# Stock Market Prediction using RNN-based Models with Random and Tuned Hyperparameters

Priyank Gupta
SOS Computer Science and Applications, Jiwaji University, Gwalior, MP, India

Sanjay Kumar Gupta
SOS Computer Science and Applications, Jiwaji University, Gwalior, MP, India

Rakesh Singh Jadon
Department of Computer Science and Engineering, MITS, Gwalior, MP, India

## ABSTRACT
The stock market in India has become more passionate in recent years. Because of the maneuverability of the stock market, it is also difficult to predict future trends and patterns in the stock market. Various Deep Learning (DL) methods, such as Recurrent Neural Networks (RNN), produce excellent results in stock market forecasting. In this paper, we integrate RNN-based models such as Long-Short-Term Memory (LSTM), Stacked LSTM, Gated Recurrent Unit (GRU), Stacked GRU, Bidirectional LSTM, Bidirectional GRU, and a Hybrid model to predict the Moving Average Convergence Divergence (MACD) of National Stock Exchange (NSE) of India, i.e., NIFTY50 market index. Some hyperparameters are also considered when training these models, as these hyperparameters control the behavior and performance of such models. Two experiments are carried out to train these RNN-based models: manually selecting and tuning hyperparameters. Metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Squared Percent Error (MAPE) are used to assess performance. Both experimental results show that the Bidirectional GRU model is the most effective at predicting MACD values in India's NIFTY50 stock market index.

## Keywords
RNN, LSTM, Stacked LSTM, Bidirectional LSTM, GRU, Stacked GRU, Bidirectional GRU, Stock Market, MACD, NIFTY50.

## 1. INTRODUCTION
Since the stock market is intricate, unpredictable, and vulnerable to various factors that might affect its behavior, predicting stock prices has long been a difficult challenge in finance. Deep learning algorithms are increasingly being considered for use in stock market forecasting because they have demonstrated promise in spotting complex patterns in financial data and generating better predictions. Nowadays, sequential DL models (RNN and its variants) have been widely employed for stock market prediction. Researchers have applied numerous optimization strategies to enhance the performance of these models. For instance, Hyejung Chung et al. [1] suggested a hybrid strategy that uses GA to enhance the LSTM architecture and the temporal window's size. In another study by Abba Suganda Girsang et al. [2], A novel metaheuristic optimization technique called Search Economics was integrated with LSTM. By reaching a low RMSE, MAE, and MAPE, their proposed Model, SE-LSTM, beat the non-optimized LSTM and ARIMA models on the stock price prediction. The authors in [3] developed a model based on the LSTM and Differential Evolution algorithms for optimizing network hyperparameters to estimate a company's stock price for the next day. Nikolaos Gorgolis et al. [4] show that GA is an efficient and viable method for fine-tuning an RNN model. These studies show how well optimization approaches boost deep learning-based models' performance in stock market forecasting.

In this article, we evaluated the performance of seven sequential DL models, namely LSTM, GRU, Stacked LSTM, Stacked GRU, Bidirectional LSTM, Bidirectional GRU, and a hybrid model to forecast the price of the Nifty50 index. The dataset contains historical details of the Nifty50 index and its technical indicators from 2013 to 2023. Historical parameters include daily Open, High, Low, Close, Volume (OHLCV), and Adjusted Close price. In addition, we derived the technical indicators like MACD, Relative Strength Index (RSI), Exponential Moving Average (EMA), Simple Moving Average (SMA), and the Stochastic Oscillator (SO) with these features. We choose MACD as the predicting feature with timesteps of 10 days. The study aimed to pinpoint the ideal model. The findings of this study can help financial experts and stock market investors make wise choices.

The work structure of this article is as follows: Section II includes a brief outline of previous studies. Section III presents the methodology applied in this article, followed by experimental results and analysis in section IV. Finally, section V presents the conclusion, and section VI covers all the references of the proposed work.

## 2. LITERATURE SURVEY
The ability to predict stock prices effectively has become one of the interesting research problems in recent years. Many methods have been put forth. This study concentrates on recent studies that used various optimization strategies and deep learning models to forecast stock prices.

