Forex Price Prediction Using Deep Learning and Comparative Analysis with Traditional Time Series Models

MSc Research Project
Data Analytics

Hamza Pasking Akhtar
Student ID: x23115033

School of Computing
National College of Ireland

Supervisor: Abid Yaqoob

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Hamza Pasking Akhtar
Student ID:	x23115033
Programme:	Data Analytics
Year:	2024
Module:	MSc Research Project
Supervisor:	Abid Yaqoob
Submission Due Date:	12th December 2024
Project Title:	Forex Price Prediction Using Deep Learning and Comparative Analysis with Traditional Time Series Model
Word Count:	6440
Page Count:	21

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Hamza Pasking Akthar
Date:	12th December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Forex Price Prediction Using Deep Learning and Comparative Analysis with Traditional Time Series Models

Hamza Pasking Akhtar
x23115033

Abstract

This research focuses on predicting Forex price trends by employing both traditional time series models and advanced deep learning techniques. The study utilizes traditional models such as ARIMA and SARIMA, alongside deep learning methodologies like Long Short-Term Memory (LSTM) which is implemented using TensorFlow and Keras frameworks. The primary objective is to explore and conduct a comparative analysis based on the performance of these models in capturing the complexities of a highly volatile Forex market and forecasting the price. The capabilities of LSTM have also been explored by adding several variations including hyperparameter tuning, integrate technical indicators (e.g., RSI, MACD, and Moving Average), and Bi-directional LSTM. The performance of these models is analysed using evaluation metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). The research reveals that among the models utilized, a fine-tuned LSTM has been able to outperform other models which showcase its ability in capturing intricate patterns. The execution time for the model shows that increase in model architecture increases execution time of that model. The insights from this research aims to contribute to the field of financial forecasting by offering different perspective on the strengths and limitations of different modelling approaches for Forex price prediction.

Keywords: ARIMA, SARIMA, Deep Learning, Long Short-Term Memory (LSTM), Bi-directional LSTM, Forex, Price Prediction, Technical indicators

1 Introduction

The foreign exchange (Forex) market is a global decentralized financial marketplace from which currencies of different countries can be traded. The estimated daily transaction volume in Forex market reaches approximately $6.6 trillion. This market has a huge effect on modern global economies in terms of interest rate, growth of an economy and its financial stability (Krušković & Maričić, 2015).The market is highly liquid and volatile. The forex price is a complex and dynamic market and to forecast the movement of a currency price is a challenging task. Forex rates are complex financial time series that are highly nonlinear and nonstationary, which makes their modelling difficult (Kayacan, Ulutas, & Kaynak, 2010). Forex market forecasting is not just important for investors and traders, but it also acts as an economic trend indicator. To achieve this, a common method used by investors called technical analysis. This type of analysis is about analysing historical price since they may provide clues about the future movement of the market. Something that is plotted on charts alongside forex prices has also been introduced called indicators. Likewise, with the passage of time, different models have been developed to predict and predict these trends. In order to forecast time series data, traditional time series models, such as Auto Regressive Integrated Moving Average (ARIMA) and Seasonal ARIMA (SARIMA) have been used. However, these models have a limitation, the nonlinear patterns in the Forex market are not captured and hence have a sub optimal predictive performance which leads to finding models that perform better.

Recently, different research has been done and forecasting models using deep learning and their applications in different domains such as finance have received a lot of interest (Fischer & Krauss, 2017); (Huck, 2019). However, Long Short-Term Memory (LSTM) is becoming popular in forecasting financial markets. It is a type of recurrent neural network with the sole purpose of overcoming a problem that is faced by traditional RNNs like the vanishing gradient problem and be able to effectively model temporal dependencies (Hochreiter & Schmidhuber, 1997).

Technical indicators are simply mathematical pattern that is taken from historical data and applied to a time series, typically a price with the intention of forecasting price. They can be considered as an assistant used to identify future price trend. Integration of multiple commonly used technical indicators such as Relative Strength Index (RSI), Moving Average (MA), Moving Average Convergence Divergence (MACD) into LSTM models have been explored in this research to analyse and enhance the predictive accuracy of the model. This research aims to address the different challenges faced by different models and developing a comparative analysis framework between traditional time series models (ARIMA and SARIMA) and deep learning models specifically LSTM. The study further enhances LSTM models by incorporating technical indicators, performing hyperparameter optimization to improve predictive performance and bi-directional LSTM.

Hyperparameters such as learning rate, epochs and batch size are included to the model. After conducting hyperparameter tuning the data is given to the model again in order to predict the closing price of the forex pair. The execution time of each models has been recorded to understand how a model architecture affects execution time. As an evaluation metric mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE) is utilized.

In this study the major pair Euro/US Dollar (EUR/USD) pair is selected for forecasting as it is the largest traded currency pain in the world and constitutes to more than 80% of the total forex volume. This research conducts a systematic evaluation of the strengths and limitations of each model and approach, which could lead to actionable insights into the practical applicability of these models in Forex forecasting. The findings are intended to be used as a contribution to the development of more robust and accurate predictive models for the Forex market, so that traders and financial institutions can make informed decisions in the volatile Forex market.

1.1 Research Problem/Question

Q1. How to build advanced LSTM and Bi-directional LSTM in contrast to traditional time series models (ARIMA and SARIMA) in terms of improved accuracy and reliability for Forex price prediction?

Q2. Which combination of technical indicators and hyperparameter settings results the optimal forecasting performance across different evaluation metrics (MSE, RMSE, MAE) using real-world forex market data?

1.2 Research Contributions

1. **Identifying Forex Market Trends:** *The primary purpose is to explore how traditional time series such as ARIMA and SARIMA compared with deep learning models, specifically Long Short-Term Memory (LSTM) can be utilized to identify forex market trends and extract valuable patterns from historical exchange rate data.*

2. **Optimization of LSTM:** *Investigating how hyperparameter tuning (e.g., learning rate, batch size, number of layers, units, and dropout rate* (Munoz, Park, Stewart, Martin, & Hedengren, 2023)*) enhances the performance of LSTM models in predicting forex prices. Understanding how this contributes to achieving more accurate and efficient forecasting. Similarly, the performance of bi-directional LSTM on price prediction is also explored.*

3. **Integrating Technical Indicators**: *Evaluating the role of widely used technical indicators such as Relative Strength Index (RSI), Moving Average (MA), and Moving Average Convergence Divergence (MACD) in improving the predictive accuracy of LSTM models for forex price movements.*

2 Related Works

The foreign exchange market is the largest and most liquid of all the financial markets. The forex market was limited to commercial, central and hedge funds over the years until recent technological growth, which has increased the forex market to retail currency traders (Talebi, Hoang, & Gavrilova, 2014). In previous years, many methods have been developed and used in order to forecast the FOREX market. Past patterns in data can be used to forecast future price points in the market was made for these methods. To date, there are different forecasting models being utilized and research being done to forecast the FOREX market, some of the mostly utilized strategies for modelling volatility in time series are Deep Learning (Brownlee, 2018), ARIMA model (Tseng, Tzeng, Yu, & Yuan, 2001) which have strong points and weak points.

