Journal of Advanced Computer Science & Technology, 12 (2) (2024) 41-52

Journal of Advanced Computer Science & Technology

Website: www.sciencepubco.com/index.php/JACST

Research paper

Forex Market Analysis Using Deep Learning Approaches

Kalluri Ram Rohith Reddy[1], Kankanala Kowsick Raja[2], P. Subham[3] and Puspanjali Mohapatra[4*]

[1]*Department of Computer Engineering, IIIT Bhubaneswar*
[2]*Department of Computer Engineering, IIIT Bhubaneswar*
[3]*Department of Computer Science Engineering, NIT Rourkela*
[4]*Department of Computer Science Engineering, IIIT Bhubaneswar*
[*]*Corresponding author E-mail: puspanjali@iiit-bh.ac.in*

Abstract

This paper compares the effectiveness of various deep learning models which includes LSTM (Long-Short Term Memory) and GRU (Gated Recurrent Unit) models. These models use three exchange currency pairs named Euro to US Dollar, British Pound to US Dollar, and Indian Rupee to Japanese Yen for the purpose of training and performance comparison. The analysis is conducted daily according to time zones. Mean Square Error (MSE), Root Mean Square Error (RMSE), and Mean Absolute Error (MAE) performance measures are used to compare different models. According to the observations, the GRU model outperformed the LSTM model in the majority of datasets.

Keywords: *Forex, Forex Market, Indian Rupee, Japanese Yen, British Pound, US Dollar, MSE, RMSE, MAE, LSTM, GRU*

1. Introduction

A decentralised international marketplace where currencies are traded is the forex market commonly referred to as the foreign exchange market [1]. With trillions of dollars being exchanged every day, it is the biggest and most liquid financial market in the world. Banks, Businesses, Governments, and individual traders all participate in the currency market. Forex analysis is crucial for understanding and predicting currency movements. The major currency trading pairs involved are EUR/USD, GBP/USD, INR/JPY [1] etc.Participants in the forex market buy and sell currencies with the intention of making money by the differences in exchange rates [2]. Traders can make predictions about whether a currency will gain or lose value and then take positions appropriately.Forex traders employ different strategies and analytical techniques to make informed trading decisions. Technical analysis involves analyzing historical price data [3] and indicators to identify trends and predict future price movements. Forex Market Researchers uses various statistical Models like ARMA [4], ARIMA [4], ARCH [5], GARCH [5] etc.. to predict the trends followed by the market by finding the relations between various effecting factors. They also use the neural and deep learning models to analyse the complex patterns and complex relations between the factors by using the MLP [6], RBFNN [7], TFLANN [8], FLANN [8], LLWNN [9], LSTM [10], GRU [10][11] models etc. This research focuses on using deep learning models like LSTM and GRU to a variety of currency trading pairs which includes EUR/USD, GBP/USD, and INR/JPY. For the purpose of minimising the error of the model, we combined the Adam optimizer algorithm with the backpropagation technique [11].

2. Long-Short Term Memory Model

LSTM (Long Short-Term Memory) model for forex market analysis revolves around capturing and learning long-term dependencies and patterns in sequential data. In the context of the forex market, LSTM model is designed to analyze historical data of currency pairs and analyze the factors effecting them and make predictions based on the learned patterns.The Input gate selects the data that should be stored in the memory cell based on the current input. The forget gate determines which data from the previous memory cell state has to be erased. The output gate controls which data should be extracted from the memory cell in order to generate forecasts or predictions.An LSTM model may successfully capture long-term dependencies in the forex market data by using these gates.Based on past events, it can learn to see patterns and trends in the currency pairs.In this paper, we have used the Back propagation algorithm along with the adam optimizer to minimize the errors and train the model more effectively.

2.1. Forward Pass:

2.1.1. LSTM Cell State:

$$C_t = f_g \odot C_{(t-1)} + i_g \odot \hat{C}_t \tag{1}$$

The memory of the LSTM model at a given time step is represented by the LSTM cell state (C_t). The forget gate (f_g) regulates how much of the previous cell state is kept while the input gate i_g regulates how much fresh information ($\hat{C}_t$) is introduced to the cell state.

2.1.2. LSTM Hidden State/Output

$$h_{st} = o_g \odot tanh(C_t) \tag{2}$$

The output of the LSTM model at time step t is the LSTM hidden state (h_st) It is computed by adding the output gate (o_g) to the cell state's (C_t) hyperbolic tangent activation function. In addition to guaranteeing that the output is within a certain range, this also captures the pertinent knowledge that the LSTM cell has acquired.

2.2. LSTM gates:

2.2.1. Input Gate:

$$i_g = \sigma(\mathrm{W}_{xi} * x_{it} + W_{hi} * h_{t-1} + b_i) \tag{3}$$

2.2.2. Forget Gate:

$$f_g = \sigma(\mathrm{W}_{xf} * x_{it} + W_{hf} * h_{t-1} + b_f) \tag{4}$$

2.2.3. Output Gate:

$$O_g = \sigma(\mathrm{W}_{xo} * x_{it} + W_{ho} * h_{t-1} + b_o) \tag{5}$$

where, x_it is the input at time step, Weights and biases are denoted by W and b. σ denotes the sigmoid activation function, $\odot$ denotes element-wise multiplication.

