy of the LSTM method was evaluated by comparing this method with other types of NNs and classical prediction methods [15–18]. Many of these comparisons and applications were used to formulate new hybrid models to improve the results of the predictions [19–21]. In this context, the results of a combination of classical prediction methods [19], neural networks [22], and recurrent neural networks [23] have helped clarify the path to creating new approximations based on standard methods applied to the currency exchange rate and stock market forecasting [24]. Most of these approaches were proposed to find the model that can provide the best forecasting in the short term according to the next-day predictions needed in currency exchange rate forecasting, which is the most challenging objective due to the inherently noisy and non-stationary behavior of the data.

In this work, we compare the modelling of ARIMA with RNN, Elman, and LSTM networks to perform out-of-sample forecasting for a EUR/USD exchange-rate dataset. The purpose of this work is to provide valuable tools not only to demonstrate the accuracy of these models and use them for financial purposes but also to show how these three methods can be used to create hybrid models to improve the forecasting of random time series. We begin this study by providing a summary of the three methods (ARIMA, Elman, and LSTM) to clarify how the algorithms work and how to optimize the models. Next, we define the datasets used for training and validation purposes, followed by exploratory analysis and data preprocessing. Then, we apply the ARIMA algorithm to determine the model that best forecasts the time series. We then define the Elman and LSTM networks by adjusting the optimal parameters. We performed training and validation of these three methods by using prediction windows with different forecast horizons. Finally, we chose the window that provides the best forecasting in the short term to evaluate, in detail, the accuracy of the three prediction methods.

2. Materials and Methods

2.1. Overview of Regression Techniques

The classical statistical methods were historically used to analyze the behavior of time series. ARIMA models are well-known parametric techniques trained through linear regressions [6,25]. The ARIMA algorithm in Figure 1a uses graphs, basic statistics, the Autocorrelation Function (ACF), the Partial Autocorrelation Function (PACF), and transformations to identify patterns and model components. This model provides estimates through least squares and maximum likelihood methods and uses the graphs, ACF, and PACF of residuals to verify the validity of the model—i.e., if the model is valid, then use that model; otherwise, go back to the first step. Lastly, the algorithm forecasts and tracks the model's performance using confidence intervals and simple statistics [26].

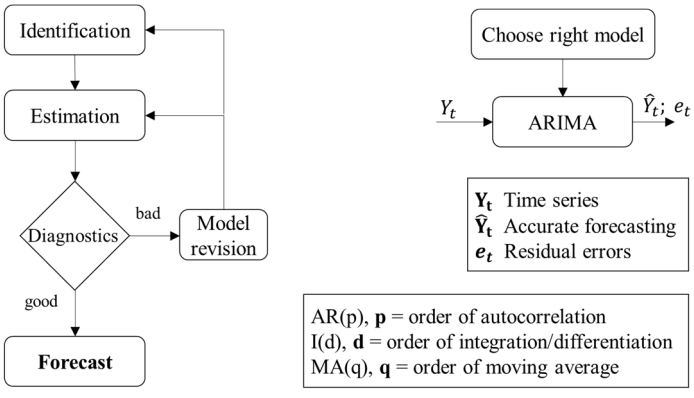

Figure 1. (**a**) ARIMA algorithm schematic; (**b**) ARIMA model assumptions where $Y_t = f(Y_{t-k}, e_{t-k}) + e_t$ and $k > 0$.

For a non-parametric technique such as ANN, the model is trained through non-linear algorithms. These self-adapting models do not require a priori assumptions of the series due to their flexibility in building model topologies and ability to easily identify and predict behavior patterns in the series [16,25]. In this case, the training is conducted point-to-point; if the amount of atypical data is minimal, then the correct data fix the error generated by the atypical data, thereby converging to the exact model [27].

The Elman Neural Network (ENN) is a subclass of RNN. The ENN algorithm (Figure 2) starts with an input layer followed by a hidden layer and a context layer (delay layer) with the same number of neurons. The feedback gives temporality to the network, providing the system with short-term memory. This memory process occurs through delay units that are fed by the neurons of the hidden layer. The weights of the connections between the hidden layer and delay units are fixed and equal to 1, allowing one to obtain a copy of the output values of the hidden layer from the previous step [28].

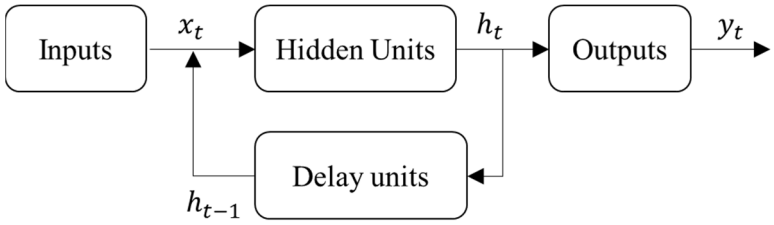

Figure 2. Elman network algorithms.

In Figure 2, the input layer variables are represented by x_t, and the hidden layer variables are represented by y_t. The hidden layer vector is represented by the expression $h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h)$, and the output weight matrix is represented by $y_t = \sigma_y(W_y h_t + b_y)$, where $W = \left[W_x\ W_y\right]^T$ and U_h are weight matrices, and $b = \left[b_h\ b_y\right]^T$ is the bias. Here, σ_h and σ_y are the activation functions for the hidden and output layers, respectively [28].