To predict the daily Korea Stock Price Index (KOSPI) price, Hyejung Chung et al. [1] proposed a hybrid technique that blends Genetic Algorithm (GA) and LSTM networks. The authors determined the ideal temporal window size and topology for the LSTM network using GA. The experimental findings demonstrated that their hybrid approach performed better in prediction accuracy than the Conventional LSTM and Autoregressive Integrated Moving Average (ARIMA) models.

In another study, Abba Suganda Girsang et al. [2] employed the Search Economics metaheuristic optimization algorithm to enhance the LSTM model's stock price prediction ability. The authors used the auto Arima function to test the performance of the optimized LSTM model against an unoptimized LSTM model and the ARIMA model. According to the result, the optimized LSTM model outperformed the other two models regarding RMSE, MAE, MAPE, and R2 scores.

Ehsan Rokhsatyazdi et al. [3] introduced a unique RNN-based model on the LSTM and Differential Evolution (DE) algorithm to estimate a company's stock price for the next trading session.

The DE algorithm is utilized to optimize the hyperparameters of LSTM to achieve a lower RMSE for prediction. The study focused on tuning ten network hyperparameters relevant to the identification of temporal patterns of a particular dataset. The proposed model outperformed the top statistical forecasting models, including NAIVE, ETS, and SARIMA, with an objective value of 8.092 RMSE. Nikolaos Gorgolis et al. [4] show a simple GA approach used for the hyperparameter tuning of a model and achieves tuning efficiency without an exhaustive search.

Five deep learning-based models (two CNN and three LSTM models) for the prediction of the NIFTY50 index of values are proposed by Jaydip Sen et al. [5]. Their study found that the univariate CNN model, which used data from the previous week as its input, was the correct and the fastest in its execution. On the other side, it was discovered that the encoder-decoder Convolutional Neural Network (CNN) LSTM model, which uses the data from the past two weeks as input, is the least accurate. A comparison between two RNN-based models, i.e., LSTM and Bidirectional LSTM, is proposed by Md Arif Istiake Sunny et al. [6] for stock market prediction, using the modified parameters between these two models BI-LSTM model shows the best result by generating lower RMSE compared to LSTM model. Anita Yadav et al. [7] proposed two experiments. In experiment one, they compared stateless and stateful LSTM models for Indian stock market prediction based on the result of standard deviation plot, box-plot, and whisker plots stateless LSTM model was more stable than the stateful one. In experiment two, the number of hidden layers for the model varied from one to seven. The findings suggested that having just one hidden layer was the optimal design for RMSE, and the test confirmed by the one-way ANOVA test was confirmed.

To predict the volatility of the Chinese stock market, Weiling Chen et al. [8] offer various technical indicators with a sentiment influence feature, which they then incorporate into a two layers RNN-GRU model. The experiment results demonstrate that their method and characteristics can predict well with minor mistakes with the GRU model. For both short- and long-term stock market forecasting using S&P500 index historical data, Khaled A. Althelaya et al. [9] compare and assess the utilization of Stacked LSTM, Stacked GRU, Bidirectional LSTM, and Bidirectional GRU architectures, their findings demonstrate that stacked architectures, such as Stacked LSTM and Stacked GRU, outperformed bidirectional architecture-developed Bidirectional LSTM and Bidirectional GRU models in terms of performance. In the method for stock market closing price prediction presented in the study of Mehrnaz Faraz et al. [10], some technical indicators and oscillators are made and added to the dataset as additional characteristics. Simulation experiments are run on the S&P 500 stock index, and the findings show that AutoEncoder (AE) LSTM is virtually more accurate than Generative Adversarial Network (GAN) at predicting the daily price.

# 3. METHODOLOGY

## 3.1 Data Description

In this work, the historical data for the Nifty50 stock market index, which represents the top 50 businesses listed on the NSE of India, was obtained from Yahoo Finance [11]. The data was collected from January-01-2013 to March-05-2023, and the information comprised daily stock prices for the Nifty50 index, including the OHLCV and Adjacent Close Volume.

To increase the predictive power of the deep learning models, we used technical indicators such as the RSI, MACD, EMA, and Stochastic Oscillator; SMA was added to the dataset in addition to the stock prices. The Python pandas_ta [12] package calculated these technical indicators.