2.1 Traditional Time Series in Forex Price Prediction

ARIMA and SARIMA are traditional time series models that are extensively used for financial forecasting. Among the most popular approach for time series forecasting are ARIMA models, as introduced by (Box & Jenkins, 1976)ARIMA models are effective for linear relationships in time series data, something which is very common in Forex. ARIMA is extended to SARIMA by the addition of seasonality component, which it makes appropriate for time series data with regular patterns of seasonality. Nevertheless, these traditional time series models are not able to effectively capture nonlinear relationships (Tsay, 2010). That's why we would carry out comparative analysis to find out which model works better and which regard it works well.

Even though ARIMA is well robust for the linear modelling, yet the restrictions of capturing non-linear relationships require more advanced modelling techniques such as deep learning. The effectiveness of the ARIMA model also relies on the parameters that they use (Montgomery, Jennings, & Kulahci, 2015) .Nevertheless, the study of (Babu, 2015) indicates that even the simplicity of ARIMA can produce reasonable forecast for short term forex predictions and is more effective than other complex nonlinear methods like neural network and fuzzy neurons.

As Forex markets continue to evolve, the role of ARIMA and SARIMA as benchmarks in comparative studies with advanced models, such as deep learning techniques, remains crucial.

2.2 Deep Learning for Forex Market Analysis

Deep learning is a field of machine learning have become quite powerful in handling complex and nonlinear data patterns. It is inspired by the structure and the way how the human brain functions. Artificial neural network with deep layers is used to extract hierarchal representations of data by deep learning models. That amounts to learning complex relationships and making accurate predictions with noisy and high dimensional datasets.

Deep learning architectures are used and Long Short-Term Memory (Wang, Dong, & Zhang, 2023) is mostly used. Unlike standard recurrent neural network (RNN) LSTM is a type of RNN model with the purpose of overcoming the vanishing gradient problem, which prevents standard RNNs from learning about long term dependencies (Hochreiter & Schmidhuber, 1997). Instead, LSTMs use a unique cell structure in which input, output, and forget gates are used. The memory cell begins with input gate, output gate. A new feature, forget gate (Felix, Schmidhuber, & Cummins, 2000) was added to this standard LSTM. They serve as these gates to control information flow around in the network and selectively remember or forget the past, which makes them well suited to Forex price prediction tasks, where both previous and historical trends can impact future movements.

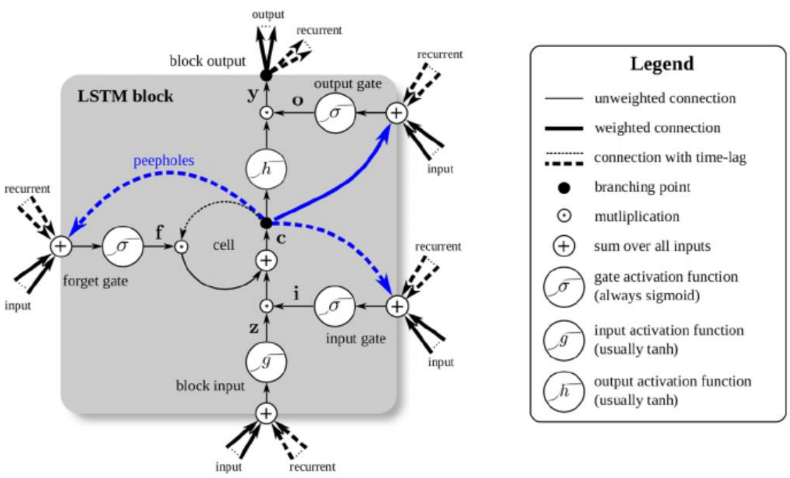

Figure 1 Vanilla LSTM (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2015)

Furthermore, features such as peephole connections and full back-propagation through time (BPTT) training are added to the LSTM architecture (Felix, Schmidhuber, & Cummins, 2000); (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2015). Addition of these modifications led to the renaming of the architecture as Vanilla LSTM (Greff, Srivastava, Koutník, Steunebrink, & Schmidhuber, 2015).

LSTMs naturally have good properties for processing sequential data, Forex prices for example, but are highly dependent on the choice of hyperparameters. The settings that govern the learning process of the model, not learnt from the data itself, are called hyperparameters. Some of these include the number of layers, hidden units, learning rate, batch size, etc. Although the study by (Chung & Shin, 2018) focuses on stock, the importance of careful hyperparameter selection for achieving optimal performance is emphasized. As shown by their work, the GA optimized LSTM network outperformed the benchmark model on all error metrics. Specifically, MAE is reduced by 12.80%, and MAPE is improved by 0.19%, while MSE is reduced by 13.11%. All these measures successfully confirmed the effectiveness of utilizing GA-LSTM in order to enhance the forecasting accuracy.

Unlike standard LSTMs, which process sequential data forward, bi-directional LSTMs take a fuller approach. They incorporate information from both past and future by utilizing two LSTMs: two that process the sequence in forward direction and two that process it in reverse. Then these two LSTMs outputs are combined to give more context aware prediction. LSTM layers consisted of a bi-directional LSTM, containing two separate LSTM layers such that the first compresses the input sequence from start to finish, and the second compresses it in reverse. These layers generally feed into the outputs, which are joined or averaged to give the final output.

Figure 2 Bidirectional LSTM structure diagram

According to (Jia, Huang, Pang, & Zhao, 2019) who research bi-directional LSTM against a stock price, using a bi-directional LSTM yielded substantial improvements in prediction accuracy. Compared to a unidirectional LSTM, RMSE and MAE were reduced by 24.2% and 19.4%, respectively, with overall accuracy increasing by 0.13%. Further optimization with a dropout rate of 0.2 resulted in even lower error values (RMSE: 1.4082, MAE: 0.9398), highlighting the effectiveness of bi-directional LSTMs for enhanced forecasting. Further optimization with a dropout rate of 0.2 resulted in even lower error values (RMSE: 1.4082, MAE: 0.9398), highlighting the effectiveness of bi-directional LSTMs for enhanced forecasting.

While raw price data is the basis for Forex forecasting, yet technical indicators can greatly improve the predictive power of LSTM models. Mathematical calculations that use historical price or volume to identify patterns, trends, measure volatility and other potential reversals in the market are called technical indicators. Inputting these indicators as input features to LSTMs will enhance their understanding of how the market works and therefore increase their forecasting accuracy. In addition to that, they can also give a better visual representation of the dynamics of the market. Research done by (Yıldırım, Toroslu, & Fiore, 2021) found that their results indicate that the technical and macroeconomic indicators combined with the LSTM model significantly outperforms a benchmark random walk model for Forex forecasting.

3 Proposed Research Methodology

This research utilizes different Python libraries for different purposes such as data processing, visualization and machine learning tasks. Python libraries like *NumPy* for numerical computations, *Panda* for handling data and matplotlib and seaborn for visualization. The date is converted to date time.