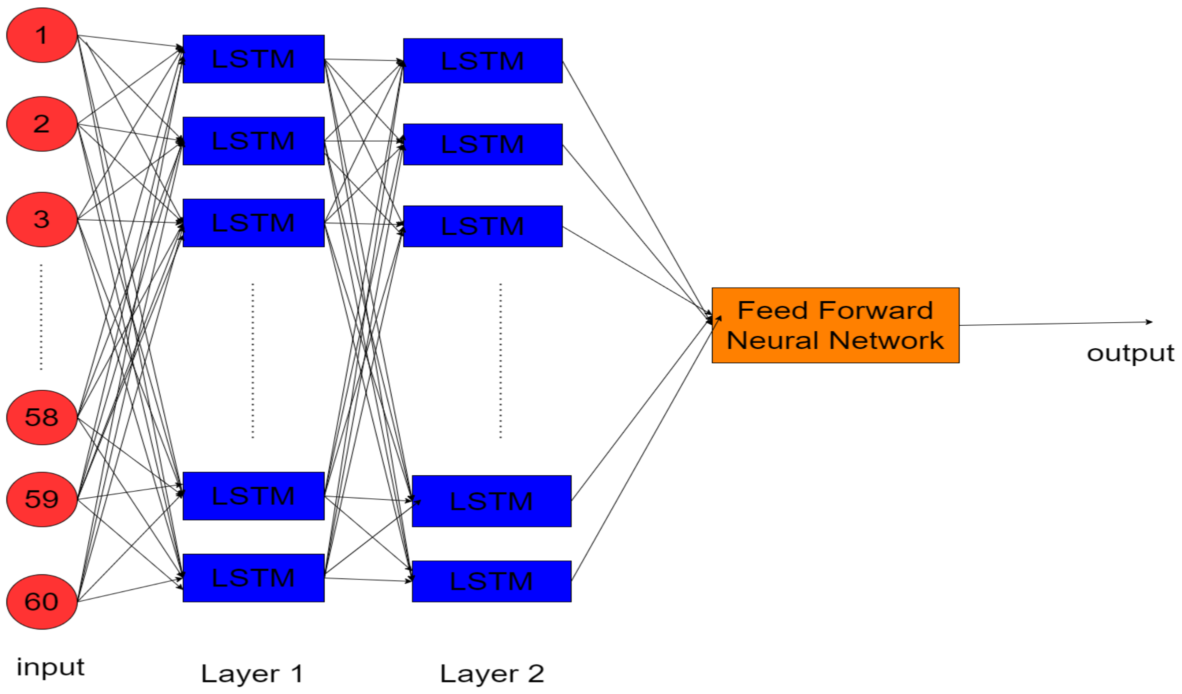

Figure 1: LSTM Model Architecture

3. Gated Recurrent Unit Model

An LSTM model and a GRU (Gated Recurrent Unit) model are both used to analyse the patterns in currency market. GRU models are a subclass of recurrent neural networks (RNNs) that are useful for analysing time series data, such as currency exchange rates, since they can recognise and learn long-term relationships in sequential data.
The availability of gating mechanisms, which let the model to pick and choose which data from earlier time steps to keep or reject, is what makes GRU models unique. An update gate and a reset gate make up these gating devices, which regulate the information flow through the network.
In tasks involving forex market research, including forecasting currency prices, spotting trends, and creating trading signals, GRU models have demonstrated promising outcomes.GRU models may efficiently capture long-term relationships and extract pertinent information from the forex market data by utilising the gating mechanisms. In the currency pairs they can discover patterns, trends, and linkages from which they might extrapolate forecasts.

3.1. Forward Pass:

During forward pass the following phases of operations will occur.

3.1.1. Update State:

$$z_t = \sigma(\mathrm{W}_z.[h_{t-1}, x_{it}]) \tag{6}$$

where W_z is the weight matrix for the update gate, σ is sigmoid activation function, h_{t-1} is the prior hidden state, and x_{it} is present input. z_t is the value of the update gate.

3.1.2. Reset State:

$$r_t = \sigma(\mathrm{W}_r.[h_{t-1}, x_{it}]) \tag{7}$$

The reset gate value is r_t , the weight matrix for the reset gate is W_r, the sigmoid activation function is σ , the prior hidden state is h_{t-1}, and the present input is W_{it}.

3.2. Current Memory Content:

$$\hat{h_t} = tanh(W_h.[r_t \odot h_{t-1}, x_{it}]) \tag{8}$$

where $\hat{h_t}$ denotes the current memory content, W_h denotes the weight matrix for memory content, tanh denotes the hyperbolic tangent activation function, r_t denotes the reset gate value, h_{t-1} denotes the previous hidden state, and x_{it} is the current input.

3.3. Hidden State:

$$h_t = (1 - z_t) \odot\ h_{t-1} + z_t \odot\ \hat{h}_t \tag{9}$$

where h_t denotes the hidden state at time step t, z_t denotes the update gate value, h_{t-1} denotes the hidden state before that, and $\hat{h}_t$ denotes the information now stored in memory.

4. Backpropagation With Adam Optimizer

In the field of machine learning particularly forex market analysis, the back propagation algorithm with the Adam optimizer is a commonly used method. It is frequently used to train neural networks that use past market data to make predictions or choices.
Error propagation backwards via the network is referred to as back propagation. In order to determine how much each weight contributes to the total mistake, it calculates gradient of an loss function with respect to the each weight. The procedure effectively computes these gradients by propagating the mistakes from output layer to the input layer using the chain rule from calculus.

A variation of the gradient descent optimisation technique that adjusts the learning rate of the each weight based on its prior gradients is called the Adam optimizer. In order to achieve quicker convergence and greater generalisation, it combines the ideas of momentum and adaptive learning rates.