The MACD was calculated with a short period of 12, a long period of 26, and a signal period of 9, whereas the stochastic oscillator, which displays two lines: %K & %D was calculated with a window of 14. Using the Python Pandas [13] package, these indicators were added as additional columns to the Nifty50 dataset.

The data was cleaned and pre-processed using an SK-Learn[14] min-max scaler, which scales each data point in a range of 0 and 1, and the backward fill approach is used to fill for any missing values to guarantee the accuracy of the dataset. With timesteps of 10 days (past two weeks' data), MACD is selected as the predictive feature as it enables traders to determine short-term trends. The time-series split method was used to divide the selected feature into training and testing sets in a ratio of 80:20.

The resulting training and testing sets had 2,000 and 499 data points, each representing a single MACD value. These variables are then used to train and evaluate our proposed deep-learning models for stock market prediction, as described in the following sections.

## 3.2 Models Used

As we discussed, we use seven different RNN-based models such as LSTM, GRU, SLSTM, SGRU, BI-LSTM, BI- GRU and a Hybrid model (combination of BI-LSTM and a GRU layer) for predicting the stock market of India, i.e., NIFTY50. The models were implemented in Python using Keras deep learning library with Tensorflow [15] backend.

## A. RNN

An RNN is a Deep Neural Network (DNN) that handles sequential data. RNNs are commonly implemented for tasks like time series prediction, speech recognition, and natural language processing. Also, they have been used for other structures line LSTM and GRU, designed to address the vanishing gradient issue, which is the major problem with RNNs as it can restrict their capacity to capture long-term dependencies and their computational cost due to sequential processing. Fig 1 shows the internal structure of RNN.

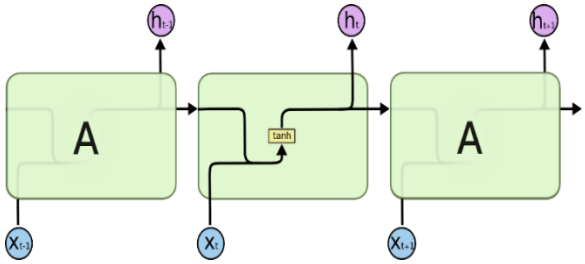


**Fig. 1: Internal structure of RNN [16]**

## B. LSTM

LSTM, a variant of RNN, is designed to solve the vanishing gradient issue and enhance the network's capacity to recognize long-term dependencies in sequential data. LSTM can also perform all the tasks of RNN, such as speech recognition, Natural Language Processing (NLP), and time series prediction, as the LSTM network employs a unique cell unit with gates such as an input gate, forget gate, and output gate that selectively allow information to be stored or discarded from the cell state. This allows the network to retain important information longer while avoiding the issue of Vanishing and exploding gradients. Fig 2 shows the repeating structure of the LSTM network.

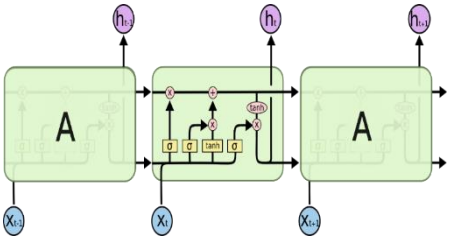


**Fig. 2: Repeating Structure of LSTM network [16]**

## C. GRU

GRU is also a type of RNN architecture; like LSTM, GRU uses gates such as reset gate and update gate to selectively renew and delete the hidden state of the network, thus allowing the network to learn long-term dependencies while avoiding vanishing gradient problem that generally occurs with traditional RNNs. GRU is simple as well as faster to train as compared to LSTM architecture. Fig 3 shows the structure of GRU.