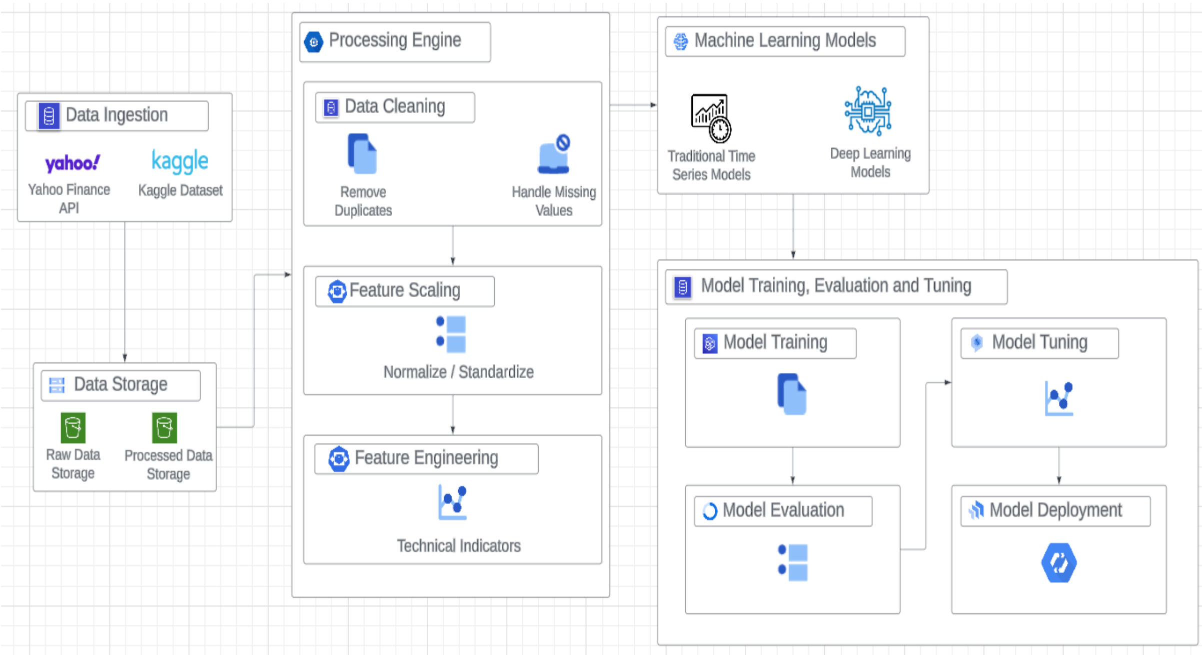

Figure 3 Proposed architecture diagram

3.1 Dataset

3.1.1 Dataset Selection

This research utilized forex price data from Yahoo Finance and focusing specifically on EURUSD as a forex pair. The covered date range for the dataset from Yahoo Finance [1] is from 1st January 2004 to 31 December 2023 and it has 5188 data entries. A test train split is done from which 80% is allocated as training and 20% for testing. These would prove useful during evaluation of the models to see how it would forecast on unfamiliar data.

[1] Yahoo Finance, "Historical Data for EURUSD," accessed December 10, 2024, https://finance.yahoo.com/quote/EURUSD=X/

Table 1. Description of columns in dataset

Column	Description
Date	The date at which a price has been closed.
Open	The forex pair opening price for that timeframe
High	The forex pair highest price for that timeframe
Low	The forex pair lowest price for that timeframe
Close	The forex pair closing price for that timeframe
Adjusted Close	Closing price after adjustments for all applicable splits and dividend distributions
Volume	Number of lots traded for that timeframe.

In this study, we preprocess the data before doing any analysis and modelling using ARIMA, SARIMA, and LSTM. Since the dataset contained no missing values, we directly proceeded to ensure its compatibility with the respective models. For ARIMA and SARIMA, we addressed the stationarity requirement by performing a stationarity test called *Augmented Dickey-Fuller (ADF) test*. Non-stationary data was transformed into a stationary series through first-order differencing, and seasonal differencing was applied where necessary for SARIMA. Seasonal patterns in the data were analysed using seasonal decomposition to identify trends, seasonality, and residual components, ensuring compliance with SARIMA assumptions.

We scaled the data using *MinMaxScaler* from the *sklearn* library for LSTM to normalize values between 0 and 1. This step is to ensure the stability and efficiency of training the neural network by equalizing the contribution of all input features. In an 80-20 split, the scaled dataset was split into training and testing sets. We prepared the data for LSTM by creating sequences of input features and their corresponding labels using a sliding window approach with 30 days of past data to predict the next day's closing price. The data was then reshaped to a three-dimensional array to fit the input requirements of LSTM models with dimensions of the number of samples, time steps and features.

Through these preprocessing steps, we ensure that the dataset is appropriately structured and scaled for each modelling approach, creating a robust foundation for accurate predictions and meaningful insights.

3.2 Model Building

3.2.1 ARIMA

The Auto Regressive Integrated Moving Average (ARIMA) model is a time series forecasting model. It incorporates three components: Auto Regressive (AR), Integrated (I), and Moving Average (MA). The AR component captures dependencies amongst an observation and a number of lagged observations, the I component handles nonstationary through differencing, and the MA component models the relationship between an observation and the residual errors of a moving average model. As was formalized by (Box & Jenkins, 1976) in their seminal work. The need for rigorous pre modeling steps, such as stationarity and white noise residuals, was emphasized. Since the ARIMA model requires stationarity in the time series, the seasonality that we are observing is often removed through differencing. The degree of differencing required to make the series stationary, is represented by the "Integrated" component, d. After stationarity is reached, the model parameters, p (AR order) and q (MA order) are estimated by the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots. The stationary series is fitted to the final ARIMA model and the forecast is generated.

3.2.2 SARIMA

Since ARIMA doesn't take seasonality into account directly, so it is extended to SARIMA. SARIMA model is designed for time series data with seasonal patterns. ARIMA is good at capturing non seasonal trends, but SARIMA adds additional seasonal parameters to model recurring patterns. The model is characterized by a set of parameters: The form (p,d,q) × (P,D,Q,m) denotes the couple (non seasonal components: autoregressive, differencing, and moving average orders) (P,D,Q) and their maximum length, m (periodicity of the seasonal cycle). Both non seasonal (d) and seasonal (D) differencing ensures stationarity. Characterizations of underlying periodic patterns often manifest in this seasonal decomposition and visualization. Periodic patterns are common in many domains including energy, sales and finance, and SARIMA is widely used in them (Hyndman & Athanasopoulos, 2018).

3.2.3 LSTM

Recurrent neural network (RNN) is a type of neural network that can learn sequential data, however, standard RNNs suffer from a problem known as the vanishing gradient problem and are not able to capture long term dependencies (Hochreiter & Schmidhuber, 1997). LSTMs works well at processing sequential data by retaining information over extended time periods through a gated mechanism. This ability makes them particularly suitable for time series forecasting, where historical trends influence future predictions.

What makes LSTMs so important is their memory cells, which can store information over long time. Compared with vanilla RNNs and update the whole hidden state at every timestep, LSTMs only update certain part of hidden state through their gates. The architecture of LSTMs includes three main gates: inputs to the forget gate, the input gate and the output gate. The flow of information both into and out of, and within, the memory cell is controlled by these gates so that the model is able to selectively remember or forget data.

