Performance Evaluation of Bidirectional Long Short-Term Memory Optimizers For Prediction of Stock Market Price

Tanushree Dwibedi [1], Nandana Devi[2], Smita Rath[3,*][0000-0003-0547-6609],Monalisa Panda[4],Deepak Kumar Patel[5]

[1,3 *,5]Department of Computer Science and Information Technology,ITER, Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar, Odisha,India
[2,4] Department of Computer Science and Engineering,ITER, Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar, Odisha,India.
tanushreedwibedi20@gmail.com,devinandanasrinivas@gmail.com,

[*]smitarath@soa.ac.in, monalisapanda@soa.ac.in, deepakpatel@soa.ac.in

Abstract. The stock market provides a gateway for analyzing a company's present and future financial value to the world of investors as a potential ground for receiving better returns than traditional savings or bonds. This paper presents a deep learning model-BiLSTM (Bidirectional Long Short-Term Memory) neural networks predicts the closing prices of main stocks: Meta, Amazon, Apple, and Tesla. The study leverages socioeconomic data as predictors to improve the accuracy of the forecasts. Comparative analysis is done on the performance of six well-known optimization algorithms in the Keras machine learning library—Adam, Adagrad, Stochastic Gradient Descent, RMSprop, and others—that are utilized to train the LSTM model.Performance metrics, such as MAE and RMSE, are used to evaluate model performance. The results allow one to understand the most efficient method for optimizing the prediction of stock prices and practical implications in terms of offering data-driven strategies for decision-making by financial analysts and investors.

Keywords: Stock market forecast ,Deep learning, Long short-term memory (LSTM), Bidirectional Long Short-Term Memory(BiLSTM),Optimizers

1 Introduction

In particular, artificial intelligence (AI) and machine learning (ML)[1], being able to sift through immense databases, identify trends, and produce predictions that usually do not come out through other methodologies, have transformed stock market forecasting. AI approaches, especially neural networks, natural language processing (NLP)[2], and sentiment analysis, increasingly are applied on news articles, social media sentiment, and historical stock prices so that more sensitive and correct market predictions become possible. Traditional methods of stock price analysis, such as

Simple Moving Average (SMA)[3], Relative Strength Index (RSI)[4], Bollinger Bands, Exponential Moving Average (EMA), and Linear Regression, have been the building blocks for decades. These approaches are often based on historical price data, possess low predictive power, and do not take into account external factors such as market sentiment or macroeconomic indicators.

New techniques in machine learning try to overcome these limitations by using complex models and algorithms. Techniques include Random Forests, XGBoost, LightGBM, AutoRegressive Integrated Moving Average (ARIMA)[5-7], Reinforcement Learning (RL), and NLP for sentiment analysis, which allows capturing non-linear relationships, integration of different data sources, and improvement in predictive accuracy. Among these, LSTM neural networks have come to the forefront because of their ability to process sequential data effectively and predict time-series trends.

This paper utilizes the LSTM model to predict closing stock prices of leading firms, such as Tesla, Meta, Apple, and Amazon, using Yahoo Finance. The research is based on the strengths of the LSTM model in capturing temporal dependencies and analyzing patterns, where it will be demonstrated and explored for its potential predictive capabilities and its ability to outperform traditional methods in the forecasting of stock prices.

Over the years, various artificial neural network (ANN) architectures are used to forecast stock market prices are as follows:

i. Feedforward Neural Network (FNN) / Multi-Layer Perceptron (MLP): ANN consists of only one hidden layer where the data moves in one way, from input to output, without any looping or feedback connections(see Fig.1)[8].

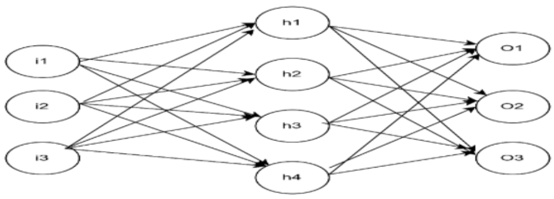


Fig. 1. FNN Architecture

ii. Deep Neural Network (DNN): an advanced version of FNN that consists of more than one hidden layer hence can process the data and give better results(see Fig.2)[9-11].

