Muhammed Raşid BAKIR, PhD Candidate (corresponding author)
E-mail: mrasidbakir@aksaray.edu.tr
Aksaray University, Turkey

Professor İbrahim BAKIRTAŞ, PhD
E-mail: ibakirtas@aksaray.edu.tr
Aksaray University, Turkey

Associate Professor Emre ÖLMEZ, PhD
E-mail: emre.olmez@bakircay.edu.tr
İzmir Bakırçay University, Turkey

STATISTICAL TECHNIQUES VS. MACHINE LEARNING MODELS: A COMPARATIVE ANALYSIS FOR EXCHANGE RATE FORECASTING IN FRAGILE FIVE COUNTRIES

Abstract. *In 2013, the Federal Reserve (Fed) announced the end of its expansionary monetary policy, which had a significant impact on certain countries. These countries, colloquially referred to as the "fragile five", were heavily dependent on financial capital flows, which led to deviations from inflation targets due to the exchange rate pass-through effect. Consequently, monetary authorities and other financial actors need accurate exchange rate forecasts to mitigate these deviations and improve the effectiveness of monetary policy. This study aims to forecast the exchange rates of the fragile five countries using both traditional statistical methods and machine learning techniques. The traditional statistical methods used in this study include Naïve Drift, Theta, Holt's Exponential Smoothing and ARIMA models, while the machine learning methods include RNN, LSTM, GRU and CNN architectures. The results show that machine learning methods outperform traditional statistical methods in terms of prediction accuracy for all countries. While statistical methods show a directional accuracy rate between 47% and 60%, RNN, one of the machine learning models, shows an accuracy rate between 80% and 90%. Overall, these results suggest that machine learning methods can provide more accurate exchange rate forecasts for the fragile five countries than traditional statistical methods. These findings may be valuable for monetary authorities and financial actors seeking to improve the effectiveness of monetary policy in these countries.*

Keywords: *machine learning, forecasting methods, emerging market, monetary policies, exchange rate*

JEL Classification: E47, G17, F31, F37, E52

DOI: 10.24818/18423264/57.3.23.18

1. Introduction

The year 2013 marked a significant period in global financial markets, particularly for developing countries. The Federal Reserve's decision to end its expansionary monetary policy led to significant capital outflows from developing countries, which increased exchange rate volatility (Nechio, 2014). A report by Morgan Stanley further indicated that Brazil, India, Indonesia, South Africa, and Turkey were the most vulnerable to the Fed's policy adjustments due to their high inflation rates and current account deficits. These countries were therefore referred to as the 'fragile five' and financial investors were advised to trade the US dollar position against their currencies. The Fed's policy shift and the Morgan Stanley report underscored the importance of reliable exchange rate forecasts for monetary authorities and financial investors.

The fragile five countries rely heavily on financial capital flows, which make their currencies volatile. This volatility creates many trading opportunities for financial investors. However, it also poses a challenge for monetary authorities, who have to take into account foreign exchange liabilities, forward options, and swap transactions that can affect macroeconomic variables. In addition, the pass-through effects of the exchange rate on imported intermediate and final goods may deviate from the target inflation rate and jeopardise the legitimacy of the monetary authority (Fendoğlu, 2020; Takhtamanova, 2010). As a result, monetary authorities and other financial market players are forced to develop predictive models to reduce the exchange rate risk.

As the use of machine learning (ML) methods increases in many scientific fields and high-frequency data becomes more accessible, the predictive performance of traditional statistical methods has been questioned compared to ML approaches. The open algorithmic design of ML methods and their ability to handle data without restrictions make them suitable for addressing nonlinear problems (Zhang, Patuwo, & Michael, 2001; Zhang G. P., 2003; Athey, 2018). This study aims to compare the success rates of exchange rate forecasting using traditional statistical methods and ML approaches for the fragile five countries. For ML modelling, unprocessed standard hyperparameters were chosen to question the success of traditional statistical methods, which are based on several procedures, in comparison to ML methods. The following sections provide a brief review of the literature and a detailed explanation of the theoretical underpinnings of both traditional statistical forecasting methods and machine learning models. The next section describes in detail the prediction results of the constructed models and provides comparisons. The conclusion section emphasises the messages that the results convey to economic stakeholders and provides guidance on which variables should be added to the models in future studies in this area.

2. Literature Review

The literature on the fragile five countries focuses on whether exchange rates are linked across countries, the interactions of exchange rates with other macroeconomic variables at the local level, and the consequences of expansionary policies of the central bank (Bhattarai, 2021; Hersi & Koy, 2020; Akel, Kandir, & Yavuz, 2015). No study has been found that compares statistical and machine learning approaches to forecasting exchange rates for the fragile five countries. In this context, the present study is expected to contribute to the new economic literature in terms of sample and procedure.

The study by Galeshchuk & Mukherjee (2017) used autoregressive moving average (ARIMA), exponential smoothing, artificial neural network (ANN), support vector machine (SVM), and convolutional neural network (CNN) models to forecast the EUR/USD, GBP/USD and JPY/USD pairs, which are the major currency pairs in the FX market. In the study comparing classification success rates, it was found that the success rate of the models created using deep learning algorithms was very high compared to statistical models.

Nagpure (2019) studied different artificial intelligence approaches to predict high-volume currencies of 11 countries using daily data from 30-39 years. Support Vector Regression (SVR), ANN with two hidden layers, and Long Short-Term Memory (LSTM) were used as artificial intelligence models. The study, which used Mean Squared Error (MSE) and criteria to determine prediction accuracy rates, found that the two-hidden-layer artificial neural network had a success rate of over 99%.

Machine learning and traditional time series forecasting approaches were used by Liao (2017) to forecast macroeconomic series. In the study, which uses quarterly data from 1994 to 2015, the input variables are the S&P 500 yield, the 30- and 20-year Treasury bill rates, the average bond rate of firms with credit ratings between BAA and AAA, the unemployment rate, and the M2 money supply growth rate, and the fixed input variables of the private sector. The investment variables are the volatility index (VIX), actual personal consumption expenditure, the federal funds rate, and the cyclically adjusted dividend rate. The output variable is GDP, which is used as a proxy for the growth rate. As a result of the analysis, it was found that increasing the number of clusters increases the forecasting success in modelling with nonlinear artificial neural networks based on the K-mean Markov switching model.