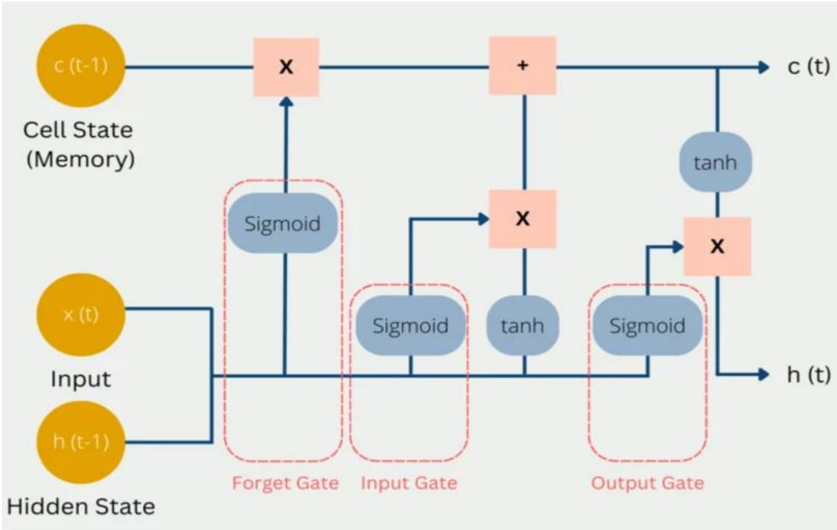

Figure 4 Employed LSTM Architecture

One of the biggest use cases of LSTMs is time series forecasting because they can learn complex temporal dynamics. In financial time series such as forex price predictions, the nonlinear relationships between features such as historical prices, volume and market indicators render traditional statistical models inadequate. Yet, in contrast, LSTMs learn these dependencies through no feature engineering (Fischer, 2018).

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

$$i_t = \sigma_g\left(W_i x_t + U_i h_{\{t-1\}} + b_i\right)$$

$$o_t = \sigma_g\left(W_o x_t + U_o h_{\{t-1\}} + b_o\right)$$

$$c_t = f_t \odot c_{\{t-1\}} + i_t \odot \sigma_c\left(W_c x_t + U_c h_{\{t-1\}} + b_c\right)$$

$$h_t = o_t \odot \sigma_h(c_t)$$

These equations represent the functioning of an LSTM (Long Short-Term Memory) network, designed to handle sequential data by managing long-term dependencies. The forget gate (f_t) decides what information from the previous cell state to retain or discard. The input gate (i_t) determines what new information to add to the cell state, while the cell state (c_t) is updated by combining retained and new information. The output gate (o_t) controls what part of the updated cell state is passed as the hidden state (h_t), which serves as the output for the current time step. Together, these gates enable LSTMs to effectively learn and remember patterns in sequential data.

3.2.4 Combinational Hyperparameters in LSTM

A Recurrent Neural Network (RNN) specially called Long short-term memory (LSTM) networks are able to capture dependencies and relationships in sequential data over an extended time. We discuss the architecture of their networks, which solves issues like the vanishing gradient problem and allows them to learn both short- and long-term patterns. As with most machine learning algorithms, optimal performance or efficiency can be limited by a number of hyperparameters, and this needs to be tuned carefully based on the dataset and task.

1. **Number of LSTM Units (Neurons):**

   This means that the more units we put in one LSTM layer, the better the model will be trained at learning complex patterns that go beyond linear models. It can capture the intricate data relationship, but have that lead to overfitting, especially if the data is limited. The critical thing is to balance how complex the model is with the amount of data that's available.

2. **Layer Count:**

   Multiple LSTM Layer Stacking allows to create deeper networks to capture hierarchical features present in the data. However, the drawback is such deeper architectures might require larger amounts of training data and computational time. Because the gradients can vanish or explode in overly deep models, there are challenges such as those, which can be overcome by techniques like gradient clipping and specialized initialization methods.

3. **Dropout Regularization:**

   Dropout is a technique that randomly deactivates a subset of neurons during training. This prevents the model from becoming overly dependent on specific neurons, thereby reducing overfitting. It promotes robustness by encouraging the model to generalize better to unseen data.

4. **Batch Size:**

   Batch size refers to the number of samples processed in a single iteration. Larger batch sizes stabilize the training process but increase memory consumption. Smaller batches can make training faster and introduce variability, which might help the model escape local minima but may also result in noisier convergence.

5. **Learning Rate:**

   The learning rate controls the step size taken during optimization. A higher learning rate accelerates convergence but risks overshooting the optimal point, while a lower rate ensures more stable convergence but increases training time. Adaptive learning rate optimizers are often employed to balance these trade-offs.

6. **Number of Epochs:**

   An epoch represents one complete pass through the training dataset. Too few epochs may lead to underfitting as the model doesn't learn sufficiently from the data, while too many epochs risk overfitting, causing the model to memorize the training data rather than generalize from it. Early stopping is a common strategy to prevent overtraining.

3.2.5 Technical Indicators

In this research the role of technical indicators has also been explored in order to evaluate their influence to the model

a. Moving Average (MA):

As a technical analysis component, we incorporate Moving Averages (MA) in this research. Moving Average which filters out noise by removing daily price fluctuations and identifying trends. For this study we use the 50 (short term) and the 200 (long term) day Moving Average. The choice of a 50-day MA and a 200-day MA is a choice that is often made because they are widely regarded as key trend analysis benchmarks in financial markets. Thus, the 50-day MA provides a balance between short- and long-term volatility, and is an intermediate term price movement. On the flip side, the 200-day MA shows us long term trends and gives us a wider view of market direction. This combination of the two moving averages does a great job of identifying a significant change of trend, as it incorporates both short and long term dynamics. The way the strategy is used is when the 50-day MA crosses above the 200-day MA it's a bullish trend, and when the 50-day MA crosses below the 200-day MA it's a bearish trend.

b. Moving Average Convergence Divergence (MACD):

The MACD is a trend following momentum indicator used in the analysis of stock and forex. MACD is versatile and widely used in making technical analysis. A sell or buy is indicated when the MACD line crosses over the signal line and the MACD line is below the signal line.

c. Relative Strength Index (RSI):

Relative Strength Index (RSI) is a momentum oscillator which measures the magnitude of recent price changes. It works on the logic of overbought and oversold. RSI is a range of numbers from 0 to 100 developed by J. Welles Wilder Jr. RSI readings above 70 traditionally indicate an overbought condition, implying that the asset in question is ready for a price correction. On the other side, readings below 30 is taken as an oversold situation and could signifies an undervaluation with the potential to come up in price.

4 Implementation

4.1 Implementation Settings

The code is run on a machine with 16GB of RAM with an i7 8^{th} Gen Processor. Jupyter Notebook is used which is present under Anaconda Navigator and running it utilizes 2GB of RAM.

4.2 LSTM Model Configuration

For the base LSTM model the configuration is as follows.

1. First LSTM Layer: 50 units with tanh activation, returning sequences to capture temporal dependencies from the last 60 days' closing prices.

2. Second LSTM Layer: 50 units with tanh activation, not returning sequences to prepare for the Dense layer.

3. Dense Layer: A fully connected layer with 1 unit to output the predicted EUR/USD closing price.

4. Data Scaling: Closing prices are scaled between 0 and 1 using MinMaxScaler for improved training performance.

5. Training: The model is trained for 20 epochs combined with the Adam optimizer with a batch size of 32, employing MSE as the loss function.

For the optimized LSTM that incorporates hyperparameter tuning to improve model performance.

1. LSTM Layers: Three LSTM layers with 100, 100, and 50 units, respectively, all using tanh activation. The first two LSTM layers return sequences, while the final layer outputs a single sequence.