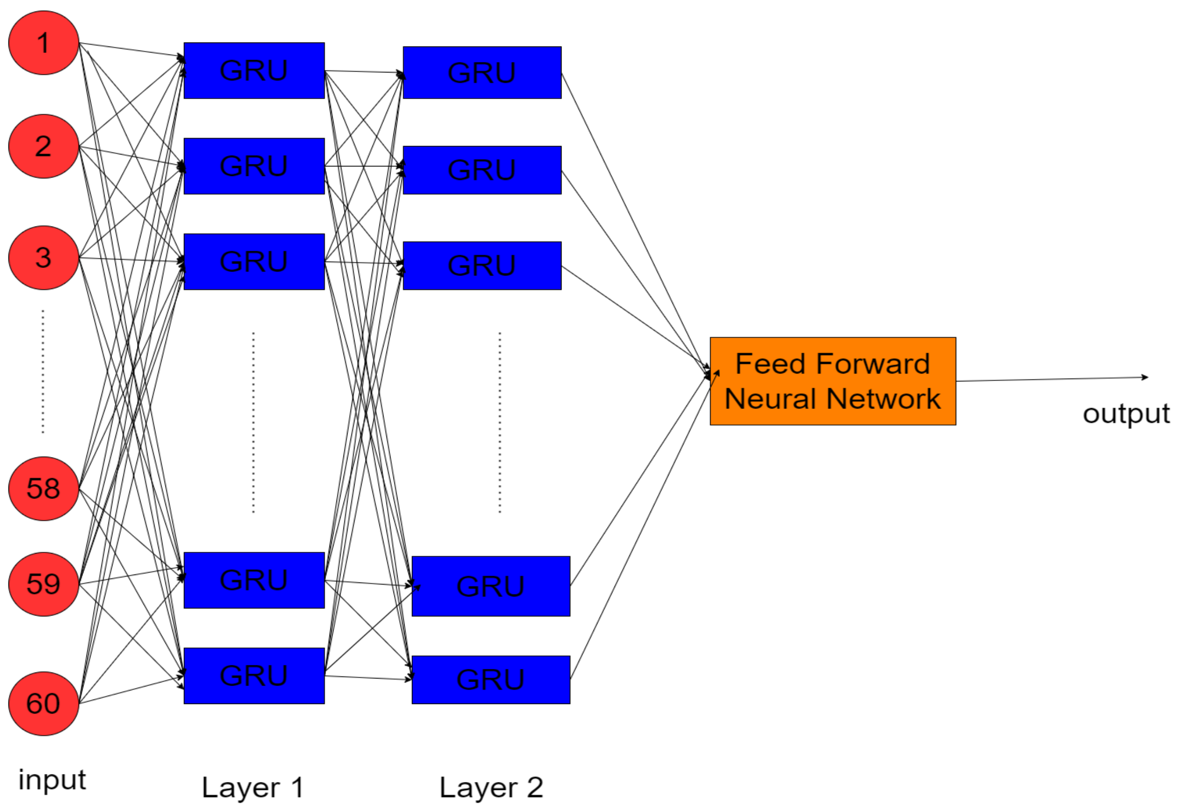

Figure 2: GRU Model Architecture

4.1. Loss Calculation

4.1.1. Mean Squared Error (MSE):

$$L = (1/N) * (\sum_{i=1}^{N} (y_i - \hat{y}_i)^2) \tag{10}$$

The loss function measures the discrepancy between the predicted value $\hat{y}_i$ and true target value y_i for each training sample. MSE is a common choice for regression problems, where the squared differences are averaged over all samples.

4.2. Backward Pass (Backpropagation):

4.2.1. Output Layer Error:

$$\delta_{out} = (\delta_L)/(\delta z_{out}) = (\delta_L)/(\delta a_{out}).f'(z_{out}) \tag{11}$$

The loss with regard to output layer's weighted sum, z_{out},is represented by the error, δ_{out}. It is calculated by dividing the derivative of an loss relative to the output,a_{out}, by the derivative of the activation function, f at z_{out}.

4.2.2. Hidden Layer Error:

The error δ_i for a hidden layer neuron i is calculated based on the error δ_{i+1} of the next layer, the derivative of weighted sum z_{i+1} with respect to z_i, and the derivative of the activation function f at z_i. This error is then back propagated through the network.

4.2.3. Weight Gradient:

$$\delta_L/\delta w_{ij} = \delta_i.\delta x_j \tag{13}$$

This equation is used to determine the gradient of the loss with regard to the weight w_{ij} that links neuron j in the layer below to neuron i in the layer above.

4.2.4. Bias Gradient:

$$\delta_L/\delta \mathrm{b}_i = \delta_i \tag{14}$$

5. 5. Weight Update (Adam Optimizer):

5.1. Moving Average of Gradients:

5.1.1. First Moment Estimate:

$$m_t = \beta_i . m_{t-1} + (1-\beta_i).(g_t) \tag{15}$$

The gradients g_t measured up to time step t are averaged exponentially to get the initial moment estimate, m_t. It uses the momentum term β_1 to keep track of previous gradients.

5.1.2. Second Moment Estimate:

$$v_t = \beta_s . v_{t-1} + (1-\beta_s).(g_t)^2 \tag{16}$$

An exponentially decaying average of the squared gradients g_t^2 measured up to time step t makes up the second moment estimate v_t . It explains how big the gradients are.

5.2. Bias Correction:

5.2.1. Corrected First Moment:

$$\hat{m}_t = m_t/(1-(\beta_i)^t) \tag{17}$$

The first moment estimate m_t is biased towards zero at the beginning of training. To address this, a bias correction factor (1 - $\beta_i{}^t$) is applied to normalize the estimate, where t represents the current time step.

5.2.2. Corrected Second Moment:

The second moment estimate v_t, like the first moment estimate, is initially biased towards zero. The estimate is modified using the bias correction factor, which is equal to (1 - $\beta_s{}^t$) .

5.2.3. Weight Update:

where,
The weight that links neuron j from the previous layer to neuron i in the current layer is represented by the string w_{ij} .
The learning rate, η, defines the weight update's step size.
$\sqrt{\hat{v}_t}$ represents the square root of the corrected second moment estimate, which acts as a scaling factor.
$\hat{m}_t$ corresponds to the bias-corrected first moment estimate, which incorporates the momentum term.
ε is a small constant (e.g., 10^{-8}) added to the denominator for numerical stability.

6. Preprocessing Of Datasets And Study Of The Performance Of LSTM And GRU Models

Dataset preprocessing involves getting the data ready before putting it into a machine learning model. It is essential for enhancing the model's functionality and accuracy. Data cleaning, feature scaling, addressing missing values are a few popular preprocessing techniques.

Data cleaning entails eliminating or fixing any discrepancies, outliers, or mistakes in the dataset. This stage guarantees the reliability and accuracy of the data. In order to cope with any incomplete or unavailable data points, handling missing values is crucial. To do this, either the rows must be eliminated or methods like mean imputation or interpolation must be used to fill in the missing information.