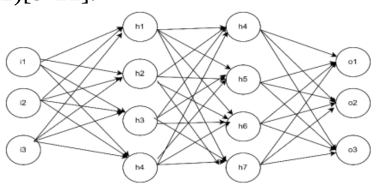


Fig. 2. DNN Architecture

iii. Recurrent Neural Network (RNN): A type of artificial neural network (ANN) modelled to specifically train sequential data. It also has a hidden layer, which is known as the memory state; unlike the traditional methods like FNN and DNN, Because of its directed cycle connections, RNNs are able to store and utilize information from prior inputs to generate exact outputs[12-15]. Although RNN is an improved version of ANN that can form loops, store previous inputs in its hidden layer, and generate output. It can only store the most recent output in the hidden layer, cannot retain information for the long term, and has shortcomings like the gradient variance and gradient explosion problem(see Fig 3).

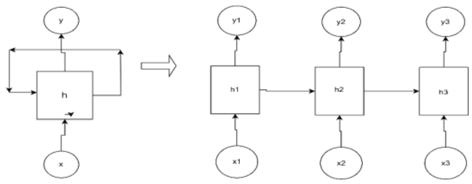


Fig. 3. RNN Architecture

Some traditional methodologies for the predictions of the stocks in the marketplace include the Statistical models such as ARIMA or Exponential smoothing and Generalized Autoregressive Conditional Heteroskedasticity, that typically assume a straight line fit under stationary and stable conditions that frequently do not have place in any real world-data[16]. Ensemble methods like Random Forest (RF), Gradient Boosting Machines (GBM), and XGBoost use the strength of multiple models to enhance the accuracy of the forecast[17]. These methods also help overcome some of the limitations of traditional approaches by capturing non-linear relationships and interactions between features. The existing literature has shown that ensemble methods improve the accuracy but rarely discusses their performance under high-frequency noisy financial data conditions. There is a lack of comparative studies on the effect of optimizers and loss functions specific to financial time series data. Future research may evaluate the performance of BiLSTM using diverse optimization strategies.

1.1 Proposed Method for Selecting the best Optimizer

The LSTMs model is being used in this research paper to predict the closing price of selected companies listed at Yahoo Finance(see Fig 4 and Fig. 5). LSTM has been developed by Hochreiter and Schmidhuber, which is advanced version of RNN specially proposed to overcome the limitations like the inability of RNNs to capture long dependencies properly and also problems like gradients vanishing or explosion during forward or backward pass[18].

Unlike traditional RNNs, which rely on only a single hidden state, the LSTM incorporates a memory cell that allows the model to retain and manage data for longer periods[19]. For instance, this is achievable with the help of the following three specialized gates within the hidden layer:

- Forget Gate: Determines whether certain specific information in the memory cell will continue to be relevant, with irrelevant data being discarded by this means so that irrelevant patterns are not taken account of.
- Input Gate: Controls the admission of new information into the cell state, so the model dynamically updates its memory.
- Output Gate: Controls the amount of information flowing out from the cell state to be leaked as the hidden state of the current time step. This allows the network to output predictions based on relevant contexts.

In order to improve the precision and efficiency of the LSTM model, the paper analyzes various optimization algorithms: Adam, Adagrad, RMSprop, and Stochastic Gradient Descent (SGD)[20-21]. Every optimizer has specific properties affecting model convergence and performance. In order to identify the best optimizer for stock price prediction, the goal is to compare them systematically in a robust and reliable manner.

This method uses the strengths of LSTM in capturing temporal dependencies while focusing on the role of optimization in enhancing model performance and the steps followed are provided in the Table 1 for prediction of stock prices.