Similar to ENN, the LSTM network can remember a relevant sequence of data and preserve it for several time instances. In this way, an LSTM network can achieve short-term memory similar to that of basic recurrent networks, as well as long-term memory. As shown in Figure 3a, each block of the LSTM network can contain several cells in a similar manner to an Elman network, only replacing the neurons and hidden units with a memory block (LSTM cell; Figure 3b).

In Figure 3, the input and output are represented by x_t and y_t, respectively, while the vector h_t represents short-term memory, and c_t represents long-term memory. For time series predictions of x_t, the LSTM system updates the memory cell c_t and outputs a hidden state h_t for each step t. Equation (1) represents the mechanism of LSTM [14,29]:

$$\begin{aligned} i_t &= \sigma\left(W_{xi}^T \cdot x_t + W_{hi}^T \cdot h_{t-1} + b_i\right) \\ f_t &= \sigma\left(W_{xf}^T \cdot x_t + W_{hf}^T \cdot h_{t-1} + b_f\right) \\ o_t &= \sigma\left(W_{xo}^T \cdot x_t + W_{ho}^T \cdot h_{t-1} + b_o\right) \\ g_t &= tanh\left(W_{xg}^T \cdot x_t + W_{hg}^T \cdot h_{t-1} + b_g\right) \\ c_t &= f_t \otimes c_{t-1} + i_t \otimes g_t \\ y_t &= h_t = o_t \otimes tanh(c_t) \end{aligned} \tag{1}$$

The forget gate (f_t), input gate (i_t), and output gate (o_t) are fed by the input x_t and a previous short-term state, h_{t-1}, that includes gate g_t, where σ stands for the standard logistic sigmoid function $\sigma(x) = 1/(1 + e^{-x})$, and $tanh$ is denoted by $tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$. The weight matrices W_{xi}, W_{xf}, W_{xo}, W_{xg} and W_{hi}, W_{hf}, W_{ho}, W_{hg} are connected to the input vector x_t and short-term h_{t-1}; b_i, b_f, b_o, b_g are the bias terms for each of the four layers, where b_f is initialized in 1 s to avoid forgetting everything at the beginning of the training [14,29].

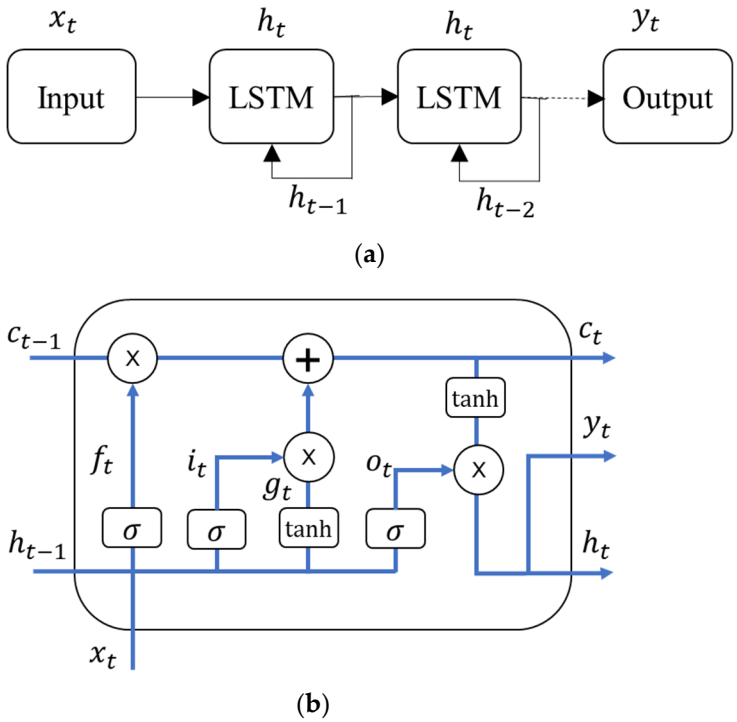

Figure 3. (**a**) Algorithm for several LSTM network layers; (**b**) LSTM cell [29,30].

2.2. Data and Sampling

ARIMA, Elman, and LSTM were used to forecast the time series to analyze the accuracy of the model. For this proposal, the time series represents the EUR/USD exchange rate's daily value. The data were obtained from the records on Investing.com from 2 January 1998, to 31 December 2019, with a total of 5737 observations. Each observation represents the daily price of the EUR/USD exchange rate from Monday to Friday. To apply the prediction techniques, the time series were divided into training and validation sets. For training, the dataset (*train*) used the expression $train = n - valid$, where n represents the total observations, and *valid* stands for the validation dataset, which includes windows with different forecasting horizons of 5, 11, 22, 35, 44, and 55 observations.

The window that provided the best forecasting in the short term was selected to evaluate the performance of the three forecasting methods. Theil's U coefficient and the Diebold–Mariano test were used to evaluate the forecasting method's accuracy.

Finally, we obtained the mean absolute percentage error (MAPE) of the selected prediction window to identify the observations where the method provided the greatest accuracy.

2.3. Application of Models

Based on the time series behavior graphically presented in Figure 4, the series does not follow a specific pattern in time, and the peaks do not oscillate around the average. Instead, the peaks are far from the average. Moreover, the series shows seasonality, indicating a random time series.