The daily Forex data for the EUR to USD, GBP to USD,and INR to JPY exchange rates are regarded as experimental data in this case. Here, predictions are being made based on data collected every day for span of atleast a year. All inputs are normalised to fall between [0, 1] using the formulae below.

$$x_{\text{scaled}} = \frac{x_{\text{orig}} - x_{\text{minimum}}}{x_{\text{maximum}} - x_{\text{minimum}}} \tag{20}$$

where,
X_{scaled} is the scaled value after normalization,
x_{orig} is the original value,
x_{minimum} and x_{maximum} are the daily minimum and maximum values.
Here,eachdatasetissplitintoatrainandtestdatasetinordertotrainthemodelandevaluateitsperformance.Thetableaboveincludesthedurationandsampl

The performance of the model is then determined using the trained model by forecasting daily against the testing dataset for span of atleast a year. The suggested Models effectiveness is evaluated using the Mean Absolute Error, Mean Square Error and Root Mean Square Error. The described measurements formulas are as follows:

6.1. Mean Square Error:

$$\text{MSE} = \left(\frac{1}{N}\right) \cdot \sum (Y_{\text{actual}} - Y_{\text{predicted}})^2 \tag{21}$$

Where,

he overall number of observations is N.

Σ denotes the summation symbol, indicating that you sum up the squared differences for each pair.

$(Y_{\text{actual}} - Y_{\text{predicted}})^2$ with respect to a certain pair denotes the squared difference between the expected value Y_{actual} and the predicted value $Y_{\text{predicted}}$.

6.2. Root Mean Square Error:

$$\text{RMSE} = \sqrt{\frac{\Sigma(Y_{\text{actual}} - Y_{\text{predicted}})^2}{N}} \tag{22}$$

Where,
The overall number of observations is N.
The value that the model predicts is $Y_{\text{predicted}}$.

Y_{actual} denotes the true observed value.

6.3. Mean Absolute Error:

$$\text{MAE} = (1/N) * (Y_{\text{actual}} - Y_{\text{predicted}}) \tag{23}$$

Where,
The overall number of observations is N.

The model's prediction is represented by the value $Y_{\text{predicted}}$.

The true value of the observation is denoted by Y_{actual}.

7. Observations

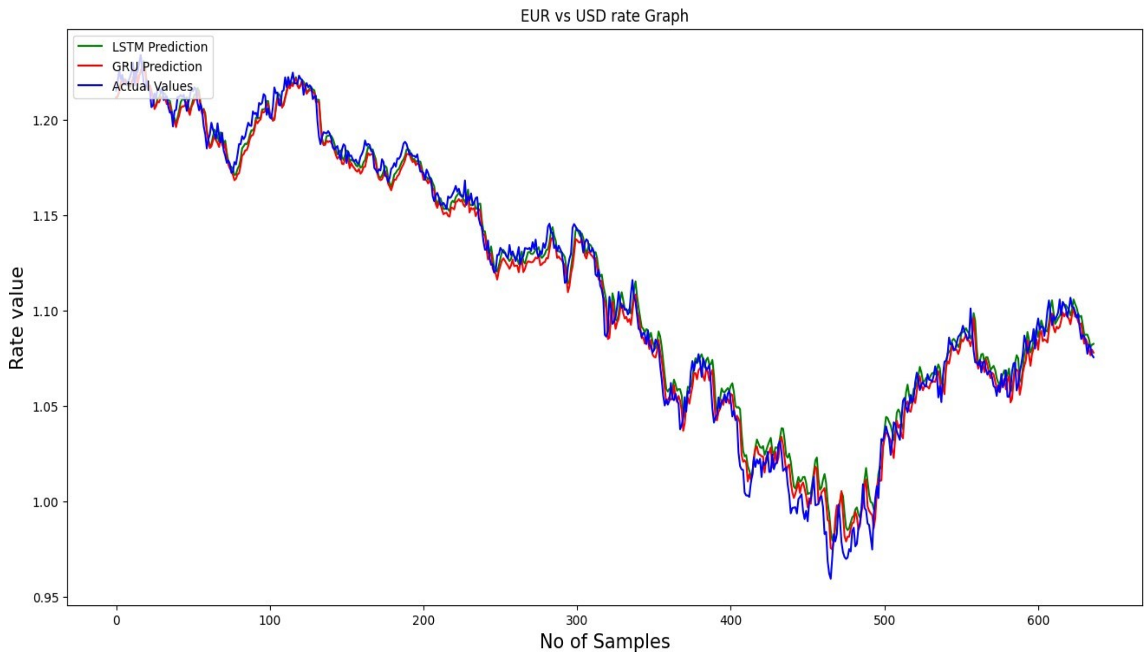

Figure 3: Two Years span Prediction during Testing of LSTM and GRU Models(Euro to US Dollar)

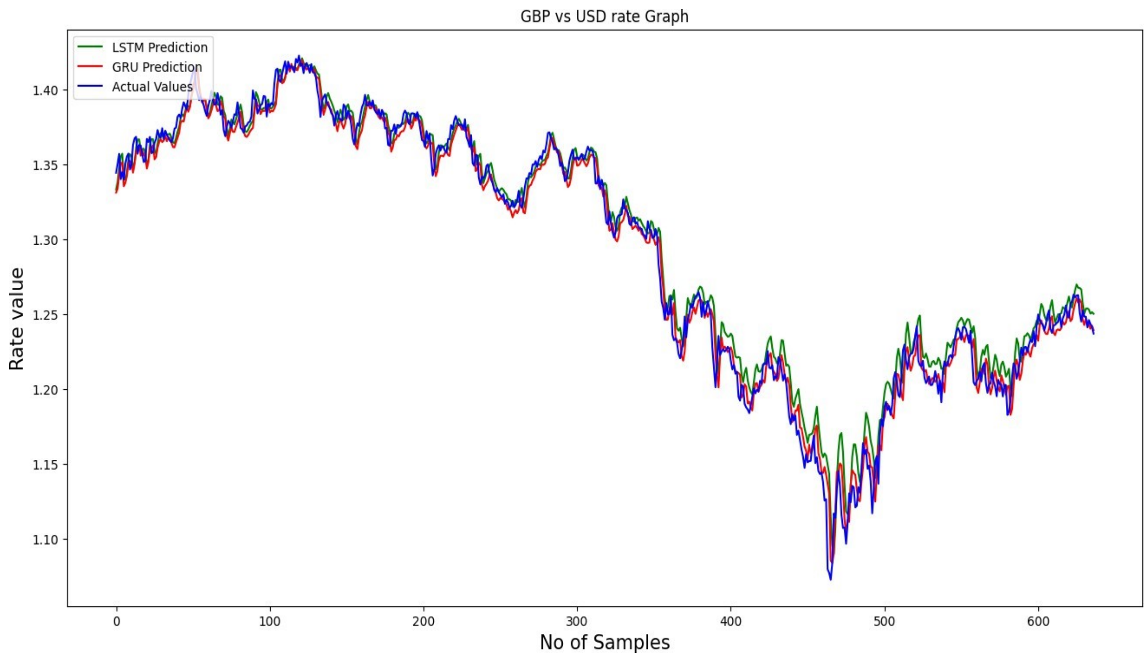

Figure 4: Two Years span Prediction during Testing of LSTM and GRU Models(Britain Pound to US Dollar)