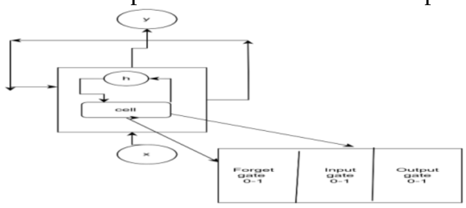


Fig. 4. LSTM Architecture

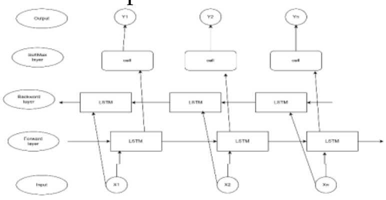


Fig. 5. BiLSTM Architecture

Table 1. Pseudocode for Stock Price Prediction using LSTM and Bidirectional LSTM

```
1. Load the Dataset:
  - Import `yfinance` library.
  - Download stock data from Yahoo Finance.
  - Store the dataset.
2. Data Preprocessing:
  - Extract the "Close" column as the target variable.
```

- Use `MinMaxScaler` to normalize the data between 0 and 1.
- Split the data into 80% training set and 20% testing set.

3. Creating Sequences:
   - Define a function `create_dataset(data, time_steps)`:
     - For each time step in the dataset, create sequences of 60 values to predict the next closing price.
     - Return X (features) and y (target).
4. Reshaping for LSTM:
   - Reshape X_train and X_test into the format [samples, time_steps, features].
5. Building the LSTM Model:
   - Initialize a sequential model.
   - Add two LSTM layers with appropriate units.
   - Add Dense layers to output predictions.
6. Training the Model:
   - Compile the model with optimizers like Adam, Adagrad, RMSprop, and MSE loss.
   - Train the model using X_train and y_train for 2000 epochs.
7. Prediction:
   - Use the trained model to predict on training (X_train) and testing (X_test) datasets.
8. Plotting Results:
   - Plot actual prices vs. predicted prices for training and testing sets.

9. Bidirectional LSTM Implementation:
   - Define a new model:
     - Add `Bidirectional(LSTM(...))` layers.
     - Set `return_sequences=True` for the first Bidirectional LSTM layer.
     - Set `return_sequences=False` for the final layer.
   - Add Dense layers to output predictions.
10. Training the Bidirectional LSTM:
    - Train the model on X_train and y_train for 2000 epochs with batch size 64.
    - Use Adam optimizer and MSE loss.
11. Evaluation:
    - Predict on training and testing datasets.
    - Calculate evaluation metrics: MSE, RMSE, and MAE.
12. Plotting Results for Bidirectional LSTM:
    - Plot actual vs. predicted prices for training and testing sets.
13. Key Considerations:

```
   - Ensure input shape is [samples, time_steps, features].
   - Inverse transform predictions to the original scale for evaluation.
# Expected Output:
  - A plot showing actual vs. predicted prices for training and testing datasets.
```

2 Simulation

The flowchart in Fig.6 provides a structured process of building and evaluating a time series prediction model, specifically using data from Yahoo Finance. The process begins with data loading having: Open, Close, High, Low, and Volume, followed by data handling, which involves data processing and sequence creation to prepare the data for modeling. Next, the models are built using LSTM (Long Short-Term Memory) and Bidirectional LSTM architectures. This training phase includes prediction followed by evaluation, where the performance of the model is tested. The process finally ends with output and visualization, where the results are presented using performance metrics and plotting tools.

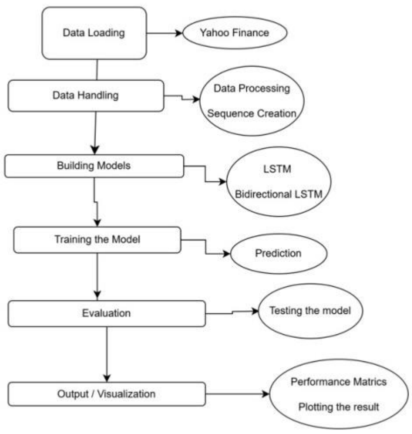


Fig. 6. Proposed Flowchart for Forecasting Closing Price of a Stock with different optimizers tested during training and testing

The following table Table 2 provides the hyper parameter tuning of the model used with various optimizers below :