In their study, Kadilar, Şimşek & Aladağ (2009) modelled in-sample and out-sample volatility for the USDTRY parity using Artificial Neural Network (ANN) and Autoregressive Conditional Heteroskedasticity (ARCH) methods. In the ANN models with logistic and linear activation functions, it was concluded that the logistic activation function increased the success rate and the predictive power of the ANN algorithm was more successful than the ARCH method.

3. Methodology and Data

The data for this study were obtained from the Yahoo-Finance database and consisted of exchange rates for the fragile five countries. The data frequency was business day, with observations spanning from 3 June 2013 to 29 March 2022. A total of 2288 observations were analysed for each country. Prior to the machine learning analysis, the data were normalised to fall within the range of 0-1.

For all models, 70% of the data was used for training and the remaining 30% for testing. Model parameters were derived from the information collected during the training process, and these parameters were used to generate predictions for the test set. To evaluate the accuracy of the predicted values, we used MAPE (Mean Absolute Percentage Error), RMSE (Root Mean Square Error), R^2 and MDA (Mean Directional Accuracy) by comparing the predicted values with the actual data.

3.1. Time Series Concept

A time series refers to a sequence of data that changes over time, denoted by $Y_t = f(t)$. If $f(t)$ is a known function, Y_t is deterministic. In such cases, we can express the equation as $Y_t = X(t)$, where $X(t)$ represents a random variable and $\{Y_t\}$ refers to a stochastic process.

3.1.1. The Concept of Stationarity in Time Series and Unit Root Test

Before carrying out a time series analysis, it is crucial to examine the trend of the process that generates the series over time. In order to obtain econometrically significant results, it is necessary for the series to be stationary. Nonstationary time series models can lead to erroneous conclusions that do not reflect the reality.

A time series Y_t is said to be stationary if the probability distribution does not change over time. In other words, if the joint distribution of ($Y_{s+1}, Y_{s+2}, \dots, Y_{s+t}$) does not depend on s, the series is stationary; otherwise, it is nonstationary. If two time series, such asX_t and Y_t $(X_{s+1}, Y_{s+1}, X_{s+2}, Y_{s+2}, \dots, X_{s+t}, Y_{s+t})$ have a joint distribution that does not depend on s, they are said to be jointly stationary. Stationarity implies that the future will be at least similar to the past, according to probabilistic inference. For a time series Y_t to be considered stationary, it must satisfy equations (1), (2), and (3):

$$E(Y_t) = \mu \tag{1}$$

$$Var(Y_t) = E(Y_t - \mu)^2 = \sigma^2 \tag{2}$$

$$\gamma_k = E[(Y_t - \mu)(Y_{t+k} - \mu) \tag{3}$$

Here γ_k is the k-lagged (or sequential) common variance between Y_t and Y_{t+k}, where k denotes the period difference between the two Y's. When $k = 0$, γ_0 is found, which is simply the variance of Y. In other words, if a time series is stationary, it has the same mean, variance, and common variance (at different lags) regardless of when it is measured. The Augmented Dickey-Fuller (ADF) unit root test is the most commonly used test for stationary. The ADF examines models with constant and trend, constant only, or neither constant nor trend in the equations in (4), (5), (6):

$$\Delta Y_t = \alpha_0 + \delta y_{t-1} + \alpha_1 t + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t \tag{4}$$

$$\Delta Y_t = \alpha_0 + \delta y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t \tag{5}$$

$$\Delta Y_t = \delta y_{t-1} + \delta_1 \Delta y_{t-1} + \cdots + \delta_{p-1} \Delta y_{t-p+1} + \varepsilon_t \tag{6}$$

The null hypotheses for the ADF test are $H_0: (\alpha_0,\ \delta,\ \alpha_1) = (0, 0, 0)$, $H_0: (\alpha_0,\ \delta) = (0, 0)$, $H_0: (\ \delta) = (0)$ (Dickey & Fuller, 1981). The calculated τ statistics ($\tau = \frac{\hat{\delta}}{S.E(\hat{\delta})}$) in equations (4), (5), and (6) are compared with the threshold values of τ, τ_μ and τ_τ, respectively. If the $|\tau|$ value exceeds the threshold value for the specified model, then the null hypothesis H_0 is rejected. The rejection of the null hypothesis implies that the Y is stationary.

It is necessary to take sufficient differences in nonstationary series to obtain stationarity. The number of lags p required for the error term to be white noise is sensitive to the ADF test ($\varepsilon_t \sim IID(0, \sigma_\varepsilon^2)$). Therefore, despite disagreements over the lag length p, it should be constrained to a degree of freedom concept that does not generate autocorrelation.

3.2 The Methods

3.2.1. Naïve Drift Method

The Naive method is a simple forecasting approach that assumes that all forecasts are equal to the last value in the observation set. However, the Naive Drift approach allows the estimates to vary over time, taking into account average deviations from historical data.

$$\hat{y}_{t+h|\mathrm{t}} = y_t + \frac{h}{t-1} \sum_{t=2}^{T} (y_t - y_{t-1}) = y_T + h\left(\frac{y_t - y_1}{t-1}\right) \tag{7}$$

Equation (7) represents the Naive Drift method and expresses a regression equation that includes the mean deviations between the first and last observations in the data set. Where y_T represents the observation value at time t, and h is the forecast horizon. By taking historical deviations into account, this method can provide more accurate forecasts than the Naive method (Hyndman, 2018).