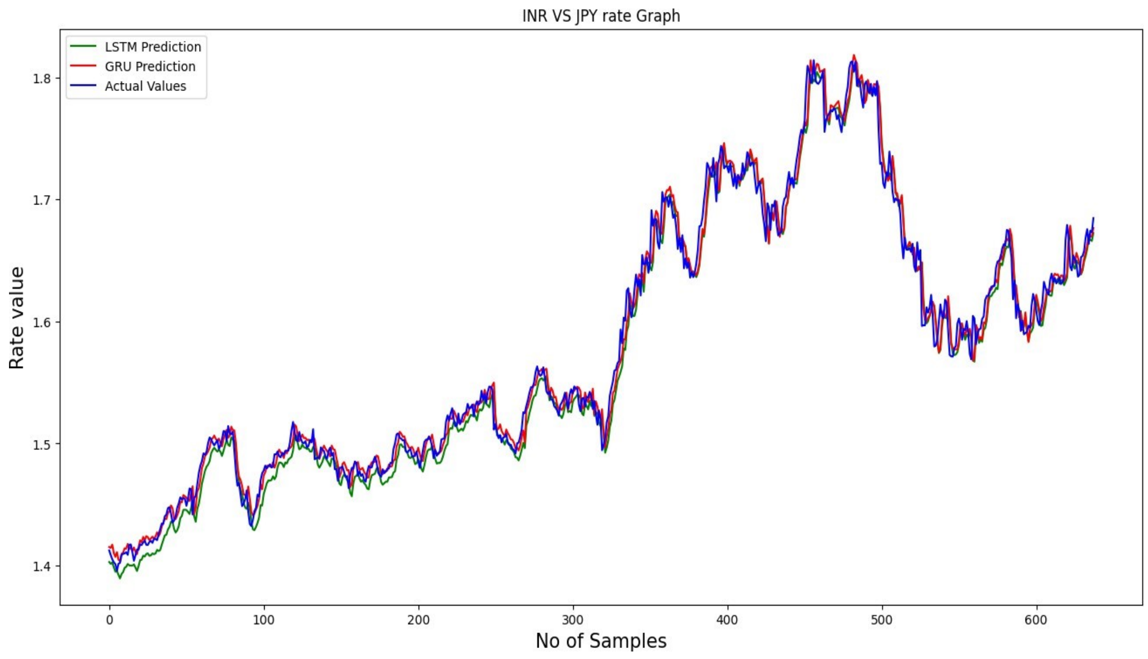

Figure 5: Two Years span Prediction during Testing of LSTM and GRU Models(Rupee to Japaneese Yen))

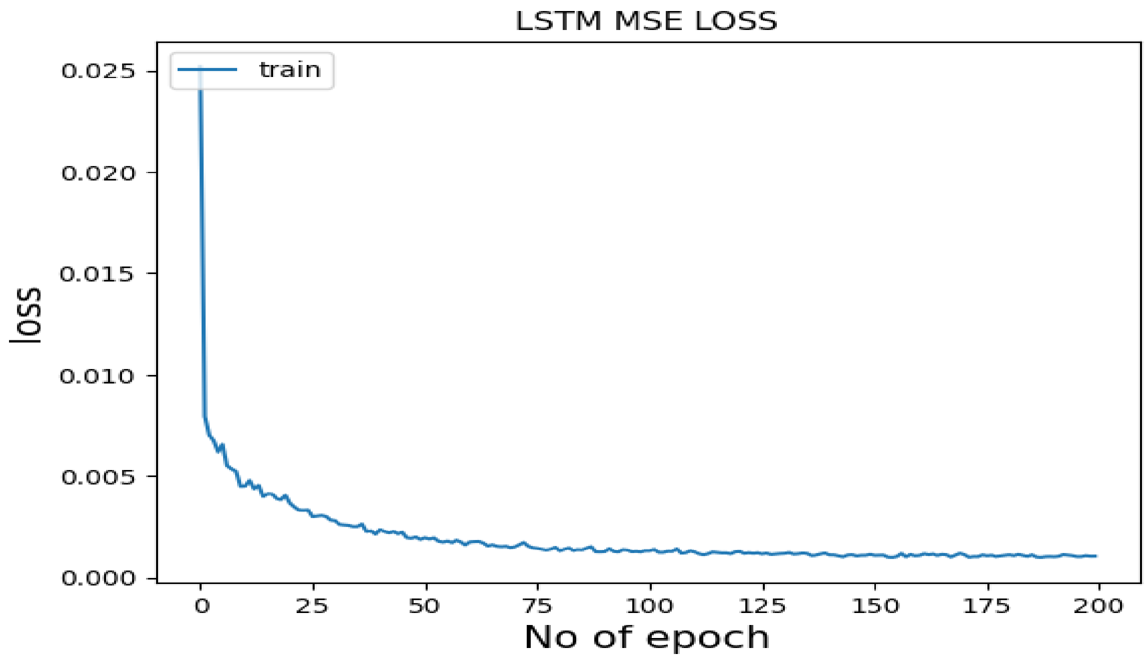

Figure 6: MSE Graph during Training of LSTM Model (Euro to US Dollar)