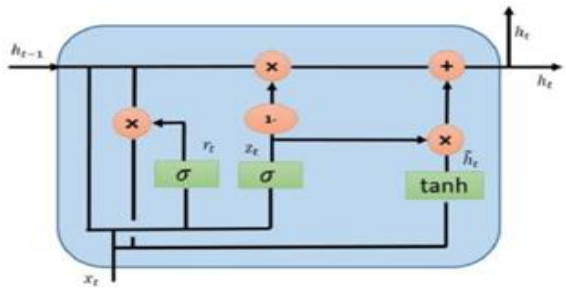


**Fig.3: Structure of GRU [17]**

## D. Stacked LSTM AND Stacked GRU

Both Stacked LSTM and Stacked GRU are RNN architectures used for sequential data processing as both Stacked LSTM and Stacked GRU are made up of multiple layers of recurrent units of their defined types, with the output of one layer serving as the input to the next layer in a stacked network. The key difference is the gate mechanism used inside them; Stacked LSTM uses LSTM gating mechanism. On the other side, a Stacked GRU uses the gating mechanism of a GRU.

## E. Bidirectional LSTM

Bidirectional Directional LSTM is a variant of LSTM that processes the sequential data in forward and backward directions. A Bidirectional LSTM architecture comprises two separate LSTMs: one that processes the sequence forward and another that processes the sequence backward. Each layer of the LSTM has its own set of weights and biases. The forward LSTM computes a forward hidden state at each time step based on the current input and the previous hidden state in the forward direction. Similarly, the backward LSTM computes a backward hidden state based on the current input and the previous backward direction. The final hidden state concatenates the forward and backward hidden states at each time step. Bidirectional LSTMs have been used in various applications, including natural language processing, speech recognition, and time series analysis, as they can learn short-term and long-term dependencies in sequential data, making them ideal for modeling complex sequences. Fig 4 shows the architecture of Bidirectional LSTM.

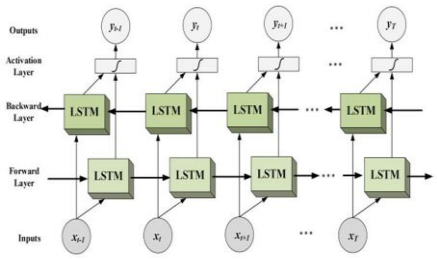


**Fig.4: Architecture of BI-LSTM [18]**

## F. Bidirectional GRU

Similar to Bidirectional LSTM, Bidirectional GRU also processes sequential data in forward and backward directions; in Bidirectional GRU, there are two different layers of GRU, one of which processes the sequence in the forward direction and the other which processes the sequence in the backward direction. The Bidirectional GRU architecture is depicted in Fig. 5.

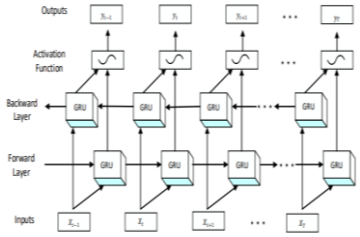


**Fig.5: Architecture of BI-GRU [19]**

## G. Hybrid Model

Our proposed work also develops a hybrid model with two recurrent layers: the GRU layer and a Bidirectional LSTM layer. The Bidirectional LSTM layer does forward and backward processing on the input sequence and produces output. This output is then passed on to the GRU layer, which performs additional processing on the sequence to identify any pertinent relationships. This model may capture long-term dependencies in sequential data and enhance task performance by integrating the strengths of both LSTM and GRU.

## 3.3 Neuro-Evolution for Hyperparameter Tuning

Neuro-evolution is a wrapper and slightly modified implementation of Matt Harvey's [20] approach, proposed by Alexander Osipenko [21]. In their study, Osipenko briefly mentions neuro-evolution as an alternative to traditional optimization methods for neural networks. Instead of designing and optimizing the architecture of a neural network by hand, neuro-evolution uses evolutionary algorithms such as genetic algorithms to evolve the architecture of the network automatically. Neuro-evolution is frequently used to solve problems where traditional optimization methods fail, such as those with high-dimensional search spaces or non-convex objective functions. It is also helpful when the optimal network architecture is unknown or difficult to define.

## 4. PROPOSED WORK

To achieve our objective, we conduct two experiments; in the first experiment, we manually select some hyperparameters like the number of epochs, dropout, layers, neurons, batch size, hidden layer activation function, output layer activation function, losses, metrics and optimizer to train the models.

Comparing these models' performances to identify which one is best at forecasting stock market prices is our primary goal. MAE, MSE, and RMSE are some of the metrics we use to assess the performance of the models.