In the period analysis, the time series achieved a minimum price of USD 0.8273 and a maximum price of USD 1.5988. The average daily price was USD 1.1992. Forty percent of the time series were below the average market rates, and 60% were equal to or exceeded the average.

2.3.1. Application of ARIMA

The time series showed random behavior, as indicated in Figure 4. Therefore, a series analysis was carried out to identify the ARIMA model that best fits the data according to the ARIMA algorithm (Figure 1).

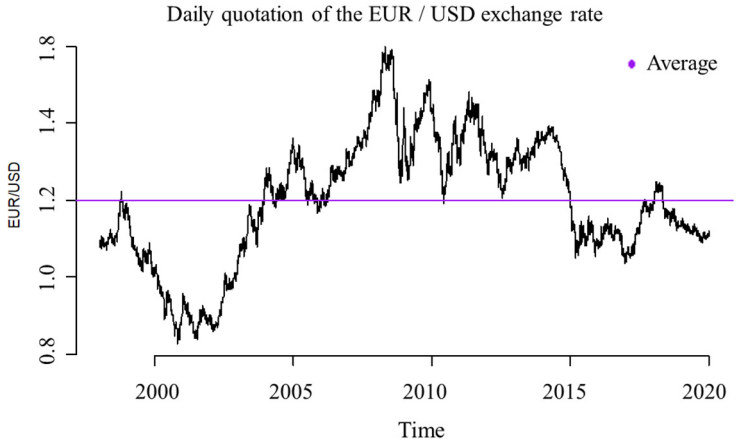

Figure 4. Daily prices of the EUR/USD exchange rate.

First, we verify the stationarity of the series using a Dickey–Fuller test at a significance level of $\alpha = 0.05$. We used the R software (v. 3.5.3) to codify ARIMA.

R output: Augmented Dickey–Fuller Test
data: train *Dickey–Fuller = −1.5839, Lag order = 0, p-value = 0.7546* *alternative hypothesis: stationary*

The p-value = 0.7546 > $\alpha = 0.05$; therefore, the time series is not stationary. To make the time series stationary, we performed first-order differentiation followed by another stationarity test.

R output: Augmented Dickey–Fuller Test
data: train_diff *Dickey–Fuller = −76.845, Lag order = 0, p-value = 0.01* *alternative hypothesis: stationary*

The p-value = 0.1 < $\alpha = 0.05$; therefore, the series is stationary. The differentiated series and the ACF were then analyzed. The former presented white-noise behavior, and the ACF corroborated the independence of the data. Based on this analysis, the best model to fit random walk was the ARIMA model (0, 1, 0), which corresponds to the following expression:

$$(1 - B)Y_t = \varepsilon_t \quad (2)$$

where B is the delay operator, $BY_t = Y_{t-1}$, and $\varepsilon_t \sim RB(0, \sigma^2)$. By testing the assumptions, we can verify the stationarity, independence, normality, and homoscedasticity.

R output: ARIMA testing of assumptions
Box–Ljung test (Independence) *data: modelo$residuals* *X-squared = 1.5593, df = 1, p-value = 0.2118* Lilliefors (Kolmogorov–Smirnov) (Normality test) *data: modelo$residuals* *D = 0.046857, p-value < 2.2 × 10^−16* Goldfeld–Quandt test (Homoscedasticity) *data: modelo$residuals ~ 1* *GQ = 1.037, df1 = 2857, df2 = 2856, p-value = 0.1656*

For stationarity, the p-value = 0.1 < $\alpha = 0.05$, so the residuals are stationary. For independence, the p-value = 0.2118 > $\alpha = 0.05$, so the residuals are independent. For normality, the p-value = 2.2^{-16} < $\alpha = 0.5$, so the residuals do not follow a normal distribution. For homoscedasticity, the p-value = 0.1656 > $\alpha = 0.05$, so the residuals present homogeneous variance. Thus, the normality test was not fulfilled. Consequently, the ARIMA model will not yield reliable results a priori; however, we applied this model to the data for comparison purposes.

2.3.2. Application of Recurrent Neural Networks

A. Data preprocessing

The data scale influences the processing of deep neural networks, mainly when using the sigmoidal or hyperbolic tangent activation functions. An alternative standardization was used to scale the data to give an absolute minimum and maximum value for each variable with intervals [−1, 1] and [0, 1]. To scale the random variable y_i, we used Equation (3):

$$z_i = \frac{y_i - \min(Y)}{\max(Y) - \min(Y)} \quad (3)$$

where min (Y) and max (Y) are the minimum and maximum values of the vector Y, and z_i is scaled between 0 and 1.

With the scaled data, we proceeded to build the RNN using Elman cells and LSTM. Each network parameter was then adjusted (activation functions, loss functions to minimize, number of layers, number of neurons in each layer, etc.) until better results were obtained.

The dropout technique was applied to the neural networks to avoid obtaining a smaller network due to overtraining [31], a process where randomly selected sets of neurons during the training phase are ignored with a probability of $1 - p$. If the parameter is closer to 0, fewer neurons are deactivated, and if it is closer to 1, more variables are deactivated. The optimal dropout value obtained for this study was 0.2. Poorer results were obtained when not using the dropout technique.