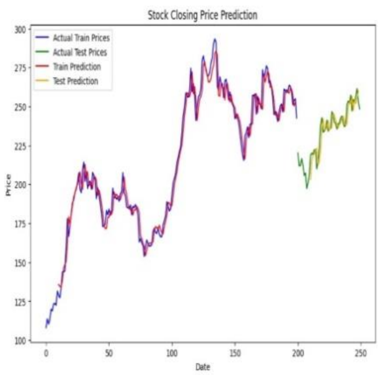
Table 2. Hyper parameter Tuning using Different Optimizers

Optimizer	Learning rate	Batch Size	Epoch
Adam optimizer	0.001	32	1000
Adam optimizer	0.0001	64	2000
RMS prop optimizer	0.001	32	1000

RMS prop optimizer	0.0005	64	2000
SGD optimizer	0.01	32	1000
SGD optimizer	0.001	64	2000
Adagrad optimizer	0.005	32	1000
Adagrad optimizer	0.0001	64	2000

3 Results and Discussions

The graph illustrates the use of stock price data for four companies-Tesla, Meta, Apple, and Amazon-for training an LSTM model to predict closing prices(see in Fig 7-10). Input data consists of the daily 'Close' prices which is preprocessed using reshape, normalizing it by using MinMaxScaler, so the output will be scaled in a value range between 0 to 1. It divides dataset into training set, 80 % and testing set, 20 % with sequence closing price creation using time step equals 10 to predict the future closing prices. The LSTM model is built, trained, and used to generate predictions for both the training and testing datasets. These predictions are then inverse-transformed to their original scale for meaningful interpretation. In the graph, the blue line represents the actual closing prices from the training dataset, while the green line shows the actual prices from the testing dataset. The red line shows the prediction that the model is making for the training data. It is shown in red to highlight how it manages to capture historical trends and fluctuations. The orange line depicts the values that the model has predicted for the testing data, thus showing that the model works perfectly for generalizing data, unseen data. This is how a clear comparison can be drawn between actual and predicted prices. The graph therefore provides valuable insights into how the model predicts and where actual prices do not match those predictions, hence giving a hint on its usability in determining future stock performance by investors and analysts.



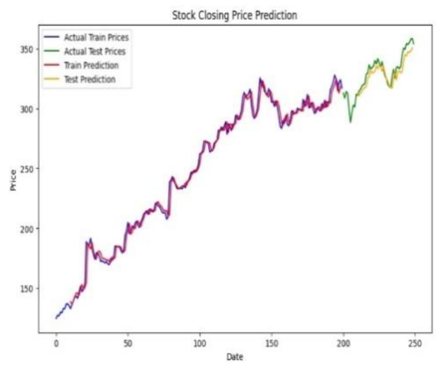
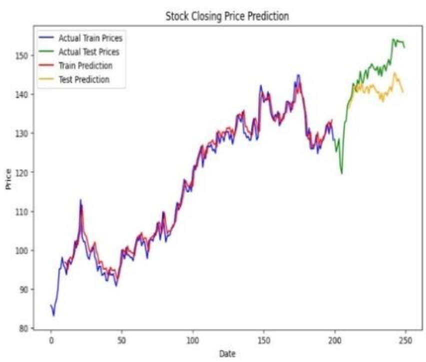