3.2.2. Theta Method

The method proposed by Assimakopoulos (2000) has attracted the attention of forecasters due to its excellent performance in the M3 forecasting contest. The method combines simple exponential smoothing with a linear regression component and can be expressed algebraically as follows:
$y''_{t,\theta} = \theta x''_t$, where $\{x_1, x_2, x_3, \dots, x_n\}$ represents observations in a univariate time series. Here $x''_t = x_t - 2x_{t-1} + x_{t+2}$ and $y''_t = y_t - 2y_{t-1} + y_{t+2}$ denote the second order differences. Using the work of Box, Jenkins, and Reinsel (1994), we can rewrite the solution of the equation
$y''_{t,\theta} = \theta x''_t$ as $y_{t,\theta} = \alpha_\theta + b_\theta(t-1) + \theta x_t$ (8)
where α_θ and b_θ are constants. Thus $y_{t,\theta}$ is equivalent to a linear function of x_t, with a linear trend added.

To determine the model parameters, we calculate the Sum of Squared Error (SSE) for each parameter combination using the equation

$$\sum_{i=1}^{t} [x_t - y_{t,\theta}]^2 = \sum_{i=1}^{t} [(1-\theta)x_t - \alpha_\theta - b_\theta(t-1)]^2 \quad (9)$$

and choose the model parameters that give the smallest SSE value. It is worth noting that when $\theta = 0$, the model produces a linear regression line, and when $\theta \neq 1$, the model is a combination of simple exponential smoothing and linear regression (Spiliotis, 2019).

3.2.3. Holt-Winters Exponential Smoothing Method

Holt and Winter extended the two-parameter technique to a three-parameter model by adding the seasonal influence variable. In this model, high-frequency signals are filtered out of time series and smoothed. Similarly to the two-parameter model, the three-parameter model does not require stationarity. The model comprises four equations for level, trend, seasonality, and forecast, respectively. These equations are:

$$L_t = \alpha \frac{y_t}{s_{t-12}} + (1-\alpha)(L_{t-1} + b_{t-1}) \quad (10)$$

$$b_t = \beta(L_t - L_{t-1}) + (1-\beta)b_{t-1} \quad (11)$$

$$s_t = \gamma \frac{y_t}{L_t} + (1-\gamma)s_{t-1} \quad (12)$$

$$F_{t+m} = (L_t + b_t m)s_{t-s+m} \quad (13)$$

Where, L_t represents the level of the time series at time t, b_t represents the slope of the time series at time t, s_t expression represents the seasonal component at time t, s represents the seasonality period, and F_{t+m} represents the predicted value for period m after time t. The smoothing coefficients α, β, and γ have values in the range [0,1] and are chosen according to the lowest SSE value.

3.2.4. ARIMA Process

The ARIMA process combines the autoregressive (AR) process with the moving average (MA) process. The ARIMA process is formed by adding the difference equation that makes the variable of interest stationary. Equation (14) shows an ARIMA process with parameters (p, d, q).

$$\left(1-\varphi_1 B-\cdots-\varphi_p B^p\right)(1-B)^d Y_t=c+\left(1+\theta_1 B+\cdots+\theta_q B^q\right)\varepsilon_t \qquad (14)$$

Where ε_t represents the error term characteristic of white noise, B,

$AR(p)=(1-\varphi_1 B-\cdots-\varphi_p B^p)$ and $MA(q)=1+\theta_1 B+\cdots+\theta_q B^q$ (15)

represent the polynomial lag operator in the process, and the parameter (d) represents the difference operator in the $(1-B)$ lag processor. φ and θ are coefficients calculated based on the lag length in the $AR(p)$ and $MA(q)$ processes, respectively (Jirsik, Trčka, & Celeda, 2019).

3.2.5. Recurrent Neural Network (RNN)

A recurrent neural network is an artificial neural network that was introduced by Elman (1990) and is used to forecast time series. In an RNN with one or more hidden layers, each layer moves with the weight from the previous layer. The output information in RNN is derived from the previous computation, where the activation function is typically of the sigmoid type (Elman, 1990; Bianchi, Maiorino, Kampffmeyer, Rizzi, & Jenssen, 2017).

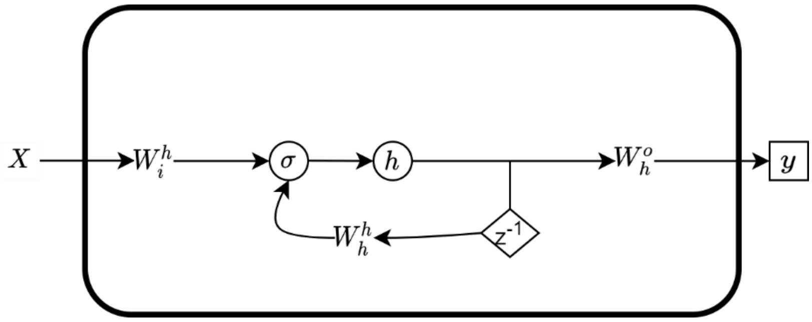

Figure 1. RNN architecture

Figure 1 illustrates the architecture of a recurrent neural network (RNN). In this figure, X represents the input vector, W_i^h, W_h^h, W_h^o represent the weighting matrices, σ represents the activation function, h represents the hidden layer, z^{-1} represents the lag operator, and y represents the output matrix. In an RNN network, raw data is transformed into information by computing equation (16) for the time t, where X_t is the time series at time t.

$$h_{t+1}=\sigma\left(W_i^h X_t+z^{-1} W_h^h\right) \qquad (16)$$

After computing the hidden layer, h_{t+1} is sent back into the network for re-processing, and the output matrix y_{t+1} is obtained using equation (17).

$$y_{t+1}=\sigma(W_h^o(h_{t+1})) \qquad (17)$$

From equation (17), it can be observed that the dynamic system in the RNN moves with the parameter z^{-1} (Salman, Heryadi, Abdurahman, & Suparta, 2018).

3.2.6. Long-Short Term Memory (LSTM)

The LSTM, which belongs to the Recurrent Neural Network (RNN) family, keeps the data circulating in the network by sending it back to previous layers to be processed again. This architecture is one of the features that distinguishes the LSTM from the RNN. The LSTM architecture, developed by (Hochreiter & Schmidhuber, 1997), consists of four layers and several operational processes (Figure 3). Here X_t is the input sequence at period t, h_{t-1} is the last output vector processed in the LSTM cell, c_{t-1} is the memory vector in the last LSTM cell, C_t is the last memory vector updated, h_t is the last output vector updated, $\oplus$ is the information added or combined in memory, $\otimes$ operator is the information that has undergone a series of replication processes, $tanh$ and σ respectively, are the tangent and sigmoid layers, respectively that remove information from linearity.