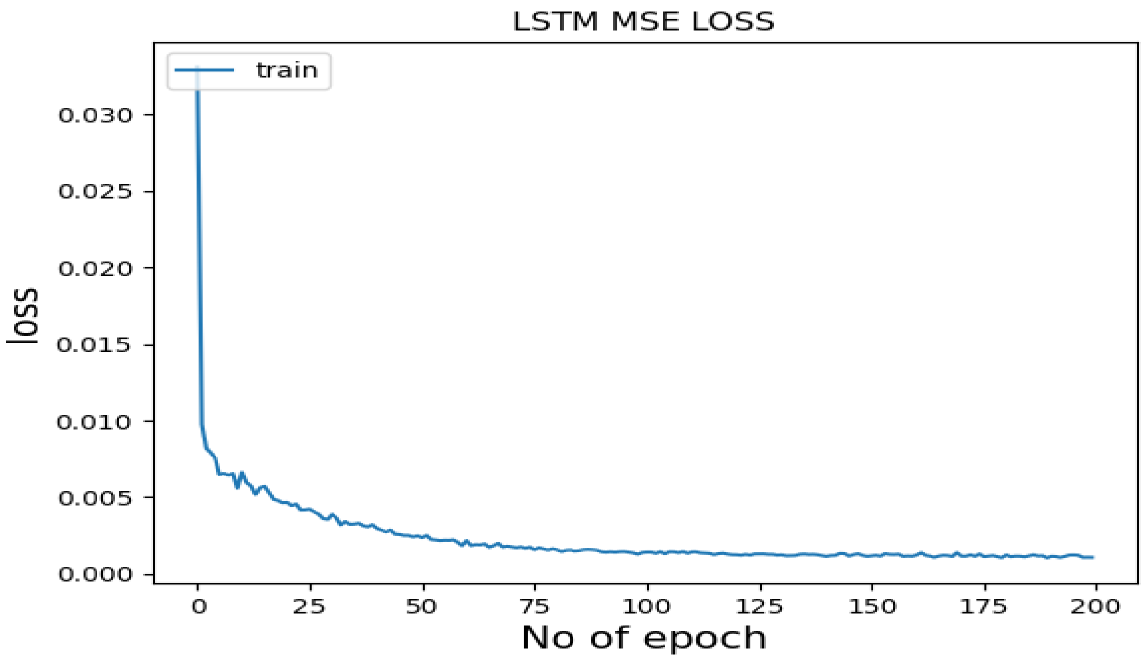

Figure 7: MSE Graph during Training of LSTM Model (Britain Pound to US Dollar)

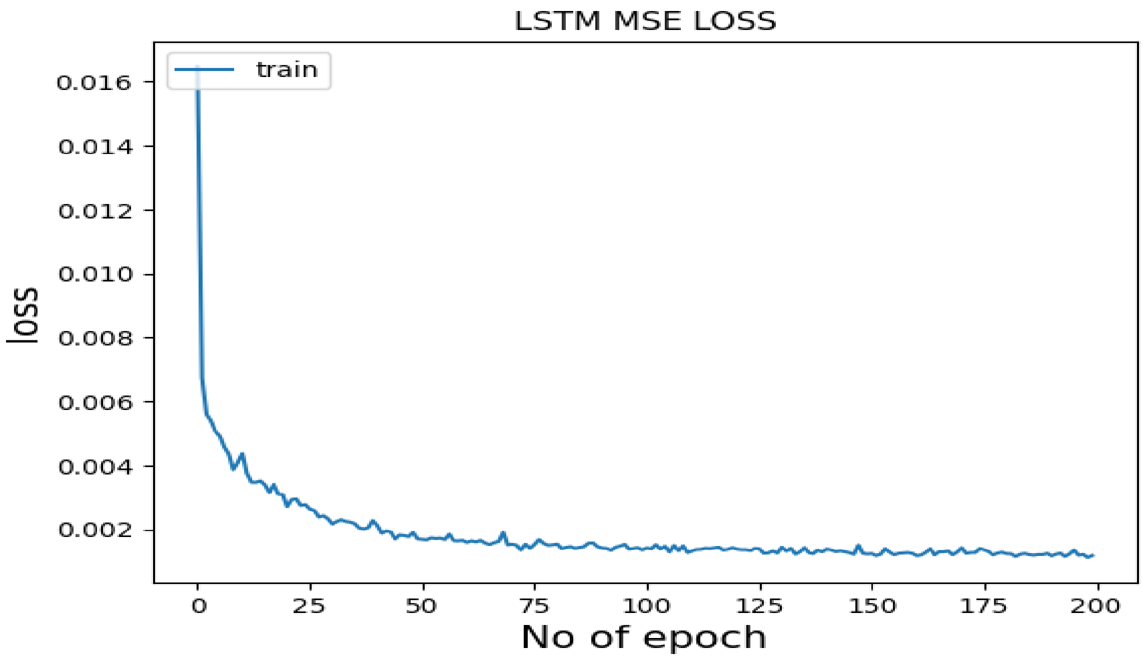

Figure 8: MSE Graph during Training of LSTM Model (Rupee to Japaneese Yen)