B. Elman Cell

This network consisted of 110 time steps (inputs) to forecast the window steps (outputs) and featured two layers with 3 and 2 neurons in each. We used the R software v. 3.5.3 to code the Elman cell.

R code: Elman cell

```
fit_elman ← elman(x = inputs, y = outputs,
size = c(3,2),
maxit = 7000,
learnFuncParams = c(0.2))
```

Here, maxit = 7000 (the number of iterations to train the model), and learnFuncParams = 0.2 (network training speed). Figure 5 shows the network error.

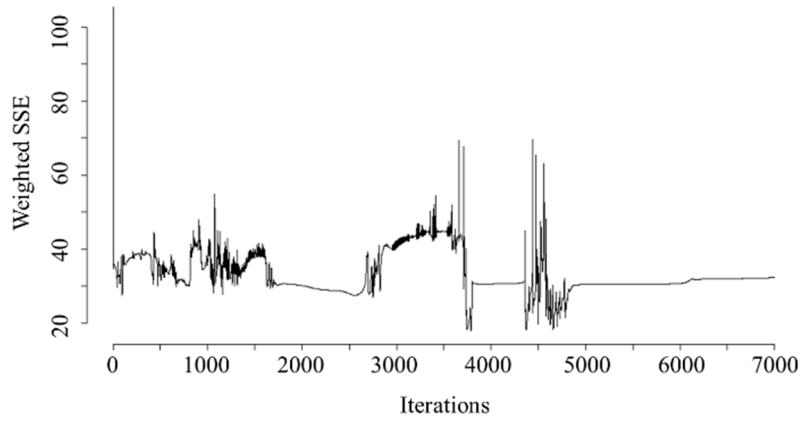

Figure 5. Elman network error with random behavior for the first 5000 iterations. The error stabilized after iteration 5000.

C. LSTM Cell

This network consisted of two hidden layers with 50 units in each, a *relu* activation function, and an output layer (Dense). In the first layer, the neurons have three-dimensional outputs (return_sequences = True), input variables (input_shape), and independent term bias (use_bias = True). We used the software Python v. 2.7.13 to code the LSTM cell.

Python code: LSTM
model.add(LSTM(50, *activation = "relu",* *return_sequences = True,* *input_shape = (n_steps_in, n_features),* *use_bias = True))model.add(LSTM(50, activation = "relu"))model.add(Dropout(0.2))* *model.add(Dense(n_steps_out))* *model.compile(optimizer = 'adam', loss = 'mae')*

In this case, the mean absolute error (MAE) was used as the loss function for the network, and (adam) was used as an optimizer. Figure 6 shows the behavior of the error as a function of the number of iterations. Here, the error is large for iterations one and two and stabilizes starting from iteration three.

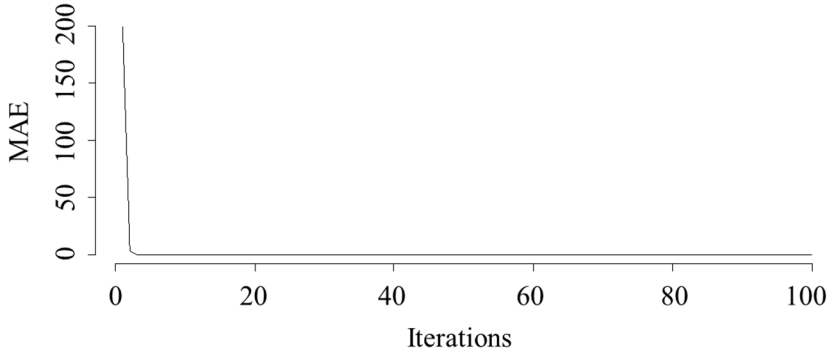

Figure 6. LSTM network error.

3. Results and Discussion

3.1. Selection the Forecasting Window

In this section, we compare the forecasting dataset calculated with the ARIMA (0, 1, 0) model, Elman, and LSTM using the validation dataset for windows of 5, 11, 22, 35, 44, and

55 days. The purpose of this comparison is to select the window that best approximates the validation dataset in the short term.

As shown in Figure 7, for the 5-day forecasting window, the Elman model provided predictions far from the validation dataset, while ARIMA and LSTM provided similar results to the validation dataset in observations 3 and 4. For the 11-day window, the forecasting dataset better approximated the validation dataset for the three models. LSTM matched in three observations, while ARIMA and ELMAN matched in only one observation. For the third prediction window of 22 days, the LSTM forecasting dataset closely approximated the validation dataset in ten observations, while Elman matched the validation dataset in just one observation. ARIMA remained constant, with predictions far from the validation dataset. The Elman network presented an upward trend in its forecasts, while for the LSTM network, the forecasting data showed random behavior.