In the second experiment, we incorporate neuro-evolution to enhance the hyperparameters of the models. We change the number of neurons, layers, epochs, batch size, dropout, loss, metrics, and activation functions of the hidden and final layers. The models are then trained using the ideal hyperparameters discovered by the neuro-evolution method, which runs with a generation size of 10 and population size of the total numbers of tuning hyperparameters required; their performance is assessed using the same set of metrics.

## 4.1 Hyperparameters

**Table 1: Hyperparameters for Experiment 1**

| Hyperparameters | Values |
|---|---|
| Epochs | 50 |
| Batch size | 32 |
| Number of layers | 4 |
| Number of neurons | 60 |
| Dropout | 0.3 |
| Optimizer | Adam |
| The activation function for hidden layers | Tanh |
| The activation function of the output layer | Linear |
| Loss function | MAE |
| Metrics | MSE |

**Table 2: Hyperparameters for experiment 2 (Before tuning)**

| Hyperparameters | Values |
|---|---|
| Epochs | 10 to 250 |
| Batch size | [10, 16, 32, 40, 64, 128] |
| Number of layers | 1 to 7 |
| Number of neurons | 15 to 160 |
| Dropout | [0.1, 0.2, 0.3, 0.4, 0.5] |
| Optimizer | [Adam, nadam] |
| The activation function in hidden layers | [tanh, relu] |
| The activation function in the output layer | Linear |
| Loss function | [MAE, MSE] |
| Metrics | [MSE, MAE] |

**Table 3: Hyperparameters for experiment 2 (After tuning)**

# 5. RESULT AND ANALYSIS

In this section, we present the result of our proposed seven models such as LSTM, GRU, Stacked LSTM, Stacked GRU, Bi-Directional LSTM, Bi-Directional GRU, and Hybrid model; the details about these models were presented in section III. We performed two individual experiments to train all the models with different hyperparameters to evaluate which is best out of seven in forecasting MACD values of the NIFTY50 index. The evaluation metrics like MSE, MAE, RMSE, and MAPE are applied to assess the performance of each model. The formula for these metrics is presented below-.

$$MSE = \frac{\sum(y_i - \hat{y}_i)^2}{n}$$

Where,

- $y_i$ - is the $i^{th}$ observed value.
- $\hat{y}_i$ - is the corresponding predicted value.
- n - the number of observations.

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

Where,

- $y_i$ - is the $i^{th}$ observed value.
- $\hat{y}_i$ - is the predicted value of y
- N – the number of observations

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

Where,

- $y_i$ - is the $i^{th}$ observed value.
- $\hat{y}_i$ - is the corresponding predicted value.
- N - the number of observations.
- √ - Square root

$$\mathrm{M} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

- n - is the number of fitted points,
- $A_t$ - is the actual value,
- $_{It}$ - is the forecast value.
- Σ - is summation notation (the absolute value is summed for every forecasted point in time).

**Table 4: Result of each model obtained from experiment 1**

| Models | MSE | MAE | RMSE | MAPE |
|---|---|---|---|---|
| LSTM | 7069.64 | 67.74 | 84.08 | 1.07 |
| SLSTM | 663.93 | 21.15 | 25.77 | 0.37 |
| GRU | 2192.88 | 38.11 | 46.83 | 0.45 |
| SGRU | 559.37 | 19.28 | 23.65 | 0.27 |
| BI-LSTM | 1831.3 | 35.5 | 42.79 | 0.58 |
| BI-GRU | 523.12 | 17.39 | 22.87 | 0.38 |
| HYBRID | 580.72 | 19.03 | 24.1 | 0.34 |

**Table 5: Hyperparameters for experiment 1**

| Hyperparameters | Values |
|---|---|
| Epochs | 193 |
| Batch size | 16 |
| Number of layers | 1 |
| Number of neurons | 43 |
| Dropout | 0.3 |
| Optimizer | Adam |
| The activation function in hidden layers | Relu |
| The activation function in the output layer | Linear |
| Loss function | MSE |
| Metrics | MAE |

## Experiment 1: Manual Selection of Hyperparameters

In the first experiment, we manually select some hyperparameter values shown in Table 1. We noted performances in terms of MSE, MAE, RMSE, and MAPE of all these seven models. Table 4 presents the performance result of these seven models obtained from experiment 1.