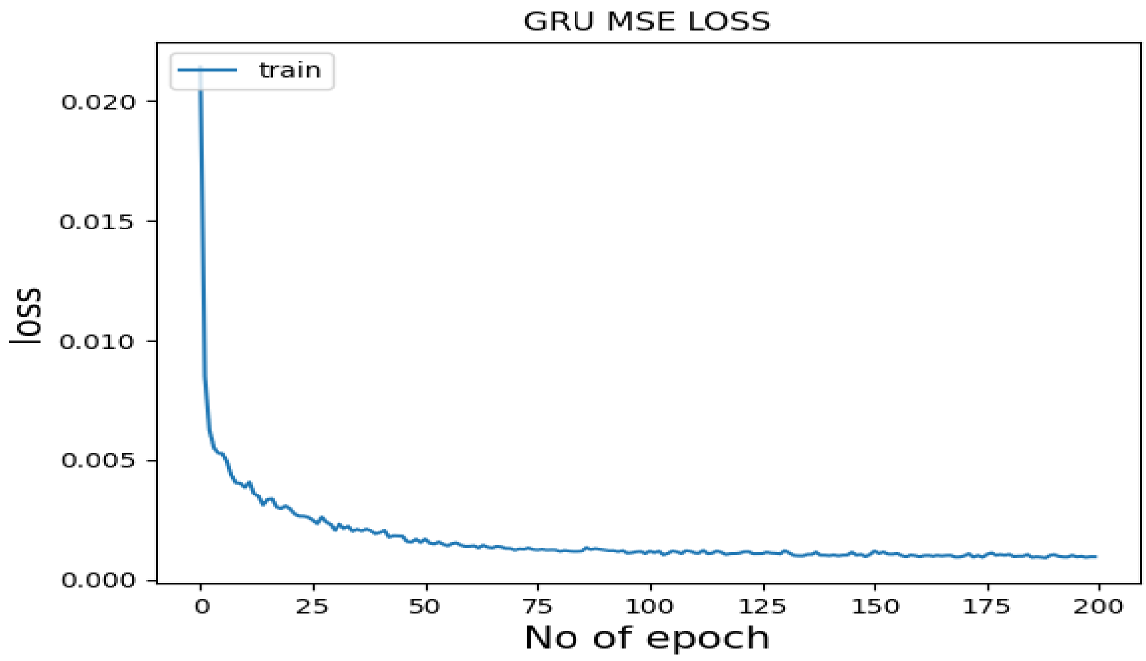

Figure 9: MSE Graph during Training of GRU Model (Euro to US Dollar)

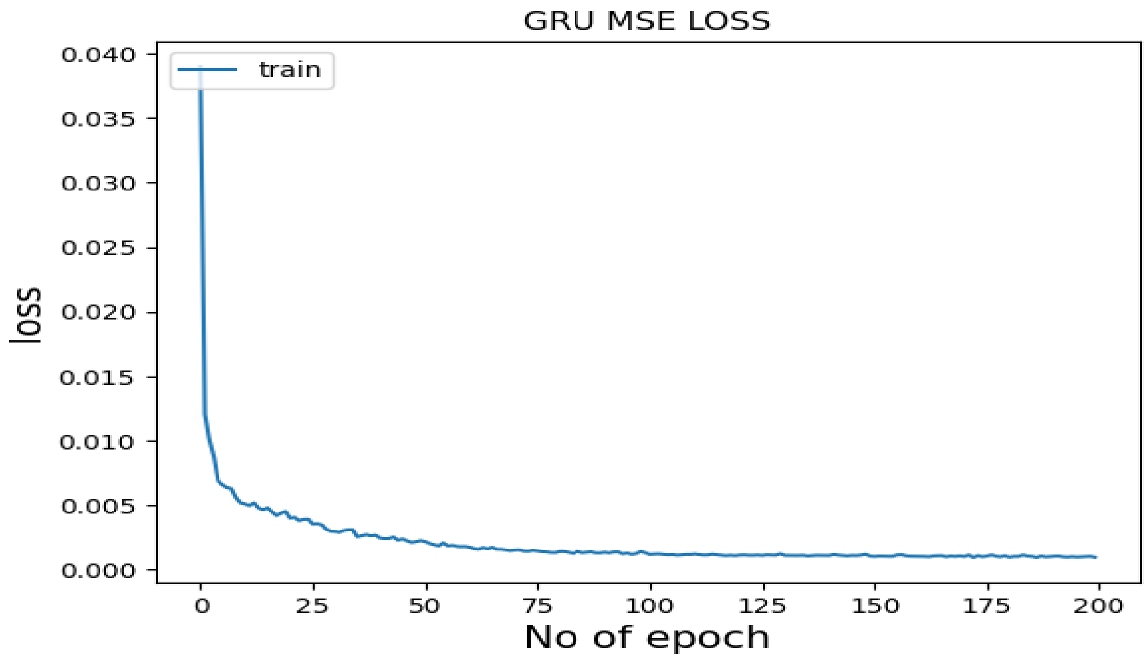

Figure 10: MSE Graph during Training of GRU Model (Britain Pound to US Dollar)

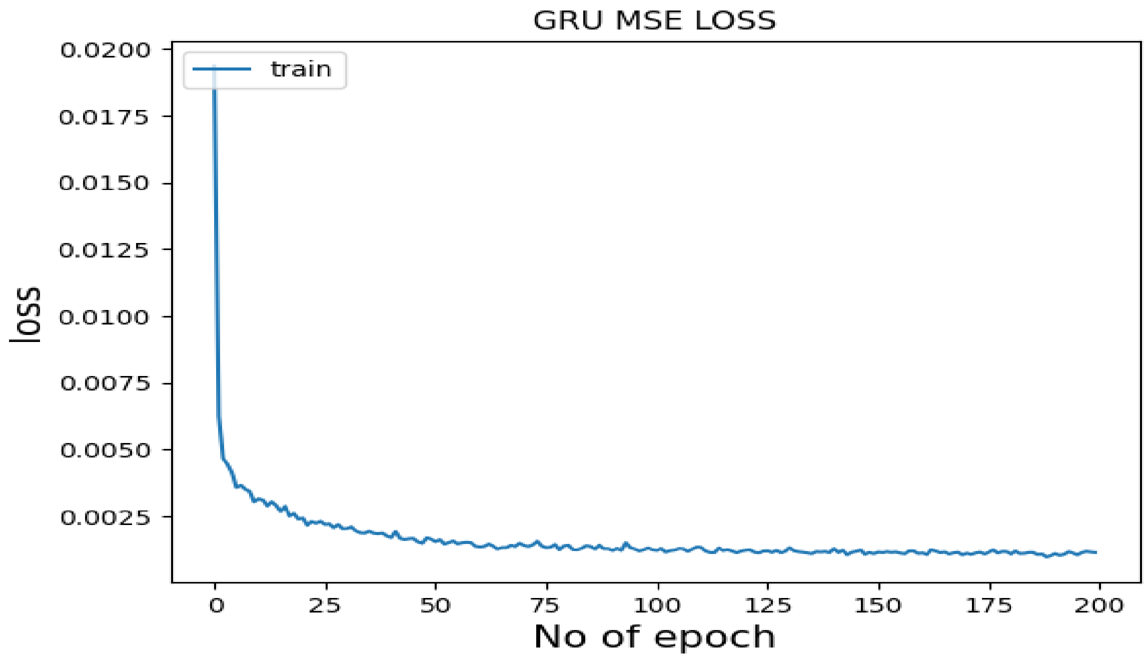

Figure 11: MSE Graph during Training of GRU Model (Rupee to Japaneese Yen)