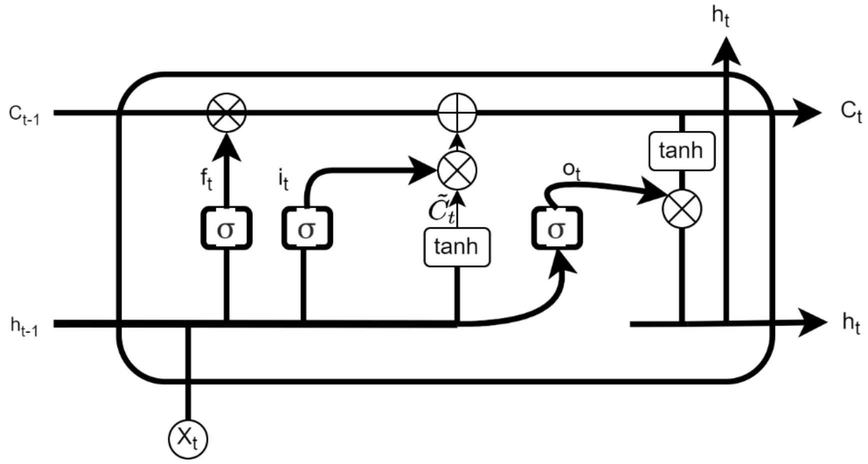

Figure 2. LSTM architecture

In an LSTM network, where the sequence of the input vector is $X = (x_1, \dots, x_t)$, the sequence of the output vector is $y = (y_1, \dots, y_t)$, the activation vector at the input gate is i_t and the memory value that has undergone a series of processes within the LSTM cell is c_t,

$$i_t = \sigma(W_i X_t + U_t h_{t-1} + b_i) \tag{18}$$

$$\widetilde{C_t} = \tanh(W_c X_t + U_c h_{t-1} + b_c) \tag{19}$$

Where, σ, denotes the logistic sigmoid function. The activation vector i_t, at the input gate allows new information to be stored in the LSTM cell. $\widetilde{C_t}$, adds a set of information vectors activated by each tangent function to the cell state. This information is then transferred to the forget gate. Since the information vector belonging to the forgetting gate being is f_t , some of the values of the information vector here are subjected to the cleaning process. After a series of cleaning processes, the cell state c_t cell state appears in the LSTM cell.

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{20}$$

$$C_t = i_t + \widetilde{C_t} + f_t + c_{t-1} \quad (21)$$

The cell state vector c_t is transported to the output gate without further purification within the LSTM cell. At the output gate, the output vector o_t is calculated using the sigmoid activation function.

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + V_0 C_t + b_0) \quad (22)$$

Then, by replicating the o_t vector with the tangent activation function and the C_t cell state vector, creates the updated final output vector h_t of the LSTM cell is generated as shown equation (23).

$$h_t = o_t \tanh(C_t) \quad (23)$$

The weight matrices W_i, W_c, W_f, W_o, U_i, U_c, U_f, U_o, and V_o, and the bias vectors b_i, b_c, b_f, and b_o take values between 0-1. The values of these vectors are obtained in the training stage (Masum, Liu, & Chiverton, 2018).

3.2.7. Gated Recurrent Unit (GRU)

The main problem with typical recurrent networks is that they have vanishing gradients (Bengio, 1993). To address this issue, Chung, Gulcehre, Cho, & Bengio, (2014) developed the GRU architecture, which has a similar structure to the LSTM but is less complex. In the GRU architecture, the algebraic operations are performed on the update (z_t) and reset (r_t) gates instead of the input, forget and output gates in LSTM. The update gate determines how much X_t and h_{t-1} will be used in the next cell, while the forget gate decides how much of the previous outputs will be forgotten. W is updated for each memory, and the calculation of the other memory is started simultaneously. The calculation steps for these operations are shown in equations (24), (25), (26), and (27):

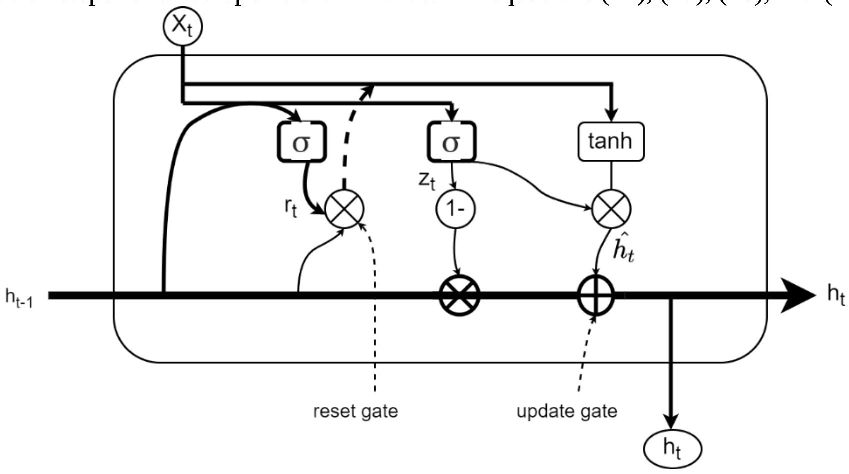

Figure 3. GRU architecture

$$z_t = \sigma(W_z * [h_{t-1}, X_t]) \quad (24)$$

$$r_t = \sigma(W_r * [h_{t-1}, X_t]) \quad (25)$$

$$\tilde{h}_t = tanh(W_{\tilde{h}} * [r_t * h_{t-1}, X_t]) \quad (26)$$

$$h_t = (1 - z_t) * \ h_{t-1} + z_t * \tilde{h}_t \quad (27)$$

3.2.8. Convolutional Neural Networks (CNN)

CNN is a type of artificial neural network that is commonly used in image processing techniques, but it can also be used in time series analysis to identify hidden non-linear structures in raw data. The architecture of a CNN mainly consists of three layers: convolutional, pooling, and fully connected. The convolutional layer is responsible for extracting features from the input data. In the pooling layer, each set of features is collected using a kernel by filtering the features extracted from the convolutional layer. The resulting features are then passed to the fully connected layer after a process of flattening and labelling, which produces outputs according to the length of the input series. The mathematical operations of these stages are shown in equations (28), (29) and (30).