Based on the result shown in Table 4, the model which outperforms the rest of the six models in terms of performance metrics was a bidirectional gated recurrent unit in short BI-GRU as BI-GRU has much smaller values of MSE, MAE, RMSE, and MAPE, which is 382.95, 14.73, 19.57 and 0.32 as compared to LSTM, SLSTM, GRU, SGRU, BI-LSTM, and Hybrid model. Conversely, SLSTM performs worst in predicting the MACD values for NIFTY50. Fig 6 depicts the graphical representation of MACD values predicted by the well-performed Bi-GRU model and their actual values for NIFTY50.

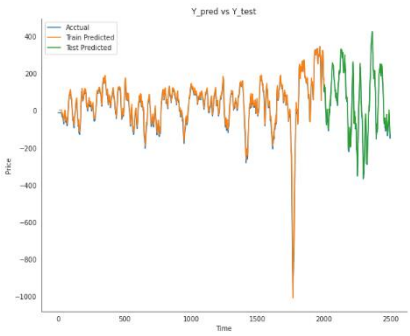


**Fig. 6: Predicted vs. Actual values from Experiment 1**

## Experiment 2: Hyperparameters Tuning

In experiment second, the hyperparameters selected for tunning are shown in Table 2, we tuned these hyperparameters using neuro-evolution with ten iterations, and the population size of the total number of hyperparameters required for tunning in this experiment is ten as we try to tune up ten hyperparameters required for training the individual Model, The hyperparameters obtained after successful tunning is present in Table 3.

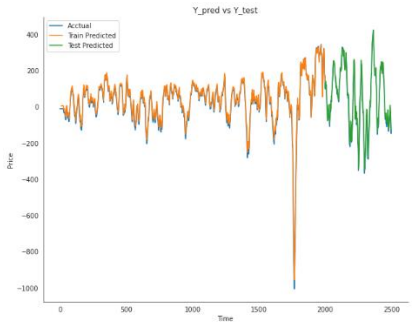


**Fig 7: Predicted Vs. Actual Values from Experiment 2**

We trained and validated the seven selected models with these obtained hyperparameters (Table 3). After running them through these hyperparameters, each model's result is presented in Table 6.

**Table 6: Result of each model obtained from experiment 2**

| Models | MSE | MAE | RMSE | MAPE |
|---|---|---|---|---|
| LSTM | 1081.45 | 25.3 | 32.89 | 0.49 |
| SLSTM | 13330.14 | 93.95 | 115.46 | 1.35 |
| GRU | 841.3 | 23.95 | 29.01 | 0.37 |
| SGRU | 595.26 | 19.29 | 24.4 | 0.35 |
| BI-LSTM | 1130.48 | 28.67 | 33.62 | 0.42 |
| BI-GRU | 382.95 | 14.73 | 19.57 | 0.32 |
| HYBRID | 502.61 | 17.77 | 22.42 | 0.32 |

According to the result shown in Table 5, Bidirectional GRU appears to have the best performance of the seven models, with the lowest MSE (523.12), MAE (17.39), and RMSE (22.87) values and reasonably low MAPE (0.38) value. On the other hand, the Stacked GRU model has slightly higher MSE, MAE, and RMSE values, which may indicate that it predicts the target variable less accurately than Bidirectional GRU. The LSTM model has the highest RMSE value, indicating that its predictions are more inaccurate from the actual values, making it the worst among the seven. Fig 7 shows the graphical representation between predicted MACD values by the best model obtained from experiment 2 and actual MACD values for NIFTY50.

## 6. CONCLUSION

Based on the result of the two experiments conducted to find the best prediction Model for the Indian NIFTY50 index price, the Bidirectional GRU model provided the best results in comparison to others based on evaluation metrics used in experiment 1 (MSE, MAE, RMSE, and MAPE). In experiment 2, neuro-evolution was used to tune hyperparameters after that, and then our proposed models were trained with these tuned hyperparameters. The result shows that the Bidirectional GRU and Stacked GRU models used the MAPE metrics best. Significantly, the Bidirectional GRU model outperformed all other models examined in both experiments regarding its ability to predict the MACD value of the NIFTY50 stock market index. In addition to giving traders and investors a more reliable tool for making educated decisions, this advancement in predictive modeling also offers new directions for research in finance and deep learning. The Bidirectional GRU model offers

a substantial improvement in our understanding of market sentiment and forecasting and has the potential to transform stock market analysis.