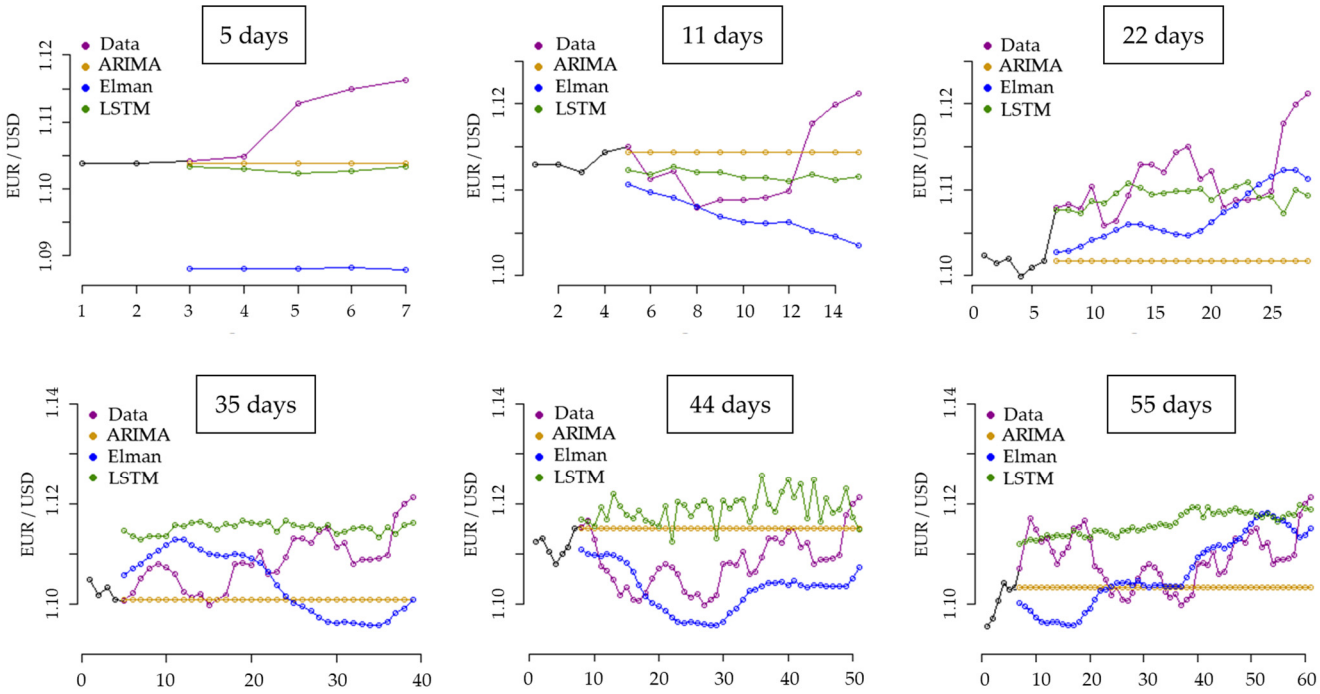

Figure 7. Forecasting method comparison using six different prediction horizons.

For the prediction windows of 35, 44, and 55 days, the forecasting dataset obtained with the Elman network best matched the validation dataset and improved further when considering long-term predictions. ARIMA and LSTM most poorly approximated the validation dataset for long-term forecasts with these types of time series behavior and conditions.

In this analysis, the 22-day forecasting window best approximated the validation dataset in the short term. According to these results, we selected this window to evaluate and compare the prediction methods in detail.

3.2. Prediction Evaluation Measures

Performance metrics were used for the evaluation measures (scale-dependent errors and percentage errors; Table 1) [32]. These metrics included the Mean Square Error (MSE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Median Absolute Error (MdAE), Mean Absolute Percentage Error (MAPE), Symmetric Mean Absolute Percentage Error (sMAPE), Root Mean Square Percentage Error (RMSPE), Root Median Square Percentage Error (RMdSPE), Median Absolute Percentage Error (MdAPE), and Symmetric Median Absolute Percentage Error (SMdAPE). As shown in Table 1, the errors obtained through the LSTM network were less significant than the errors obtained with the Elman

network and the ARIMA model (0, 1, 0). These results corroborate the forecast accuracy obtained with these three methods (Figure 7; 22-day window).

Table 1. Performance metrics for the three prediction models.

	Scale-Dependent Errors				Percentage Errors					
Model	**MSE**	**RMSE**	**MAE**	**MdAE**	**MAPE**	**sMAPE**	**RMSPE**	**RMdSPE**	**MdAPE**	**SMdAPE**
ARIMA	0.0001102	0.010496	0.009673	0.0084	0.869%	0.2183%	0.9414%	0.7572%	0.7567%	0.1898%
ELMAN	3.391×10^{-5}	0.00582	0.00494	0.0053	0.444%	0.111%	0.5226%	0.483%	0.483%	0.121%
LSTM	1.615×10^{-5}	0.004	0.0031	0.0020	0.282%	0.0706%	0.3607%	0.181%	0.181%	0.0453%

3.3. Accuracy Analysis

By using the first expression for U1 given in [33], we obtained Theil's U coefficients from the forecasting set to evaluate the accuracy of the models. Theil's coefficients are bounded by 0 and 1, where the lower boundary indicates a perfect forecast, and the upper boundary indicates unreliable forecasting. Coefficients close to 1 represent fully impractical situations for exchange forecasting. The use of these values would involve repeatedly performing the forecasting process to find negative forecasts or negative exchanges. Theil's U coefficients for the three forecasting methods are listed in Table 2.

Table 2. Theil's U coefficient.

ARIMA	**ELMAN**	**LSTM**
0.004743	0.002625	0.001808

The three models give values close to 0. Therefore, the three techniques provide reliable predictions where the coefficient for the LSTM network is lower than that of the ELMAN and ARIMA models.