$$C_r(t) = f\left(\sum_{i=1}^{l}\sum_{j=1}^{k} x(i + s(t-1), j)W_r(i,j) + b(r)\right) \quad (28)$$

Where f represents the activation function, k, l, and N denote to kernel size, filter length and time series length, respectively $x \in \mathcal{R}^{Nxk}$ denotes the input time series or the output of the previous layer, s represents the convolution stride, $C_r(t)$ refers to the tth component of the rth feature map, $W_r \in \mathcal{R}^{lxk}$ and $b(r)$ refer to the weights and bias of the rth convolution filter.

$$P_r(t) = g\Big(C_r\big((t-1)l + 1\big), C_r\big((t-1)l + 2\big), \dots, C_r(tl)\Big) \quad (29)$$

The function g in equation (29) represents the pooling strategy, with averaging or maximum pooling being the most commonly used methods.

$$O(j) = f(\textstyle\sum_{i=1}^{N} z(i)W_f(i,j) + b_f(j)), j = 1,2, \dots, n \quad (30)$$

Equation (30) shows that z denotes the final feature map in the feature layer, b_f represents the bias of the output layer and $W_f \in \mathcal{R}^{Nxn}$ refers to the connection weights between the feature layer and the output layer.

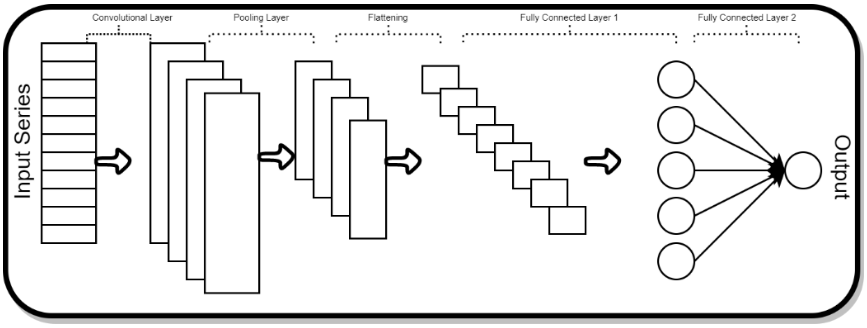

Figure 4. CNN architecture

As shown in Figure 4, the architecture of a CNN consists of multiple convolutional and pooling layers, with the output of each layer being fed as input to the next layer. The fully connected layer processes the final output and generates predictions (Lewinson, 2020).

3.3. Performance comparisons

Four criteria were used to evaluate the performance of the established models: MAPE, RMSE, R^2 and MDA. While the first three criteria are used to assess point estimation, MDA measures the directional accuracy of the predicted value. The calculations for these criteria are expressed in equations (31), (32), (33) and (34).

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| * 100 \quad (31)$$

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{n}} \quad (32)$$

$$R^2 = 1 - \frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{n}(y_t - \mu)^2} \quad (33)$$

$$MDA = \frac{1}{N}\sum_{t=1}^{n} 1_{sign}(Y_t - Y_{t-1}) == sign(\hat{Y}_t - Y_{t-1}) \quad (34)$$

4. Results

This section presents the results of our study, including descriptive statistics, stages of model development, and parameters used in the models. We used 70% of the data set for model training and the remaining 30% for model validation. The exchange rate data used in the study was obtained from Yahoo Finance and all analyses were performed in Python 3.9 environment. The data frequency was chosen as the business day for statistical models. A Box-Cox transformation was applied to the data during preprocessing (lambda = -0.5) for statistical models, and the data were normalised to the 0-1 range for machine learning methods.

Our analysis shows that the currencies of the fragile five countries depreciated between 36.28% and 856% in the nine years studied after the Federal Reserve announced the end of monetary expansion. Table 1 shows the descriptive statistics for the exchange rate data used in the analysis.

Table 1. Descriptive Statistics

	Train Set-Test Set	Minimum	Maximum	Standart Deviation	Mean	Median
USDBRL	1601-687	2.1219 (05/06/2013)	5.9234	1.0487	3.7440	3.6446
USDIDR	1601-687	9792.5 (06/06/2013)	16678	1119.4	13485	13663

	Train Set-Test Set	Minimum	Maximum	Standart Deviation	Mean	Median
USDINR	1601-687	56.581 (04/06/2013)	77.051	4.8506	67.769	68.125
USDTRY	1601-687	1.8572 (14/06/2013)	17.780	2.6630	4.7622	3.7591
USDZAR	1601-687	9.6296 (19/09/2013)	19.071	1.9493	13.606	13.900

Source: Our calculations based on pandas library

We used raw data (transformed with Box-Cox) for the Naïve, Theta, and Exponential Smoothing methods, as the stationarity assumption was not necessary. We determined the periodicity value for the Theta and Exponential Smoothing methods using Fast Fourier Transform and autocorrelation functions. The predictions were based on the smallest mean absolute percentage error (MAPE) value, which was then verified using an optimisation algorithm. The MAPE criterion was chosen because of its significant response to small deviations. Table 2 and Table 3 show the hyperparameters used for the Exponential Smoothing and Theta models, respectively.