2. Dropout Regularization: A 30% dropout rate is applied after each LSTM layer to mitigate overfitting.

3. Dense Layer: A fully connected Dense layer with 1 unit generates the final prediction.

4. Training Configuration: The model is trained for up to 50 epochs with a batch size of 32, using the Adam optimizer and mean_squared_error loss function. Early stopping is employed to terminate training if validation loss does not improve for 10 consecutive epochs. Model checkpoints save the best-performing model based on validation loss.

5. Sequence Length and Scaling: The model uses the previous 60 days of closing prices to predict the next day’s price. Data is scaled to the range [0, 1] using MinMaxScaler for faster convergence.

4.3 Model Evaluation Metrics

When conducting an evaluation on deep learning projects such as forex, using deep learning like LSTM then the metrics used for evaluation of the model are Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE).

4.3.1 Mean Squared Error (MSE): The Mean Squared Error is the average of the squared differences between a predicted and an actual value. It's sensitive to outliers, this penalizes large errors more than small ones. The better predictive accuracy is denoted as contribution of smaller MSE of that model. Especially for comparison of models, it draws attention to bigger prediction errors.

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

Note: n is the number of observations, y_i is the actual value, $y^{\wedge}_i$ is the predicted value.

4.3.2 Root Mean Squared Error (RMSE): MSE is the square root of Root Mean Squared Error. That's why it's easy to read, since it gives error information in the same units as the data, so you can really understand where you're off in the context of what you're trying to solve. MSE is less intuitive as compared to RMSE because it expresses gains and losses, and not in terms of the target variable's units. Smaller numbers leave a better indication of performance like MSE.

$$RMSE = \sqrt{\frac{1}{n} \sum_{\{i=1\}}^{\{n\}} (y_i - \hat{\ } y_i)^2}$$

4.3.3 Mean Absolute Error: Mean Absolute Error gives us the Mean of absolute difference between the actual and predicted values. It isn't squared errors as with MSE and RMSE, so it's much less affected by outliers. The average error magnitude is provided as a straightforward interpretation of prediction errors by MAE. The smaller the model accuracy, the higher the value.

$$MAE = \frac{1}{n} \sum_{\{i=1\}}^{\{n\}} |y_i - \hat{\ } \{y\}_i|$$

5 Evaluation and Result Analysis

In this research forecasting forex price using traditional time series models, ARIMA and SARIMA were compared to a deep learning model, LSTM. An optimized LSTM where hyperparameter tuning was conducted turned out to yield the best result.

5.1 ARIMA

With an MSE of 3.35e-05, the ARIMA model is of a strong predictive capability with low error. The model's predictions were on average less than one cent off from the actual values, with the Root Mean Squared Error (RMSE) being 0.0058. The Mean Absolute Error (MAE) was 0.0045. In contrast, during periods of high volatility, its predictions differ significantly from actual test data. The low RMSE and MSE suggest good numerical behavior, but visual mismatch suggests that modeling forex price behavior is not without its limits.

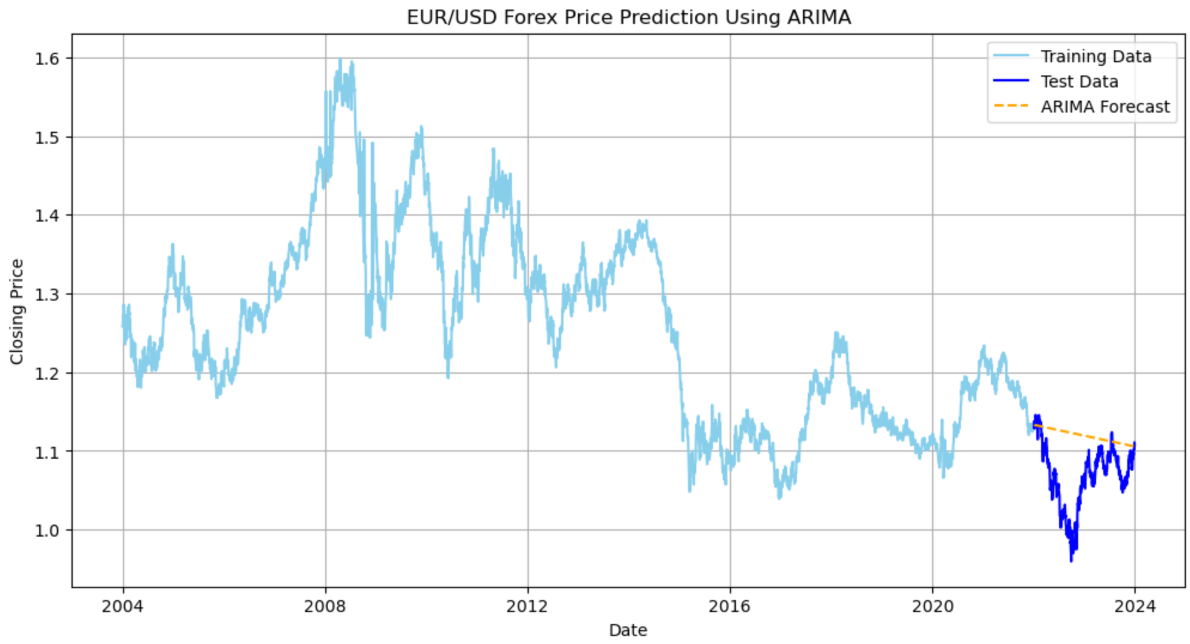

Figure 5 Forecast vs Actual Price using ARIMA

5.2 SARIMA

ARIMA and LSTM models resulted in a Mean Squared Error (MSE) of 0.0046, whereas the SARIMA model produced a higher prediction error. The RMSE was 0.0677, which was much less accurate than the other models. Interestingly, the Mean Absolute Error (MAE) of the ARIMA model was the same at 0.0045. SARIMA does not capture all seasonal elements and fails to accurately predict the sharp, recent movements in the test data. The fact that this discrepancy exists indicates that traditional statistical models are not appropriate for use in highly volatile markets.