A Diebold–Mariano test was then performed to compare the prediction accuracy of the three techniques. Table 3 presents the different p values for the two–sided and *greater* hypotheses.

two–sided

H_0: The two techniques offer the same prediction precision.

H_1 : The two techniques do not offer the same prediction precision.

greater

H_0: The two techniques offer the same prediction precision.

H_1 : Technique 2 is more accurate than technique 1.

As shown in Table 3, by comparing the p-values obtained with a significance level of $\alpha = 0.05$, we can deduce the following results:

ARIMA vs. Elman: For two–sided, the value of $p = 1.694 \times 10^{-4} < \alpha$; therefore, H_0 is rejected, and it can be concluded that the two techniques did not have the same accuracy. For *greater*, $p = 8.468 \times 10^{-5} < \alpha$, and H_0 is rejected; thus, it can be concluded that the predictions performed with Elman were more precise than those made with ARIMA.

ARIMA vs. LSTM: For two–sided, $p = 4.287 \times 10^{-5} < \alpha$, and H_0 is rejected; thus, it can be concluded that the two techniques did not have the same accuracy. For *greater*, $p = 2.144 \times 10^{-5} < \alpha$; thus, H_0 is rejected, and it can be concluded that the predictions performed with LSTM were more accurate than those made with ARIMA.

Table 3. Diebold–Mariano test.

	ARIMA vs. Elman	**ARIMA vs. LSTM**	**Elman vs. LSTM**
two–sided	1.694×10^{-4}	4.287×10^{-5}	1.54×10^{-2}
greater	8.468×10^{-5}	2.144×10^{-5}	7.7×10^{-3}

Elman vs. LSTM: For two–sided, the value $p = 0.0154 < \alpha$, and H_0 is rejected; thus, the two techniques did not have the same accuracy. For *greater*, $p = 0.0077 < \alpha$, and H_0 is rejected; thus, it can be concluded that the forecasting performed with LSTM was more accurate than that performed with Elman.

Based on these results, the predictions made with the LSTM network were more accurate than the forecasts performed with the ARIMA (0, 1, 0) model and the Elman network for short-term forecasting.

3.4. Accuracy Based on Observations

To identify the observations where the method provided the greatest accuracy, we calculated the MAPE of the forecasting data obtained with LSTM for the 22-day window. The prediction data corresponded to observations from 2 December to 31 December 2019. As shown in Figure 8, the MAPE results presented randomness, with the last day of predictions showing a significantly higher error percentage than the other days. The error rates were high on 2, 3, 4, 6, 9, 16, 27, and 30 December and low on 5, 10, 12, 17, 20, 25, and 26 December. The lowest error rates were observed on 11, 13, 18, and 24 December. These results may be due to the volatility of this type of time series.

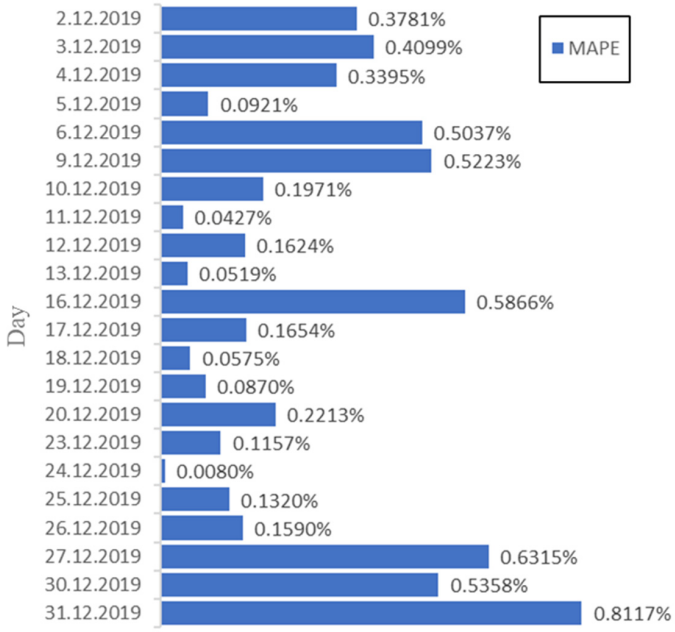

Figure 8. MAPE of the 22 predictions.

LSTM network performance in out-of-sample forecasting using the EUR/USD exchange rate dataset.

4. Conclusions

In this paper, we compared three methods—ARIMA models, Elman, and LSTM networks—to perform out-of-sample forecasting of the EUR/USD exchange rate dataset. The time series did not present a trend, seasonality, or stationarity; therefore, the time series was determined to be a random walk. An ARIMA (0, 1, 0) model was selected to analyze a first-order differentiated series and its ACF. Elman and LSTM networks were modeled using systematic simulations by adjusting parameters until we obtained the best results. These three models were then used to forecast and compare six windows for the validation datasets. Through this comparison, we determined that ARIMA generated constant numbers, while LSTM provided the best forecasts up to a 22-day window in the short term. With Elman, we obtained better results in the long term. The selected window was then evaluated in detail to identify the observations with the lowest errors,

finding only four observations among the 22 windows where LSTM best approximated the validation dataset.