Table 2. The Results of Parameters of Exponential Smoothing Models

	Trend Mode	Seasonality Mode	Seasonal Period	Smoothing Level (α)	Smoothing Trend (β)	Smoothing Seasonal (γ)	SSE (Sum of Squared Errors)
USDBRL	Multiplicative	Multiplicative	255	0.914	9.52e-07	4.34e-08	0.041
USDIDR	Multiplicative	Additive	243	0.959	0.0001	0.0001	0.001
USDINR	Additive	Additive	247	0.995	0.0001	9.76e-05	0.000
USDTRY	Additive	Multiplicative	286	0.999	6.62e-06	2.00e-08	0.044
USDZAR	Multiplicative	Additive	243	0.995	0.0006	9.07e-05	0.010

Table 3. The Results of Parameters of Theta Models

	Seasonality Mode	Theta	Seasonal Period	Smoothing Level (α)	Initial Level	BIC (Bayesian Information Criterion)
USDBRL	Multiplicative	-3.05	255	0.914	0.642	-16854
USDIDR	Multiplicative	1.39	243	0.937	1.980	-32112
USDINR	Multiplicative	-1.83	248	0.995	1.737	-24267
USDTRY	Additive	5.43	286	1	0.555	-17261
USDZAR	Additive	3.01	256	0.995	1.371	-19177

Before determining the parameters of the ARIMA model, the stationarity of the exchange rates of the fragile five countries was tested. The Augmented Dickey-Fuller method was used to perform stationarity tests by adding constant and trend variables to the regression equation. The maximum values of (p,d,q) for all ARIMA models were determined as (5,1,5), and the model was selected based on the lowest BIC value. After determining the model, the Ljung-Box (Q) test was used to detect the presence of autocorrelation between errors. It was found that the identified models passed the Ljung-Box test, but the issue of heteroskedasticity could not be resolved. Furthermore, it was also found that the USDZAR parity exhibited a random walk process (Table 4).

Table 4. The Results of Parameters of Arima Models

	ADF Statistics at Level	**ADF Statistics (First Difference)**	**ARIMA Parameters (p, d, q)**	**Bayesian Information Criterion**	**Ljung-Box (Q) Statistics**	**Heteroskedasticity (Probability)**
USDBRL	-1.463 (0.55)	-9.833 (0.00)	(0, 1, 1)	-12037	0.00 (0.97)	0.47 (0.00)
USDIDR	-3.017 (0.04)	-22.52 (0.00)	(0, 1, 1)	-27389	0.01 (0.94)	0.22 (0.00)
USDINR	-1.852 (0.35)	-10.43 (0.00)	(2, 1, 0)	-19562	0.01 (0.93)	0.38 (0.00)
USDTRY	-0.518 (0.88)	-6.742 (0.00)	(0, 1, 1)	-12442	0.00 (0.98)	1.33 (0.00)
USDZAR	-2.005 (0.28)	-19.48 (0.00)	(0, 1, 0)	-14344	0.00 (0.97)	1.14 (0.12)

For the ADF test, the probability value in parentheses is less than .05, indicating that φ and θ parameters in the equation (14) are significantly differ from 0. For the Ljung-Box (Q) test, the values in parentheses greater than .05, indicate that there is no autocorrelation between errors.

Source: Our calculations based on darts library

Each of the traditional statistical methods was optimised separately, while no attempt was made to optimise the parameters of the RNN-LSTM-GRU-CNN networks. The primary objective was to test the argument that the prediction error of any machine learning approach modelled with the same parameters is smaller than the errors in traditional statistical methods. For each neural network model, the number of hidden layers was set to 2, the look-back value was set to 5, the batch size was set to 32, the number of epochs was set to 400, and the total dropout rate was set to 0.4. Adam optimisation algorithm was used for all architectures, MSE was preferred as the loss function. For the RNN, LSTM, and GRU architectures, the number of neurons was 128-64. For the CNN architecture, the number of filters was 128, the kernel size was 8, the number of strides was 4, the pool size was 1, the number of flatten layers was 1, and the number of dense layers was 8. The architectures of the machine learning models are shown in Figure 5, based on the above statements.

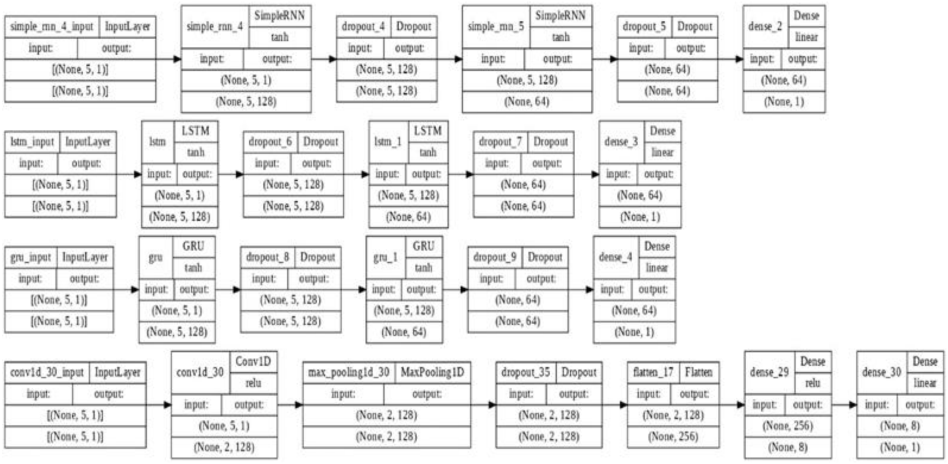

Figure 5. Architectures of ML-based models

Source: our presentation basen on keras library

4.1. Performance Evaluation

The performance evaluation of each model was based on four criteria: MAPE, RMSE, R^2, and MDA. The statistical results for these criteria are presented in Tables 5, 6, 7, and 8. The success of the prediction models was determined by the lowest MAPE and RMSE values, and the highest R^2 and MDA values.

