Vol. 08, No. 1 (2026) 295-304, doi: 10.24874/PES08.01A.002

Proceedings on Engineering Sciences

www.pesjournal.net

MACHINE LEARNING – BASED TARGET AND STOP-LOSS PREDICTION FOR INTRADAY PROFIT MAXIMIZATION AFTER TAXES

Raj Gupta
Kaja Mohideen A[1]

Received 14.04.2025
Revised 09.06.2025
Accepted 15.07.2025

Keywords:

Stock Market, Intraday Profit, Long-Short Term Memory (LSTM), Generative Adversarial Network (GAN), Gated Recurrent Unit (GRU)

A B S T R A C T

The stock market facilitates public share offers, which is essential for company funding. Aiming for profitability, trading involves a diverse range of players, including institutional investors (FIIs and DIIs) and individual traders. For retail traders, effective risk management is essential. This study uses dynamic stop-loss and target value adjustments to mitigate retail trading risk. We explore the models such as GAN, bidirectional LSTM, stacked LSTM, and GRU. A CNN-based discriminator and a GRU-based generator are characteristics of the GAN. We examine the NMDC stock and identify relevant factors using data from Yahoo Finance. Totally 50 features are produced by applying ARIMA and feature selection analysis on the dataset. Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) are used to evaluate the performance of the model. Through sophisticated modelling techniques and extensive datasets, this research offers methods to optimize decision-making and enhance trading outcomes in dynamic markets, therefore shedding light on effective risk management for retail traders.

© 2026 Published by Faculty of Engineering

1. INTRODUCTION

The stock exchange functions as a comprehensive platform for corporations to raise funds by issuing their shares to the public. This diverse market attracts a wide spectrum of players, including individual retail traders and international and local institutional investors. However, mastering the nuances of stock market financial research is a daunting task in today's investment scene. Scholars have worked hard to develop predictive systems that incorporate both technical and statistical research to identify profitable trading opportunities.

The distinction between fundamental and technical analysis forms the foundation of stock market forecasting. Fundamental analysis examines stock prices from the intrinsic value perspective to determine a security's fair value. On the other hand, technical analysis uses past price data and chart patterns to predict future changes. As predictive models have developed over time, linear frameworks have become widely used stock market forecasting techniques.

In order to increase retail traders' daily earnings after taxes, this article conducts a focused investigation into the prediction of target and stop-loss levels for shares. By

[1] Corresponding author: Kaja Mohideen A
Email: kajamohideen.a@vit.ac.in

utilizing sophisticated predictive models like Autoregressive Integrated Moving Average (ARIMA), Gated Recurrent Unit (GRU), Generative Adversarial Network (GAN), and Long Short-Term Memory (LSTM), this research aims to maximize trading strategies by matching the current trends of particular shares. Through extensive research and a deep dive into the complexities of model design, this paper aims to further enhance predictive techniques in the field of stock market trading.

This paper is arranged as follows in the sections that follow: The Problem Statement is outlined in Section 2, and the prior works are reviewed in Section 3. Section 4 discusses and analyses the model designs, dataset, and feature engineering done, which is then followed with a result analysis, conclusion, and future work in the Sections 5, 6, and 7 respectively.

2. PROBLEM STATEMENT

Despite the number of materials and analytical tools available for stock market investment, retail traders frequently struggle to maximize earnings because of the fundamental complexity and unpredictability of financial markets. The challenge is not just properly predicting market prices but also developing successful trading techniques to benefit from these predictions while limiting risks. Additionally, despite the availability of many prediction models and methodologies, there remains a deficiency in the literature about the optimization of target and stop-loss levels for specific stocks. A significant portion of market participants are retail traders, who typically rely on instinct or simple strategies to guide their trading decisions since they lack robust frameworks. This can lead to worse-than-ideal outcomes. As a result, the fundamental issue addressed in this paper is the need for a comprehensive and the best model to projecting target and stop-loss levels for stocks, with the ultimate goal of increasing retail traders' profitability. This study intends to address a vacuum in the literature by employing advanced predictive algorithms and investigating the relationship between fundamental and technical analysis. It will also provide practical tips for retail traders to better navigate the stock market.