Fig.7. Stock prediction of Actual vs Predicted for TESLA

Fig. 8. Stock prediction of Actual vs Predicted for META



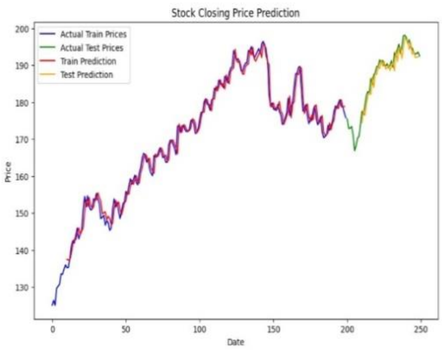


Fig. 9. Stock prediction of Actual vs Predicted for APPLE

Fig. 10. Stock prediction of Actual vs Predicted for AMAZON

Table 3. Error detection in Training Data using Different Optimizer

Optimizer	**Company**	**Training Data**			**Testing Data**		
		MAE	**MSE**	**RMSE**	**MAE**	**MSE**	**RMSE**
Adam Optimizer	**META**	3.944	31.602	5.621	5.271	41.328	6.428
	TESLA	5.352	48.138	6.932	4.818	40.92	6.347
	APPLE	4.520	4.520	2.126	1.633	4.124	2.030
	AMAZON	1.843	5.643	2.375	5.653	43.72	6.612
RMS prop optimizer	**META**	4.001	35.068	5.486	3.271	21.32	4.428
	TESLA	5.001	42.068	6.486	5.303	46.04	6.785
	APPLE	1.613	4.247	2.061	1.740	4.599	2.144
	AMAZON	1.843	6.643	2.375	4.653	25.72	5.612
SGD optimizer	**META**	7.137	88.836	9.632	10.271	146.3	10.428
	TESLA	11.13	185.836	13.632	5.875	54.81	7.403
	APPLE	3.208	17.029	4.126	4.103	23.32	4.829

	AMAZON	1.843	5.643	2.375	4.653	28.72	5.612
Adag-rad opti-mizer	**META**	3.409	15.348	3.713	11.93	175.4	3.247
	TESLA	11.72	204.098	14.286	5.958	55.66	7.461
	APPLE	3.29	17.374	4.168	4.347	25.45	5.044
	AMAZON	3.086	15.378	3.921	4.717	29.47	5.428

Table 4. Statistical value of t-satatistic and p-value on actual and predicted values

Optimizer Comparison	t-statistic	p-value
Adam vs RMSprop	1.13	0.341
Adam vs SGD	-1.20	0.317
Adam vs Adagrad	-0.85	0.457

The table in Table 3 summarizes the performance of four distinct optimizers (Adam, RMSprop, SGD, and Adagrad) across four companies (META, TESLA, APPLE, and AMAZON) on both training and testing data and tested using MAE, MSE, and RMSE metrics. Adam puts out balanced results with consistent performance and relatively low errors on all axes. Error is moderate for META and TESLA, and APPLE and AMAZON show exceptionally low error values in training but increase errors in testing for META and AMAZON, which is indicative of slight over fitting. RMSprop performs well, with META and APPLE showing the best performance (lower errors in both training and testing). TESLA experiences higher errors, especially in testing. AMAZON also maintains relatively low training errors but increases during testing, suggesting model variability.

SGD has the worst performance in general with significant high errors in all companies, but most especially in META and TESLA (high MSE and RMSE). Although APPLE and AMAZON performed slightly better, the overall trend showed poor convergence and underperformance compared to other optimizers. Adagrad performs inconsistently. Training errors are low for META and AMAZON, but testing errors spike significantly (e.g., META's MSE of 175.491). TESLA and APPLE also show poor generalization, with errors rising during testing. Table 4 provides the following interpretation as stated below:

- Adam vs RMSprop: The p-value (0.341) indicates no statistically significant difference in MAE performance.
- Adam vs SGD: The p-value (0.317) suggests no significant difference in performance.

- Adam vs Adagrad: The p-value (0.457) confirms no significant difference between the two optimizers.

Among the optimizers, Adam performs the best overall, offering a balance between training and testing errors. RMSprop follows closely, performing well for APPLE and META but struggling with TESLA.SGD has the highest error values, thus being the least effective optimizer in this case. Adagrad performs well during training but generalizes poorly on test data, and the error increases by a large amount. In summary, Adam is the most reliable optimizer, followed by RMSprop, and SGD is the weakest for this dataset.