The average accuracy for the 22-day window was 71.76%. By comparing our results with the results of previous studies, we can conclude the following. First, it is difficult to make a direct comparison since there is no standardized method for selecting training and validation datasets. Second, previous studies reported an average accuracy lower than 70% for different approaches [34–36] and achieved greater than 70% average accuracy when using NN, combining models, or introducing a hybrid model [37–40]. Based on the comparison of the three forecasting models in this work, LSTM fit better in the short term, although the results were not entirely desirable, as LSTM only coincided in four observations with 95.99% average accuracy.

The advantage of using an RNN of the Elman or LSTM type is its efficiency when working with a time series. Both techniques have similar characteristics in their networks; the only difference lies in their memory capacity. The limitation of ARIMA is that it represents a general univariate model in which the assumptions must be fulfilled to succeed. A combination of these methods as a type of hybrid model could be an aim of future studies, similar to the model reported in [41,42].

Author Contributions: Programming, figures formatting, and draft the process, W.A.; first draft preparation; P.E.; supervision, conceptualization, P.E., J.P.; writing—review and editing, W.A., P.E., J.P.; funding acquisition, P.E. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Indoamerican Technological University and Chimborazo's Superior Polytechnic School.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goodman, S.H. Foreign Exchange Rate Forecasting Techniques: Implications for Business and Policy. *J. Financ.* **1979**, *34*, 415–427. [CrossRef]
2. Pacelli, V. Forecasting Exchange Rates: A Comparative Analysis. *Int. J. Bus. Soc. Sci.* **2012**, *3*, 12.
3. Canova, F. Modelling and Forecasting Exchange Rates with a Bayesian Time-Varying Coefficient Model. *J. Econ. Dyn. Control* **1993**, *17*, 233–261. [CrossRef]
4. Jung, G.; Choi, S.-Y. Forecasting Foreign Exchange Volatility Using Deep Learning Autoencoder-LSTM Techniques. *Complexity* **2021**, *2021*, e6647534. [CrossRef]
5. Tripathi, M.; Kumar, S.; Inani, S.K. Exchange Rate Forecasting Using Ensemble Modeling for Better Policy Implications. *J. Time Ser. Econom.* **2021**, *13*, 43–71. [CrossRef]
6. Nyoni, T. Modeling and Forecasting Naira / USD Exchange Rate In Nigeria: A Box—Jenkins ARIMA Approach. *MPRRA* **2018**, *88622*, 1–36.
7. Arango, F.O.; Cabrera Llanos, A.I.; Herrera, F.L. Pronóstico de los índices accionarios DAX y S&P 500 con redes neuronales diferenciales. *Contaduría Y Adm.* **2013**, *58*, 203–225. [CrossRef]
8. Poznyak, A.S.; Sanchez, E.N.; Yu, W. *Differential Neural Networks for Robust Nonlinear Control*; World Scientific: Singapore, 2001; pp. 12–50. [CrossRef]
9. Arango, F.O.; Aranda, F.C. Redes Neuronales Diferenciales: Una Alternativa Confiable Para Analizar Series Financieras. In *Proceedings of the XVI Congreso Internacional de Contaduría Administración e Informática*; Mexico, D.F., Ed.; Medellín, U. D.: Ciudad de México, Mexico, 2011; pp. 49–64.
10. Zapata, L.A.; Hugo, D. Predicción Del Tipo de Cambio Peso-Dólar Utilizando Redes Neuronales Artificiales (Rna). *Pensam. Gestión* **2008**, *24*, 29–42.
11. Van Houdt, G.; Mosquera, C.; Nápoles, G. A Review on the Long Short-Term Memory Model. *Artif. Intell. Rev.* **2020**, *53*, 5929–5955. [CrossRef]
12. Yıldırım, D.C.; Toroslu, I.H.; Fiore, U. Forecasting Directional Movement of Forex Data Using LSTM with Technical and Macroeconomic Indicators. *Financ. Innov.* **2021**, *7*, 1. [CrossRef]
13. Zhang, C.; Fang, J. Application Research of Several LSTM Variants in Power Quality Time Series Data Prediction. In *Proceedings of the 2nd International Conference on Artificial Intelligence and Pattern Recognition*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 171–175.