Table 5. Performances of the Models by MAPE (%) Value

	USDBRL	USDIDR	USDINR	USDTRY	USDZAR
Naïve Drift	18.88	4.63	2.93	15.70	6.67
Exponential Smoothing	13.07	3.47	1.86	7.45	5.77
Theta	6.91	2.09	1.90	7.63	5.91
Arima	17.41	3.33	2.66	9.35	6.82
RNN	1.34	1.19	0.59	1.44	0.78
LSTM	0.81	0.33	0.32	4.33	1.13
GRU	1.25	0.47	0.44	1.60	0.96
CNN	1.71	1.28	0.65	1.96	1.11

Source: Authors' calculations

Table 6 Performances of the Models by RMSE Value

	USDBRL	USDIDR	USDINR	USDTRY	USDZAR
Naïve Drift	0.845	1217.5	2.321	1.068	1.533
Exponential Smoothing	0.641	494.89	2.009	1.216	1.447
Theta	0.591	483.35	2.013	1.097	1.464
Arima	0.808	799.97	2.465	4.133	1.554
RNN	0.086	176.05	0.455	0.381	0.147
LSTM	0.056	74.67	0.310	0.399	0.212
GRU	0.082	109.46	0.395	0.331	0.194

	USDBRL	USDIDR	USDINR	USDTRY	USDZAR
CNN	0.112	212.95	0.556	0.400	0.234

Source: Authors' calculations

Table 7. Performances of the Models by R^2 Value

	USDBRL	USDIDR	USDINR	USDTRY	USDZAR
Naïve Drift	0.483	0.000	0.313	0.806	0.037
Exponential Smoothing	0.422	0.009	0.261	0.746	0.035
Theta	0.430	0.047	0.247	0.797	0.051
Arima	0.443	0.000	0.295	0.873	0.041
RNN	0.998	0.991	0.996	0.991	0.995
LSTM	0.992	0.973	0.981	0.990	0.987
GRU	0.986	0.949	0.964	0.981	0.974
CNN	0.986	0.944	0.971	0.982	0.974

Source: Authors' calculations

Table 8. Performances of the Models by MDA (%) Value

	USDBRL	USDIDR	USDINR	USDTRY	USDZAR
Naïve Drift	51.1	48.9	47.7	57.4	47.9
Exponential Smoothing	48.8	48.6	51.8	49.9	50.4
Theta	49.5	51.2	47.5	51.2	51.8
Arima	51.0	48.9	47.8	57.3	47.9
RNN	89.7	82.4	84.7	81.6	84.9
LSTM	69.1	69.1	69.1	70.1	68.7
GRU	58.8	55.3	57.1	60.3	56.3
CNN	57.4	51.9	57.8	60.1	58.7

Source: Authors' calculations

The tables show that there was no significant difference in the estimation performance of the statistical methods. However, among the machine learning models, RNN and LSTM had the most accurate predictions. In addition, Table 8, which shows the directional accuracy of the predicted values, revealed that the RNN architecture had a remarkably high success rate compared to all models. These models are also visually shown in Figure 6.

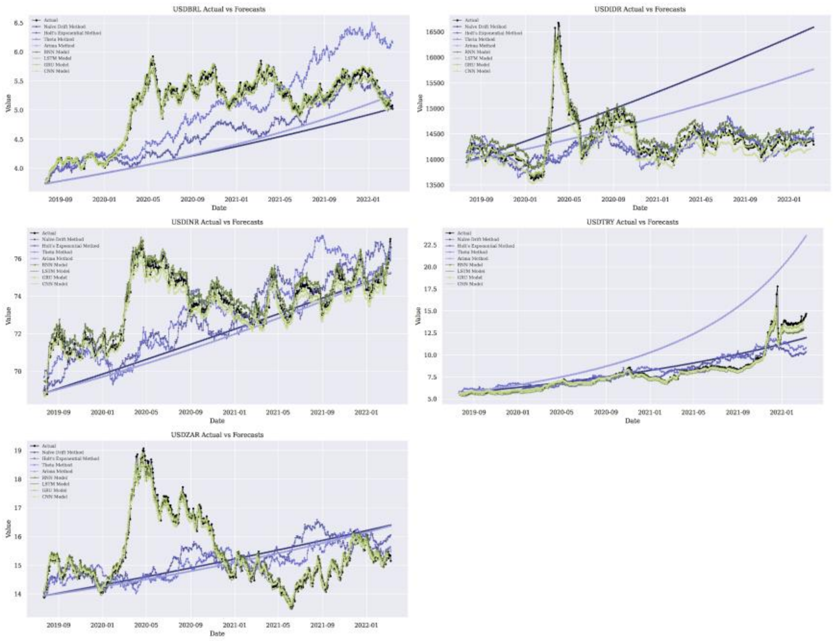

Figure 6. Graphical Presentation of Forecasting Results
Source: our presentation based on python environment

Looking at the performance criteria tables and Figure 6, it is clear that the statistical models provide information primarily on the trend of the series. Although this trend-based approach can create unforeseen difficulties for monetary authorities and financial investors, machine learning techniques have been observed to provide reliable predictive capabilities that can significantly benefit both parties. In particular, LSTM and RNN architectures have been shown to produce more robust results than GRU and CNN architectures. This difference can be attributed to the sensitivity of GRU and CNN architectures to hyperparameters. Nevertheless, weaker architectures can potentially achieve improved direction and point estimation accuracy through optimisation efforts. Ultimately, these findings are critical for informed decision making and improving market efficiency.

5. Conclusions

This study contributes to the existing literature on exchange rate forecasting by comparing the performance of machine learning and statistical models in the Fragile Five countries. Our study used daily exchange rate data from June 2013 to March 2022 to assess the forecasting accuracy of different models. Our results suggest that machine learning models, particularly RNN and LSTM, outperform statistical models in both point and directional exchange rate forecasting. The superior performance of machine learning models can be attributed to their ability to capture nonlinear patterns in the exchange rate data. Despite their extensive optimisation and diagnostic testing, none of the statistical models- including Exponential Smoothing, Theta, and ARIMA- were able to outperform the Naïve method. This result is mainly due to the inability of statistical models to adapt to changing market conditions. In contrast, machine learning models are more adaptive and can learn from new data, making them suitable for forecasting exchange rates in rapidly changing economic environments. Our findings suggest that policymakers and financial institutions in emerging market economies can benefit from adopting machine learning models for exchange rate forecasting. These models can improve the accuracy of monetary policy decisions and reduce the impact of exchange rate shocks. They can also enhance the credibility of financial institutions by providing more reliable exchange rate forecasts. Future research can build on our study by exploring the potential of multivariate machine learning models that incorporate additional economic variables such as VIX and ETF. These variables can improve the accuracy of exchange rate forecasts by capturing the impact of global economic and financial market conditions. In addition, researchers can investigate the effectiveness of machine learning models in forecasting exchange rates during periods of high volatility, such as economic crises. In conclusion, our study highlights the potential benefits of machine learning models for exchange rate forecasting in emerging markets. The use of these models can lead to more accurate and reliable exchange rate forecasts, which can ultimately contribute to better-informed monetary policy decisions and improved financial stability.