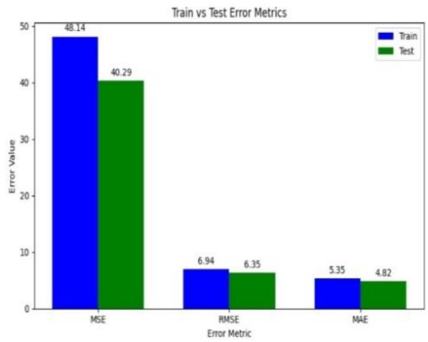


Fig.11. Error evaluation for closing price prediction of Tesla (2023-2024)

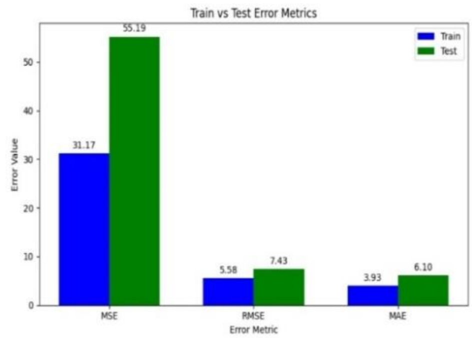


Fig.12. Error evaluation for closing price prediction of META (2023-2024)

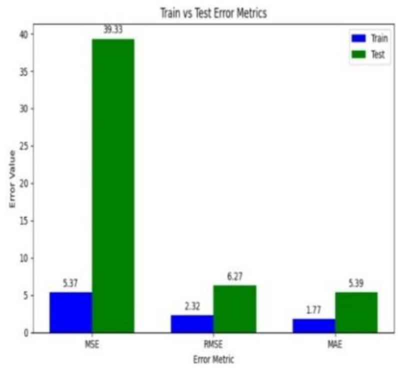


Fig.13. Error evaluation for closing price prediction of Apple (2023-2024)

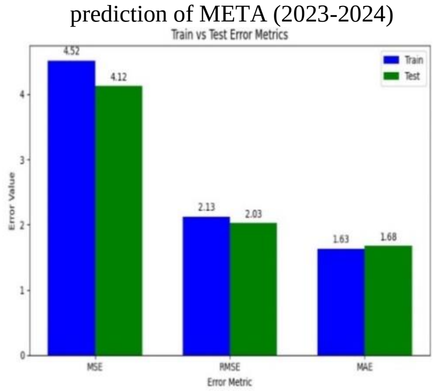


Fig.14. Error evaluation for closing price prediction of Amazon (2023-2024)

The grouped bar chart(see Fig.11-14) generated above is a is a comparison of MAE , MSE , RMSE , of both training(80%) and testing(20%) dataset with blue bars illustrating the errors for the training data and green bars representing the errors for the testing data. Each bar displays the value of the corresponding error metric, enabling a clear visual comparison of the model's performance on both the training and testing sets. The reliability of the model on the training and test data can be easily compared to the x-axis presentation of the three error metrics and the y-axis' measurement of the error values. Fig15-18 Line Graph evaluates the loss graph. The progression from Fig.15 to Fig 18 presents a clear picture on the performance of the model improving, with errors becoming more balanced between training and testing. This indicates better generalization as over fitting decreases.