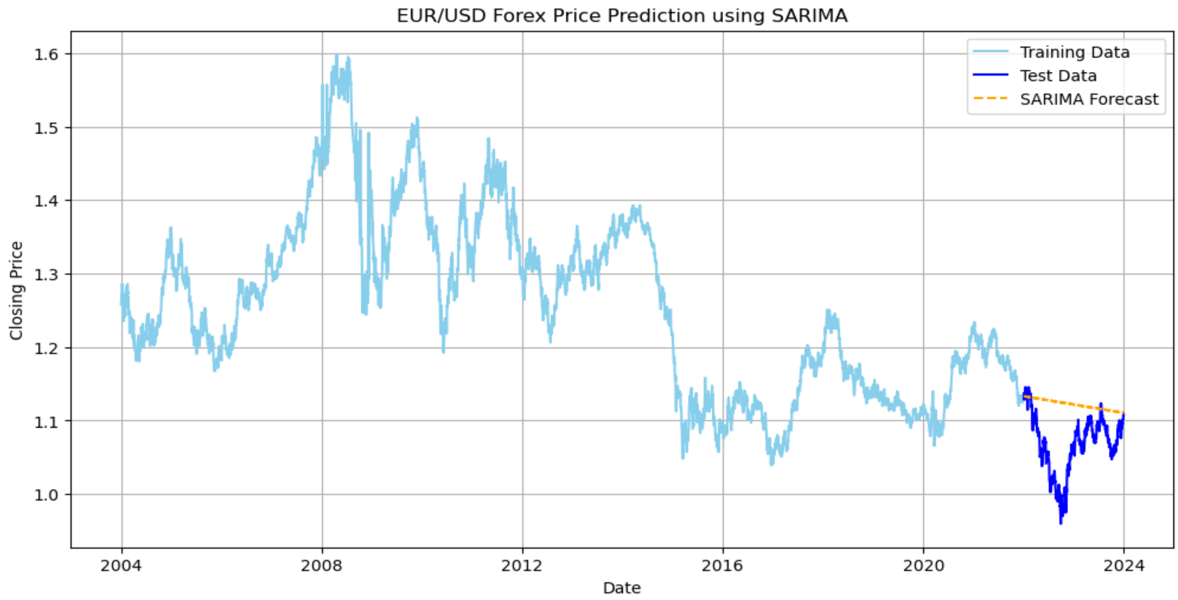

Figure 6 Forecast vs Actual Price using SARIMA

5.3 LSTM

Relatively low prediction error was obtained in the LSTM model with Mean Squared Error (MSE) of 7.97e-05. Finally, the Root Mean Squared Error (RMSE) turned out to be 0.0089, which means that the model's predictions, on average, are off by about 0.009. Furthermore, a Mean Absolute Error (MAE) of 0.0068 was also returned, meaning that the average prediction was in error by 0.0068 with the actual value. We find that the LSTM model is able to make reasonably accurate predictions, though it is not as good as more advanced models.

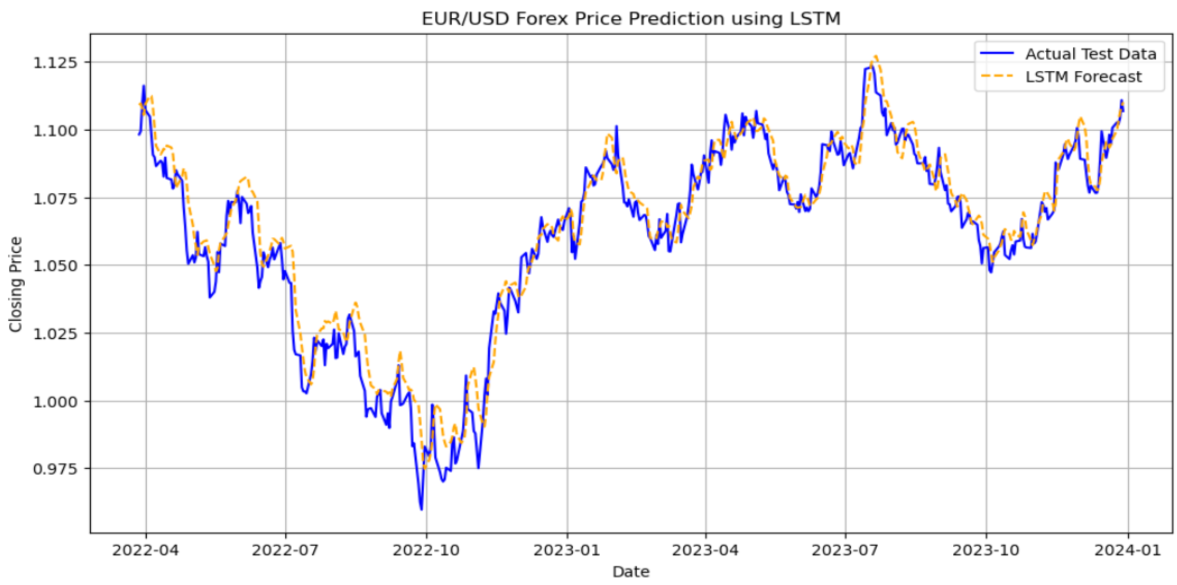

Figure 7 Actual vs Forecast Values Graph for LSTM

5.4 Optimized LSTM

The standard LSTM model had a Mean Squared Error (MSE) of 7.14e-05, while the Optimized LSTM model showed a slightly better accuracy of 7.14e-05 MSE. The Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) of LSTM model were 0.0084, 0.0064 respectively, which were slightly better than LSTM model. These improvements imply that the model's performance as a more accurate predictor was improved by hyperparameter tuning, for example, adjusting the number of units and layers.

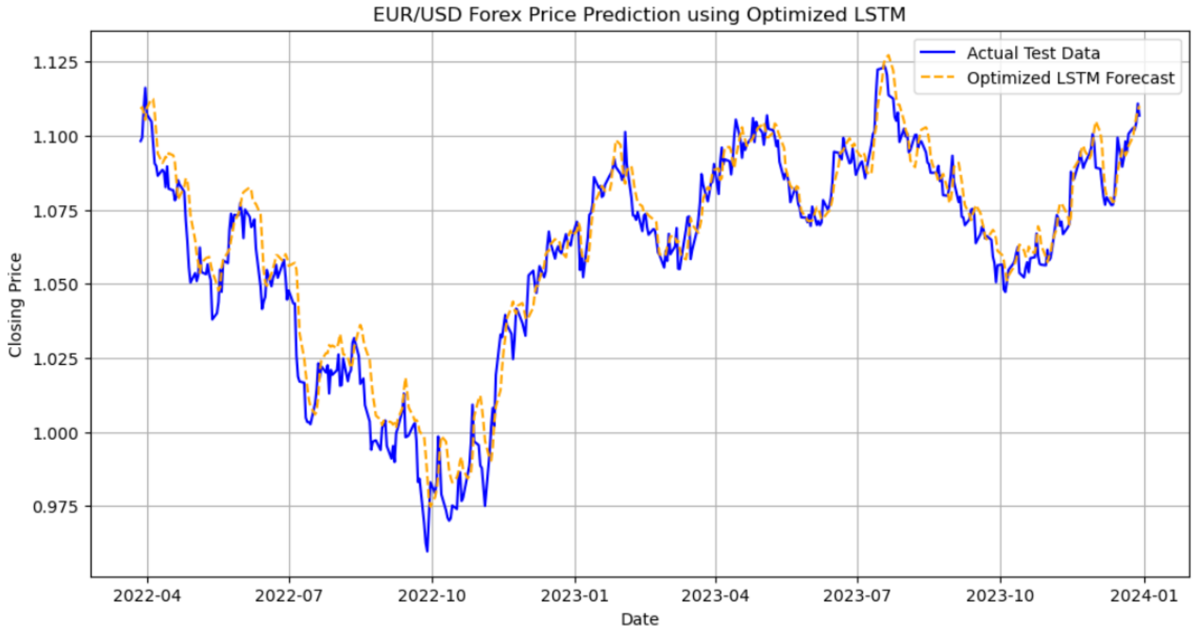

Figure 8 Actual vs Forecast Values Graph for Optimized LSTM

5.5 Bi-Directional LSTM

The standard LSTM model had a Mean Squared Error (MSE) of 7.14e-05, while the Optimized LSTM model showed a slightly better accuracy of 7.14e-05 MSE. The Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) of LSTM model were 0.0084, 0.0064 respectively, which were slightly better than LSTM model. This improvement indicates that the hyperparameter tuning, e.g., the number of units and the number of hidden layers, might also improve the performance of the same model and make it more accurate in predicting.