REFERENCES

[1] **Akel, V., Kandir, S., Yavuz, Ö. (2015),** ***Dynamic Relationship between Stock Prices and Exchange Rates in Emerging Markets: Evidence from Fragile Five Economies*****.** In *In International Business: Concepts, Methodologies, Tools, and Applications*, 2257-2273, IGI Global. doi:10.4018/978-1-4666-7288-8.ch011;

[2] **Assimakopoulos, V., Nikolopoulos, K. (2000),** ***The theta model: a decomposition approach to forecasting.*** *International journal of forecasting*, 16(4), 521-530;

[3] **Athey, S. (2018),** ***The impact of machine learning on economic. The economics of artificial intelligence.*** *An agenda*, 507-547;

[4] **Bengio, Y., Frasconi, P., Simard, P. (1993),** ***The problem of learning long-term dependencies in recurrent networks.*** *IEEE international conference on neural networks*, 1183-1188, IEEE;

[5] **Bhattarai, S. (2021),** ***Effects of US Quantitative Easing on Emerging Market Economies.*** *Journal of Economic Dynamics and Control*(122)**,** 104031;

[6] **Bianchi, F.M., Maiorino, E., Kampffmeyer, M.C., Rizzi, A., Jenssen, R. (2017),** ***Recurrent Neural Networks for Short-term Load Forecasting: An Overview and Comparative Analysis.*** Springer;

[7] **Chung, J., Gulcehre, C., Cho, K., Bengio, Y. (2014),** ***Empirical evaluation of gated recurrent neural networks on sequence modeling.*** arXiv preprint arXiv:1412.3555.

[8] **Elman, J.L. (1990, March),** ***Finding Structure in Time.*** *Cognitive science,* 14(2), 179-211;

[9] **Fendoğlu, S., Çolak, M.S., Hacıhasanoğlu, Y.S. (2020),** ***Foreign-Currency Debt and the Exchange Rate Pass-Through.*** *Applied Economic Letters***,** 27(8), 657-666;

[10] **Galeshchuk, S., Mukherjee, S. (2017),** ***Deep Networks for Predicting Direction of Change in Foreign Exchange Rates.*** *Intelligent Systems in Accounting, Finance and Management,* 100-110;

[11] **Hersi, K.Y., Koy, A. (2020),** ***The Relationship between Exchange Rates and Stock Markets for the Fragile Five Countries.*** *Journal of International Trade,* Logistics and Law**,** 6(1), 1-13;

[12] **Hochreiter, S., Schmidhuber, J. (1997),** ***Long Short-Term Memory.*** *Neural computation***,** *9*(8), 1735-1780;

[13] **Hyndman, R.J. (2018),** ***Forecasting: Principles and Practice.*** OTexts;

[14] **Jirsik, T., Trčka, Š., Celeda, P. (2019),** ***Quality of Service Forecasting with LSTM Neural Network.*** *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)* (s. 251-260). Arlington: IEEE;

[15] **Kadilar, C., Şimşek, M., Aladağ, H. (2009),** ***Forecasting the Exchange Rate Series with ANN: The Case of Turkey.*** *İstanbul University Econometrics and Statistics e-Journal (9),* 17-29;

[16] **Lewinson, E. (2020),** ***Python for Finance Cookbook: Over 50 recipes for applying modern Python libraries to financial data analysis.*** Packt Publishing Limited;

[17] **Liao, Y. (2017),** ***Machine Learning in Macro-Economic Series Forecasting.*** *International Journal of Economics and Finance,* 9(12), 71-76;

[18] **Masum, S., Liu, Y., Chiverton, J. (2018),** ***Multi-Step Time Series Forecasting of Electric Load Using Machine Learning Models.*** *International conference on artificial intelligence and soft computing***,** 148-159, Springer, Cham;

[19] **Nagpure, A. (2019),** ***Prediction of Multi-Currency Exchange Rates Using Deep Learning.*** *International Journal of Innovative Technology and Exploring Engineering,* 8(6), 316-322;

[20] **Nechio, F. (2014),** ***Fed tapering news and emerging markets.*** *FRBSF Economic Letter (6),* 014-06;

[21] **Salman, A.G., Heryadi, Y., Abdurahman, E., Suparta, W. (2018),** ***Single Layer & Multi-Layer Long Short-Term Memory (LSTM) Model with Intermediate Variables for Weather Forecasting.*** *Procedia Computer Science,* 89-98;

[22] **Spiliotis, E., Assimakopoulos, V., Nikolopoulos, K. (2019),** ***Forecasting with a Hybrid Method Utilizing Data Smoothing, a Variation of the Theta Method and Shrinkage of Seasonal Factors.*** *International Journal of Production Economics,* 209, 92-102;

[23] **Takhtamanova, Y.F. (2010),** ***Understanding Changes in Exchange Rate Pass-Through.*** *Journal of Macroeconomics,* 32(4), 1118-1130;

[24] **Zhang, G.P. (2003),** ***Time Series Forecasting Using a Hybrid ARIMA and Neural Network Model.*** *Neurocomputing,* 50, 159-175;

[25] **Zhang, G., Patuwo, B.E., Michael, Y.H. (2001),** ***A Simulation Study of Artificial Neural Networks for Nonlinear Time-Series Forecasting.*** *Computers&Operation Research,* 28(4), 281-396.