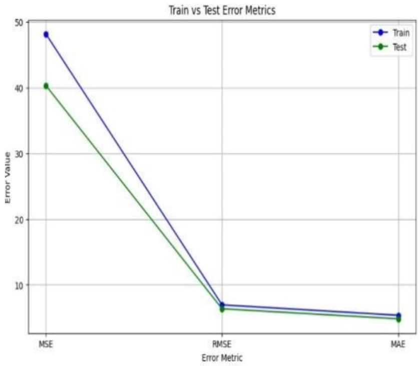


Fig.15. Error in Tesla closing price prediction

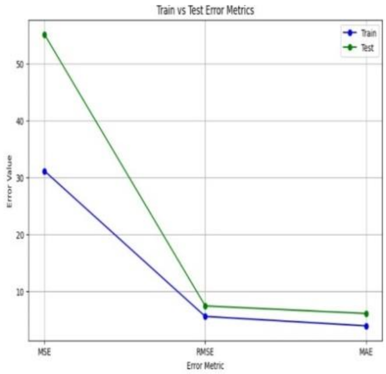


Fig.16. Error in Meta closing price prediction

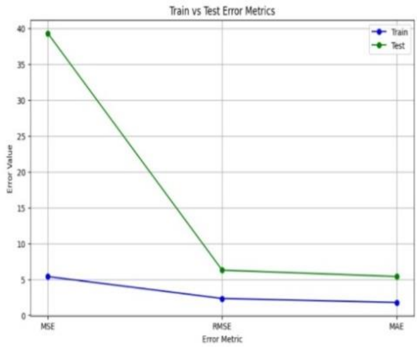


Fig.17. Error in Apple closing price prediction

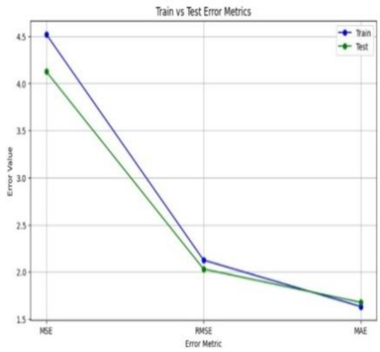


Fig.18. Error in Amazon closing price prediction

From the table Table 3, LSTM shows a significant improvement over SVR with lower MAE (84.701), RMSE (111.786), and MAPE (7.135%).The computational cost is higher than SVR, which takes 13 minutes to run .Bidirectional LSTM (BiLSTM) achieves the best results on all error metrics and also presents a minimum MAE (66.258), RMSE (98.854), and MAPE (5.117%).The model possesses the best prediction capabilities yet has slightly more computation expenses than the LSTM with the time consumed equal to 14 minutes. In terms of accuracy, BiLSTM proves to present the highest accurate prediction result, then the LSTM algorithm, while SVR performed poorest.SVR is the fastest but sacrifices accuracy, whereas BiLSTM achieves the best accuracy at the cost of longer computing time.Between computational time and accuracy, BiLSTM emerged as the most effective model for stock price forecasting.

4 Conclusion

This paper presents the successful application of Bi-Long Short-Term Memory (LSTM) networks with different optimizers on stock closing price prediction of META, Amazon, Apple, and Tesla. With historical stock data, it captures the temporal patterns, which make this LSTM model a good tool for future price forecast. Evaluation metrics such as MSE, RMSE, and MAE showed that this model was performing reasonably both for the training and testing set in terms of accuracy about which information can be derived from such models. The Adam optimizer turns to be the most effective here that is because of its properties related to adaptive learning rate and momentum, and, particularly, its ability to accommodate noisy, nonstationary data in finance. Despite the promising results of the study, it highlights many inherent challenges in financial forecasting, such as volatility and external factors that can influence the stock prices. These complexities can affect the model's predictive accuracy, further emphasizing the need for continued refinement. Future improvements would involve incorporating additional features like trading volume, sentiment analysis, and economic indicators in order to enhance the robustness of the model. This study affirms that, despite the challenges ahead, deep learning models, specifically LSTMs, are well on their way to changing the financial forecasting world. As long as these models have something meaningful to say about stock price trends, investors and analysts would benefit from using these models as an aid in decision-making processes.