8. Analysis Of The Result

Table 1: Details of the Forex Datasets

Forex Data Sets	Training values Range	Testing Values Range
EUR TO USD	2011-01-01 to 2022-01-13	2022-01-14 to 2023-05-17
GBP TO USD	2011-01-01 to 2022-01-13	2022-01-14 to 2023-05-17
INR TO JPY	2011-01-01 to 2022-01-13	2022-01-14 to 2023-05-17

Table 2: Performance of LSTM Model (Testing)

Forex Data Pairs	MSE (Test)	RMSE (Test)	MAE (Test)
Euro to US Dollar	0.01874429	0.13690979	0.01572706
Britain Pound to US Dollar	0.01286213	0.11341136	0.01017417
Rupee to Japanese Yen	0.01997007	0.14131551	0.01637114

The Details of the Different currency Datasets splitting for the purpose of the training and testing of the Model are shown in the TABLE I respectively.
The performance of the LSTM and GRU models listed in TABLE II and TABLE III using performance metrices MSE, RMSE and MAE of three different currency datasets named Euro to US Dollar, Britain Pound to US Dollar, Rupee to Japanese Yen are shown respectively.
Fig.3 explains the actual and predicted graph of the LSTM and GRU models during Testing phase for the dataset Euro to US Dollar. Fig. 6 and Fig. 9 describes the MSE loss of the LSTM and GRU models with No of Epochs during training phase respectively.
Fig.4 explains the actual and predicted graph of the LSTM and GRU models during Testing phase for the dataset Britain Pound to US Dollar. Fig. 7 and Fig. 10 describes the MSE loss of the LSTM and GRU models with No of Epochs during training phase respectively.
Fig.5 explains the actual and predicted graph of the LSTM and GRU models during Testing phase for the dataset Rupee to Japanese Yen. Fig. 8 and Fig. 11 describes the MSE loss of the LSTM and GRU models with No of Epochs during training phase respectively.
GRU models offer several advantages over LSTM models. First, GRU models have a simpler architecture with fewer parameters, making them computationally more efficient. This efficiency is particularly beneficial when working with large datasets or limited computational resources.
Second, GRU models have been observed to require less training time and data to converge, which can accelerate the model development process and reduce the need for extensive hyperparameter tuning.
It is crucial to remember that based on the particular dataset and issue at hand, the performance comparison between LSTM and GRU models may change. To make an informed decision, it is advised to experiment with both models and evaluate how well they perform on the relevant dataset from the forex market.

Table 3: Performance of GRU Model (Testing)

Forex Data Pairs	MSE (Test)	RMSE (Test)	MAE (Test)
Euro to US Dollar	0.0102316	0.1011515	0.00782408
Britain Pound to US Dollar	0.0115609	0.1006688	0.01074564
Rupee to Japaneese Yen	0.0156178	0.1249713	0.01187943

9. Conclusion

In conclusion, Forex market analysis using LSTM and GRU models has proven to be effective in capturing temporal dependencies and making predictions. These models have the ability to process sequential data and learn complex patterns from historical price and volume data.Ultimately, the choice between LSTM and GRU models depends on the specific dataset, computational resources, and desired trade-off between accuracy and efficiency. Thorough experimentation and evaluation are crucial to determine the optimal model for Forex market analysis.

In summary, the integration of advanced deep learning models in Forex forecasting can provide significant advantages to various stakeholders. By leveraging the strengths of these models, financial analysts, traders, investment firms, and policymakers can make more informed decisions, ultimately leading to improved financial outcomes and economic stability.

References

[1] 1. Violeta, Gaucan. ”Introduction to the foreign exchange market.” (2010): 1-14.

[2] 2. Neely, Christopher J. ”Technical analysis in the foreign exchange market: a layman’s guide.” Federal Reserve Bank of St. Louis Review Sep (1997): 23-38.

[3] 3. Yıldırım, Deniz Can, Ismail Hakkı Toroslu, and Ugo Fiore. ”Forecasting directional movement of Forex data using LSTM with technical and macroeconomic indicators.” Financial Innovation 7 (2021): 1-36.

[4] 4. Babu, A. S., and S. K. Reddy. “Exchange rate forecasting using ARIMA.” Neural Network and Fuzzy Neuron, Journal of Stock Forex Trading, vol. 4, no. 3, 2015, pp. 01-05.

[5] 5. Bollerslev, Tim. “A conditionally heteroskedastic time series model for speculative prices and rates of return.” The Review of Economics and Statistics, 1987, pp. 542-547. JSTOR.

[6] 6. Chantarakasemchit, Orawan, Siranee Nuchitprasitchai, and Yuenyong Nilsiam. “Forex rates prediction on EUR/USD with simple moving average technique and financial factors.” 2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2020, pp. 771-774. IEEE.

[7] 7. Yu, Lean, Kin Keung Lai, and Shouyang Wang. “Multistage RBF neural network ensemble learning for exchange rates forecasting.” Neurocomputing, vol. 71, no. 16-18, 2008, pp. 3295-3302. Elsevier.

[8] 8. Behera, Sudersan, Sarat Chandra Nayak, and AVS Pavan Kumar. “A comprehensive survey on higher order neural networks and evolutionary optimization learning algorithms in financial time series forecasting.” Archives of Computational Methods in Engineering, vol. 30, no. 7, 2023, pp. 4401-4448.

[9] 9. Mohapatra, Puspanjali, Munnangi Anirudh, and Tapas Kumar Patra. ”Forex forecasting: A comparative study of llwnn and neurofuzzy hybrid model.” International Journal of Computer Applications 66.18 (2013): 46-53.

[10] 10. Sako, Kady, Berthine Nyunga Mpinda, and Paulo Canas Rodrigues. ”Neural networks for financial time series forecasting.” Entropy 24.5 (2022): 657.

[11] 11. Alizadeh, Milad, et al. ”An empirical study of binary neural networks’ optimisation.” International conference on learning representations. 2018.