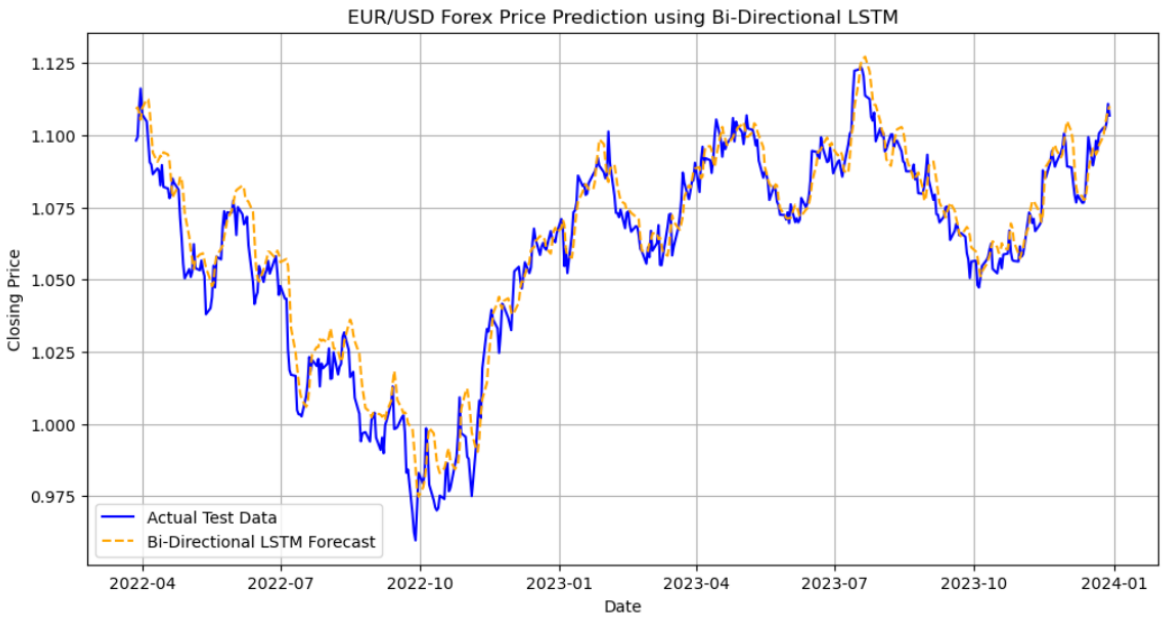

Figure 9 Actual vs Forecast Values Graph for Bi-Directional LSTM

5.6 LSTM with Technical Indicators

The LSTM with Technical Indicators model showed identical results to the standard LSTM model, with a Mean Squared Error (MSE) of 7.97e-05, Root Mean Squared Error (RMSE) of 0.0089 EUR/USD, and Mean Absolute Error (MAE) of 0.0068 EUR/USD. This suggests that incorporating technical indicators did not significantly improve the model's performance. It may indicate that, for the specific dataset and the time period considered, the raw price data alone might be sufficient for accurate predictions, and adding indicators did not provide substantial additional value.

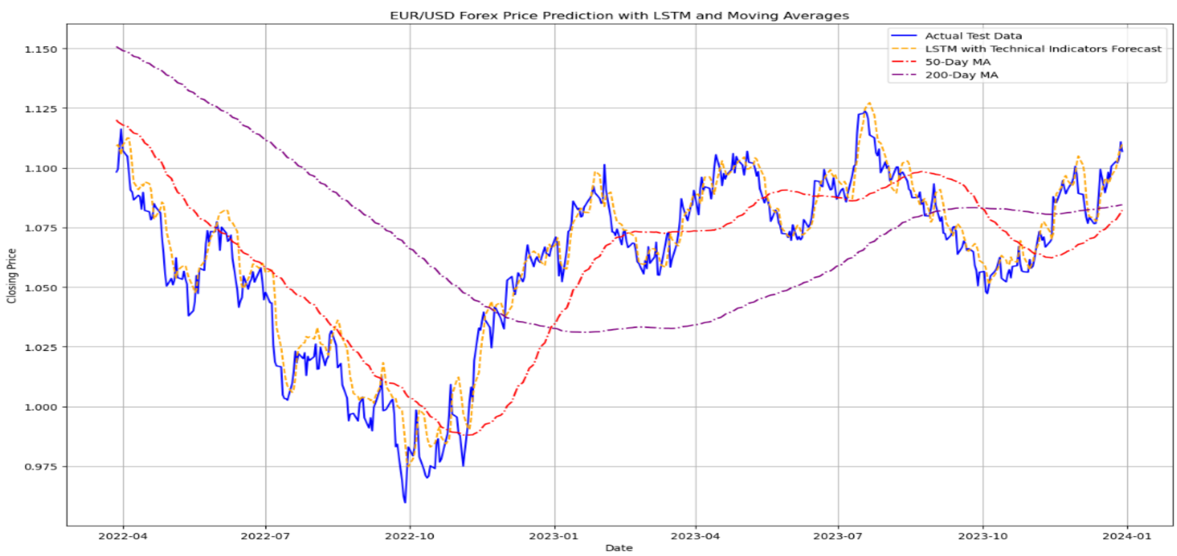

Figure 10 Actual vs Forecast with MA LSTM

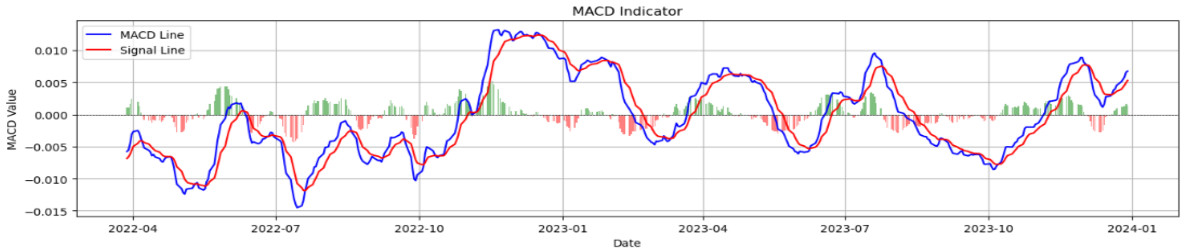

Figure 11 EURUSD MACD Indicator

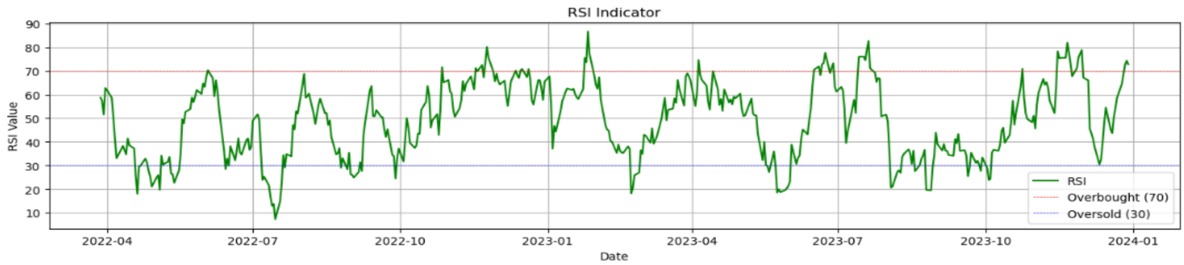

Figure 12 EURUSD RSI Indicator

5.7 Discussion

Table 2. Result Comparison Table

Model	**MSE (Mean Squared Error)**	**RMSE (Root Mean Squared Error)**	**MAE (Mean Absolute Error)**
ARIMA	3.35e-05	0.0058	0.0045
SARIMA	0.0046	0.0677	0.0045
LSTM	7.97e-05	0.0089	0.0068
Optimized LSTM	7.14e-05	0.0084	0.0064
Bi-Directional LSTM	0.00017	0.0130	0.0112
LSTM with Indicators	7.97e-05	0.0089	0.0068