References

1. Billah, M. M., Sultana, A., Bhuiyan, F., & Kaosar, M. G.: Stock price prediction: comparison of different moving average techniques using deep learning model. Neural Computing and Applications, 36(11), 5861-5871(2024).
2. Singh, S., Joshi, K., & Kanna, D.: Stock market price prediction analysis (LSTM vs EMA). In AIP Conference Proceedings (Vol. 3075, No. 1). AIP Publishing, (2024, July).
3. Daniswara, D. A., Widjanarko, H., & Hikmah, K.: The Accuracy Test of Technical Analysis of Moving Average, Bollinger Bands, and Relative Strength Index on Stock Prices of Companies Listed in Index Lq45. Indikator, 6(2), 411842(2022).
4. Garraux, H.: The Measure of Potential Profitability in Intraday Trading Utilizing the Relative Strength Index (RSI) with Various Simple Moving Averages (SMA) on the S&P 500(2024).
5. Karim, R., Alam, M. K., & Hossain, M. R.: Stock market analysis using linear regression and decision tree regression. In 2021 1st International Conference on Emerging Smart Technologies and Applications (eSmarTA) (pp. 1-6). IEEE. (2021, August).
6. Hartanto, A. D., Kholik, Y. N., & Pristyanto, Y.: Stock Price Time Series Data Forecasting Using the Light Gradient Boosting Machine (LightGBM) Model. JOIV: International Journal on Informatics Visualization, 7(4), 2270-2279(2023).
7. Kavinnilaa, J., Hemalatha, E., Jacob, M. S., & Dhanalakshmi, R.: Stock price prediction based on LSTM deep learning model. In 2021 International Conference on System, Computation, Automation and Networking (ICSCAN) (pp. 1-4). IEEE(2021, July)..
8. Khanderwal, S., & Mohanty, D.: Stock price prediction using ARIMA model. International Journal of Marketing & Human Resource Research, 2(2), 98-107(2021).

9. Yang, H., Liu, X. Y., Zhong, S., & Walid, A.: Deep reinforcement learning for automated stock trading: An ensemble strategy. In Proceedings of the first ACM international conference on AI in finance (pp. 1-8) (2020, October)..
10. Patel, R., Choudhary, V., Saxena, D., & Singh, A. K.: Lstm and nlp based forecasting model for stock market analysis. In 2021 First International Conference on Advances in Computing and Future Communication Technologies (ICACFCT) (pp. 52-57). IEEE(2021, December).
11. Jabin, S.: Stock market prediction using feedforward artificial neural network. International Journal of Computer Applications, 99(9), 4-8(2014).
12. Gupta, S., & Wang, L. P.: Stock forecasting with feedforward neural networks and gradual data sub-sampling. Australian Journal of Intelligent Information Processing Systems, 11(4), 14-17(2010)..
13. Rubi, M. A., Chowdhury, S., Rahman, A. A. A., Meero, A., Zayed, N. M., & Islam, K. A.: Fitting multi-layer feed forward neural network and autoregressive integrated moving average for Dhaka Stock Exchange price predicting. Emerging Science Journal, 6(5), 1046-1061(2022).
14. Gu, Y., Shibukawa, T., Kondo, Y., Nagao, S., & Kamijo, S.: Prediction of stock performance using deep neural networks. Applied Sciences, 10(22), 8142(2020).
15. Srivinay, Manujakshi, B. C., Kabadi, M. G., & Naik, N.: A hybrid stock price prediction model based on PRE and deep neural network. Data, 7(5), 51(2022).
16. Alshammari, T.T. et al.: Forecasting Stock Volatility Using Wavelet-based Exponential Generalized Autoregressive Conditional Heteroscedasticity Methods. Intelligent Automation Soft Computing. 35, 2589–601 (2023).
17. Toochaei, M. R., & Moeini, F.: Evaluating the performance of ensemble classifiers in stock returns prediction using effective features. Expert Systems with Applications, 213, 119186 (2023).
18. Yong, B. X., Abdul Rahim, M. R., & Abdullah, A. S.: A stock market trading system using deep neural network. In Modeling, Design and Simulation of Systems: 17th Asia Simulation Conference, AsiaSim 2017, Melaka, Malaysia, August 27–29, 2017, Proceedings, Part I 17 (pp. 356-364). Springer Singapore(2017)..
19. Jahan, I., & Sajal, S.: Stock price prediction using recurrent neural network (RNN) algorithm on time-series data. In 2018 Midwest instruction and computing symposium. Duluth, Minnesota, USA: MSRP(2018).
20. Thong, L., Jeenanunta, C., & Chaysiri, R.: Stock price prediction with long short-term memory recurrent neural network (Doctoral dissertation, Thammasat University) (2018).
21. Kamijo, K. I., & Tanigawa, T.: Stock price pattern recognition-a recurrent neural network approach. In 1990 IJCNN international joint conference on neural networks (pp. 215-221). IEEE (1990, June).