# 7. REFERENCES

[1] Chung, Hyejung, and Kyung-shik Shin. "Genetic algorithm-optimized long short-term memory network for stock market prediction." Sustainability 10, no. 10 (2018): 3765.

[2] Girsang, Abba Suganda, Fernando Lioexander, and Daniel Tanjung. "Stock price prediction using LSTM and search economics optimization." IAENG International Journal of Computer Science 47, no. 4 (2020): 758-764.

[3] Rokhsatyazdi, Ehsan, Shahryar Rahnamayan, Hossein Amirinia, and Sakib Ahmed. "Optimizing LSTM Based Network For Forecasting Stock Market." In 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1-7. IEEE, 2020.

[4] Gorgolis, Nikolaos, Ioannis Hatzilygeroudis, Zoltan Istenes, and Lazlo–Grad Gyenne."Hyperparameter optimization of LSTM network models through genetic algorithm." In 2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA), pp. 1-4. IEEE, 2019.

[5] Sen, Jaydip, Sidra Mehtab, and Gourab Nath. "Stock price prediction using deep learning models." Lattice: The Machine Learning Journal 1, no. 3 (2020): 34-40.

[6] Sunny, Md Arif Istiake, Mirza Mohd Shahriar Maswood, and Abdullah G. Alharbi. "Deep learning-based stock price prediction using LSTM and bi-directional LSTM model." In 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), pp. 87-92. IEEE, 2020.

[7] Yadav, Anita, C. K. Jha, and Aditi Sharan. "Optimizing LSTM for time series prediction in Indian stock market." Procedia Computer Science 167 (2020): 2091-2100.

[8] Chen, Weiling, Yan Zhang, Chai Kiat Yeo, Chiew Tong Lau, and Bu Sung Lee. "Stock market prediction using neural network through the news on online social networks." In 2017 international smart cities conference (ISC2), pp. 1–6. IEEE, 2017.

[9] Althelaya, Khaled A., El-Sayed M. El-Alfy, and Salahadin Mohammed. "Stock market forecast using multivariate analysis with bidirectional and stacked (LSTM, GRU)." In 2018 21st Saudi Computer Society National Computer Conference (NCC), pp. 1–7. IEEE, 2018.

[10] Faraz, Mehrnaz, Hamid Khaloozadeh, and Milad Abbasi. "Stock market prediction-by-prediction based on autoencoder long short-term memory networks." In 2020 28th Iranian Conference on Electrical Engineering (ICEE), pp. 1-5. IEEE, 2020.

[11] URL: https://finance.yahoo.com

[12] URL: https://pypi.org/project/pandas-ta/

[13] URL: https://pypi.org/project/pandas/

[14] URL: https://scikit-learn.org/stable/

[15] URL: https://www.tensorflow.org/

[16] C. Olah, “Understanding Lstm,” accessed: 2020-03-12. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[17] Muhammad, L. J., Ahmed Abba Haruna, Usman Sani Sharif, and Mohammed Bappah Mohammed. "CNN-LSTM deep learning based forecasting model for COVID-19 infection cases in Nigeria, South Africa, and Botswana." Health and Technology (2022): 1–18.

[18] “Deep dive into Bidirectional LSTM” 2019 [Online].

[19] Available:https://www:i2tutorials:com/deep-dive-into-bidirectional-lstm/

[20] Ju, Yun, Min Zhang, and Huixian Zhu. "Study on a New Deep Bidirectional GRU Network for Electrocardiogram Signals Classification." In 3rd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA 2019), pp. 355-359. Atlantis Press, 2019.

[21] URL: https://github.com/harvitronix/neural-network-genetic-algorithm

[22] Osipenko, Alexander. "Genetic algorithms and hyperparameters—Weekend of a Data Scientist."(2019) [Online]. Available:https://medium.com/cindicator/genetic-algorithms-and-hyperparameters-weekend- of-a-data-scientist-8f069669015e