After the evaluation of the different models used in the prediction of forex price there are a certain conclusion we can have based on the result. ARIMA and SARIMA in volatile periods, its predictions deviate significantly from the actual test data. ARIMA, with an MSE of 3.35e-05 and an RMSE of 0.0058, outperformed SARIMA, which had higher error metrics (MSE: 0.0046, RMSE: 0.0677). While the low RMSE and MSE indicate reasonable performance numerically, the visual mismatch highlights potential limitations in modeling forex price behavior. Among the different deep learning models that has been utilized, the Optimized LSTM model performed the best, with lower error metrics compared to the standard LSTM and Bi-Directional LSTM backed with visual representation. The Bi-Directional LSTM model exhibited the highest error rates, suggesting that the additional complexity did not improve predictive accuracy for this task. Interestingly, the addition of Technical Indicators in the model did not provide any notable advantage, as its performance metrics mirrored the standard LSTM. This implies that the extra input features could have been noisy or redundant, and thus did not offer the model extra capability of prediction. Overall, Optimized LSTM emerged as the best-performing models for Forex price prediction.

Table 3. Execution Time Comparison Table

Technique	Execution Time
ARIMA	1.7462947368621826 seconds
SARIMA	7.699831008911133 seconds
LSTM	83.64203333854675 seconds
Optimized LSTM	425.5994005203247 seconds
Bi-directional LSTM	321.83670926094055 seconds
Technical Indicators LTSM	209.5267128944397 seconds

The above result show that there is a tradeoff between model complexity and computational time. ARIMA and SARIMA were computationally efficient with 1.75s and 7.70 respectively. This is due to their simple statistical nature.

However, LSTM on the other hand required more time and with the increase in complexity of the model the time increased too. Triple layer architecture took more time compared to basic dual layer LSTM evident on the recorded execution time.

6 Conclusions and Future Works

This project focused on the comparison of traditional time series models like ARIMA and SARIMA against deep learning model like Long Short-Term Memory (LSTM) in the domain of Forex prediction, more specifically the EURUSD pair. The main objective was to shed light to the limitations of traditional time series models which is then overcome by LSTM. Other variations of LSTM such as optimizing it through hyperparameter tuning, Bi-directional LSTM, and LSTM combined with Technical Indicators were also explored for comparative analysis. Through the comparative analysis conducted and its results through the evaluation metrics used combined with visualization it could be concluded that an optimized LSTM performs ideally and gives the best result therefore the role of hyperparameter tuning is of great importance.

For future work, incorporating hybrid models where we could leverage the strength of two methodologies could be explored. Similarly, integrating macroeconomic indicators or external variables could possibly provide a better insight and improve the robustness of the predictions. Testing the models on other currency pairs such as XAU/USD, AUDUSD, NZDUSD or Crypto such as BTC/USD could validate their adaptability to other diverse datasets.

7 References

Babu, S. (2015, 01). Exchange Rate Forecasting using ARIMA, Neural Network and Fuzzy Neuron. *Journal of Stock & Forex Trading, 04}*. doi:10.4172/2168-9458.1000155

Box, G., & Jenkins, G. (1976). *Time Series Analysis: Forecasting and Control.* Holden-Day.

Brownlee, J. (2018). *Better Deep Learning: Train Faster, Reduce Overfitting, and Make Better Predictions.* Machine Learning Mastery. Retrieved from https://books.google.ie/books?id=T1-nDwAAQBAJ

Chung, H., & Shin, K.-s. (2018, 10). Genetic Algorithm-Optimized Long Short-Term Memory Network for Stock Market Prediction. *Sustainability, 10,* 3765. doi:10.3390/su10103765

Felix, G., Schmidhuber, J., & Cummins, F. (2000, 10). Learning to Forget: Continual Prediction with LSTM. *Neural computation*, 2451-71. doi:10.1162/089976600300015015

Fischer, T., & Krauss, C. (2017, 12). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research, 270*. doi:10.1016/j.ejor.2017.11.054

Greff, K., Srivastava, R., Koutník, J., Steunebrink, B., & Schmidhuber, J. (2015, 03). LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems, 28*. doi:10.1109/TNNLS.2016.2582924

Hochreiter, S., & Schmidhuber, J. (1997, 12). Long Short-term Memory. *Neural computation, 9,* 1735-1780. doi:10.1162/neco.1997.9.8.1735

Huck, N. (2019, 04). Large data sets and machine learning: Applications to statistical arbitrage. *European Journal of Operational Research, 278*. doi:10.1016/j.ejor.2019.04.013

Hyndman, R., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice.* Monash: OTexts.

Jia, M., Huang, J., Pang, L., & Zhao, Q. (2019, 01). Analysis and Research on Stock Price of LSTM and Bidirectional LSTM Neural Network. doi:10.2991/iccia-19.2019.72

Kayacan, E., Ulutas, B., & Kaynak, O. (2010). Grey system theory-based models in time series prediction. *Expert Systems with Applications, 37,* 1784-1789. doi:https://doi.org/10.1016/j.eswa.2009.07.064

Krušković, B., & Maričić, T. (2015). Empirical Analysis of the Impact of Foreign Exchange Reserves to Economic Growth in Emerging Economics. *Applied Economics and Finance*.

Montgomery, D., Jennings, C., & Kulahci, M. (2015). *Introduction to Time Series Analysis and Forecasting.* Hoboken: John Wiley & Sons.

Munoz, S., Park, J., Stewart, C., Martin, A., & Hedengren, J. (2023). Deep Transfer Learning for Approximate Model Predictive Control. *Processes, 11*. doi:10.3390/pr11010197

Talebi, H., Hoang, W., & Gavrilova, M. (2014, 06). Multi-Scale Foreign Exchange Rates Ensemble for Classification of Trends in Forex Market. *Procedia Computer Science, 29*. doi:10.1016/j.procs.2014.05.190

Tsay, R. S. (2010). *Analysis of Financial Time Series.* New Jersey: John Wiley & Sons, Inc.

Tseng, F., Tzeng, G.-H., Yu, H.-C., & Yuan, B. (2001, 02). Fuzzy ARIMA model for forecasting the foreign exchange market. *Fuzzy Sets and Systems*, 9-19. doi:10.1016/S0165-0114(98)00286-3

Wang, X., Dong, S., & Zhang, R. (2023). An Integrated Time Series Prediction Model Based on Empirical Mode Decomposition and Two Attention Mechanisms. *Information*. doi:10.3390/info14110610

Yıldırım, D., Toroslu, I., & Fiore, U. (2021, 12). Forecasting directional movement of Forex data using LSTM with technical and macroeconomic indicators. *Financial Innovation*. doi:10.1186/s40854-020-00220-2