14. Choi, J.Y.; Lee, B. Combining LSTM Network Ensemble via Adaptive Weighting for Improved Time Series Forecasting. *Math. Probl. Eng.* **2018**, *2018*, 1–8. [CrossRef]
15. Babu, A.S.; Reddy, S.K. Exchange Rate Forecasting Using ARIMA, Neural Network and Fuzzy Neuron. *J. Stock Trad.* **2015**, *4*. [CrossRef]
16. Adebiyi, A.A.; Adewumi, A.O.; Ayo, C.K. Comparison of ARIMA and Artificial Neural Networks Models for Stock Price Prediction. *J. Appl. Math.* **2014**, *2014*, 1–7. [CrossRef]
17. Li, M.; Ji, S.; Liu, G. Forecasting of Chinese E-Commerce Sales: An Empirical Comparison of ARIMA, Nonlinear Autoregressive Neural Network, and a Combined ARIMA-NARNN Model. *Math. Probl. Eng.* **2018**, *2018*, 1–12. [CrossRef]
18. Son, H.; Kim, C. A Deep Learning Approach to Forecasting Monthly Demand for Residential–Sector Electricity. *Sustainability* **2020**, *12*, 3103. [CrossRef]
19. Wang, J.-J.; Wang, J.-Z.; Zhang, Z.-G.; Guo, S.-P. Stock Index Forecasting Based on a Hybrid Model. *Omega* **2012**, *40*, 758–766. [CrossRef]
20. Islam, M.S.; Hossain, E. Foreign Exchange Currency Rate Prediction Using a GRU-LSTM Hybrid Network. *Soft Comput. Lett.* **2020**, 100009. [CrossRef]
21. Musa, Y.; Joshua, S. Analysis of ARIMA-Artificial Neural Network Hybrid Model in Forecasting of Stock Market Returns. *Asian J. Probab. Stat.* **2020**, 42–53. [CrossRef]
22. Wang, J.-N.; Du, J.; Jiang, C.; Lai, K.-K. Chinese Currency Exchange Rates Forecasting with EMD-Based Neural Network. *Complexity* **2019**, *2019*, e7458961. [CrossRef]
23. Qiu, Y.; Yang, H.-Y.; Lu, S.; Chen, W. A Novel Hybrid Model Based on Recurrent Neural Networks for Stock Market Timing. *Soft Comput.* **2020**, *24*, 15273–15290. [CrossRef]
24. Hu, Z.; Zhao, Y.; Khushi, M. A Survey of Forex and Stock Price Prediction Using Deep Learning. *Appl. Syst. Innov.* **2021**, *4*, 9. [CrossRef]
25. Menacho, C. Comparación de Los Métodos de Series de Tiempo y Redes Neuronales. *An. Científicos* **2014**, *75*, 245–252. [CrossRef]
26. Box, G.E.P.; Jenkins, G.M.; Reinsel, G.C.; Ljung, G.M. *Time Series Analysis: Forecasting and Control*, 5th ed.; Wiley: Hoboken, NJ, USA, 2015; ISBN 978-1-118-67502-1.
27. Deb, C.; Zhang, F.; Yang, J.; Lee, S.E.; Shah, K.W. A Review on Time Series Forecasting Techniques for Building Energy Consumption. *Renew. Sustain. Energy Rev.* **2017**, *74*, 902–924. [CrossRef]
28. Lewis, N.D.C. *Deep Time Series Forecasting with Python: An Intuitive Introduction to Deep Learning for Applied Time Series Modeling*; CreateSpace Independent Publishing Platform: Scotts Valley, CA, USA, 2016; ISBN 978-1-5408-0908-7.
29. Géron, A. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2017; ISBN 978-1-4919-6229-9.
30. Dautel, A.J.; Härdle, W.K.; Lessmann, S.; Seow, H.-V. Forex Exchange Rate Forecasting Using Deep Recurrent Neural Networks. *Digit Financ.* **2020**, *2*, 69–96. [CrossRef]
31. Gal, Y.; Ghahramani, Z. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. *arXiv* **2016**, arXiv:1512.05287.
32. Botchkarev, A. Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology. *arXiv* **2018**, arXiv:10.28945/4184*14*, 45–79.
33. Bliemel, F. Theil's Forecast Accuracy Coefficient: A Clarification. *J. Mark. Res.* **1973**, *10*, 444–446. [CrossRef]
34. Yao, J.; Tan, C.L. A Case Study on Using Neural Networks to Perform Technical Forecasting of Forex. *Neurocomputing* **2000**, *34*, 79–98. [CrossRef]
35. Zhang, L.; Aggarwal, C.; Qi, G.-J. Stock Price Prediction via Discovering Multi-Frequency Trading Patterns. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; Association for Computing Machinery: New York, NY, USA, 2017; pp. 2141–2149.
36. Shen, F.; Chao, J.; Zhao, J. Forecasting Exchange Rate Using Deep Belief Networks and Conjugate Gradient Method. *Neurocomputing* **2015**, *167*, 243–253. [CrossRef]
37. Huang, W.; Nakamori, Y.; Wang, S.-Y. Forecasting Stock Market Movement Direction with Support Vector Machine. *Comput. Oper. Res.* **2005**, *32*, 2513–2522. [CrossRef]
38. Galeshchuk, S.; Mukherjee, S. Deep Networks for Predicting Direction of Change in Foreign Exchange Rates. *Intell. Syst. Account. Financ. Manag.* **2017**, *24*, 100–110. [CrossRef]
39. Özorhan, M.O.; Toroslu, İ.H.; Şehitoğlu, O.T. A Strength-Biased Prediction Model for Forecasting Exchange Rates Using Support Vector Machines and Genetic Algorithms. *Soft Comput.* **2017**, *21*, 6653–6671. [CrossRef]
40. Bao, W.; Yue, J.; Rao, Y. A Deep Learning Framework for Financial Time Series Using Stacked Autoencoders and Long-Short Term Memory. *PLoS ONE* **2017**, *12*, e0180944. [CrossRef] [PubMed]
41. Lin, H.; Sun, Q.; Chen, S.-Q. Reducing Exchange Rate Risks in International Trade: A Hybrid Forecasting Approach of CEEMDAN and Multilayer LSTM. *Sustainability* **2020**, *12*, 2451. [CrossRef]
42. Sreeram, L.; Sayed, S.A. Short-Term Forecasting Ability of Hybrid Models for BRIC Currencies. *Glob. Bus. Rev.* **2020**, 0972150920954615. [CrossRef]