3. LITERATURE REVIEW

Althelaya et al. (2018) evaluated and compared the performance of different deep-learning architectures for financial time series prediction, specifically bidirectional and stacked LSTM models. The author collected the S&P500 index stock price from 01/1/2010 to 30/11/2017 from Yahoo Finance. The authors concluded that they found bidirectional LSTM models outperformed stacked LSTM models. Shallow neural networks, and simple form of LSTM network have performed better in terms of prediction accuracy for short-term prediction. However, for long-term prediction stacked LSTM performed better than bidirectional LSTM models. One limitation of this study is that it only used one dataset (S&P500) for evaluation, which may not be representative of other financial time series data. Furthermore, a crucial factor in practical applications, the interpretability of the Deep Learning (DL) models, was not investigated by the authors. Vezeris et al., (2018) compared multiple ways to make profit and stop-loss Trading Strategies. Six strategies were explored to take profit and stop-loss which include Simple take profit, Dynamic take profit, faster take profit, SSL (Simple Stop-Loss), Sliding and Variable ATR Window (SVASL) are combined with Moving Average Convergence Divergence (MACD). Trading system which is widely used in the financial Markets. This research was conducted on specific assets from the Forex, Metals, Energy, and Cryptocurrency categories. The author concluded that a faster take profit strategy is not superior to a simple MACD strategy. The SVASL strategy outperforms the other stop-loss strategies. These strategies are effective in generating profits and reducing the risk of losses, but effectiveness depends on various factors such as the testing period and asset selected for trading.

Zhang (2019) investigates the use of a novel GAN architecture, which combines Multi-Layer Perceptron (MLP) and LSTM, to estimate stock values. The authors show encouraging results in predicting daily closing prices with S&P500 Index data, beating established methods such as Support Vector Regression (SVR) and Artificial Neural Network (ANN). However, the study admits limitations such as the complexity and unpredictability of financial markets, recommending additional validation under various market situations. Overall, the research emphasizes the potential of deep learning approaches, particularly GAN, to improve stock market analysis and decision-making, opening the way for future advances in financial forecasting. Zhou (2018) studies the Generative Adversarial Network for stock market price prediction, using a combination of LSTM and Convolutional Neural Network (CNN) architecture. The methodology section is transparent about data sources, model architecture, training techniques, and evaluation metrics, which improves the study's trustworthiness. While the limits are not expressly addressed, their examination would provide useful insights into potential biases or restraints. The conclusion successfully summarizes major findings, discusses ramifications, and proposes future research directions, bringing the study to a close and underlining its contributions to stock market prediction with GANs. Overall, the research provides useful information about using advanced neural network architectures for financial forecasting.

To ascertain the ideal time frame for projecting future share values, Ghosh et al. (2019) looked at the expansion of businesses across a range of industries in 2019. The author collected data, cleaned the data, selected the

features, and trained the neural network. The author concluded that their LSTM-based model can accurately predict stock prices on long-term historical data. The Author suggests that the model's accuracy can be improved by training it with more data sets and conducting specific business analyses. There were several limitations in their study, including the fact that their model is based on historical data and may not accurately predict future trends and that there may be other factors that influence stock prices that are not captured. Idrees et al. (2019) presented a method for forecasting stock market volatility in 2019 based on time series data. An ANN and the ARIMA model were combined. The S&P500 index was used to assess the model's accuracy and effectiveness. The mean absolute error and the root mean squared error were the two primary indicators of performance that the author computed to assess the performance as well as accuracy. The suggested model performs better in terms of prediction accuracy than the GARCH and ARIMA models, according to the author's conclusion.

Lee (2019) created the ground-breaking Chaotic Oscillatory Multi-agent-based Neuro Computing System (COSMOS) for intelligent trading and global financial forecasting. The suggested method is integrated with the Quantum Finance Forecast Centre to provide projections and sophisticated trading platforms for 129 financial products globally. Chaotic neural oscillator technology is successfully integrated into two components by COSMOS: (i) COSMOS Trader: Random RBF-based Actor-Critic Reinforcement-learning agents for trading strategy optimization; (ii) COSMOS Forecaster: a chaotic supervised learning agent based on FFBP for global financial forecasting. The author concluded that COSMOS is a cutting-edge and successful financial trading and prediction system that effectively handles the issues of massive data over-training and deadlock that arises with RBF and traditional recurrent neural networks. COSMOS is a dependable and efficient financial investment tool for traders and investors because it offers a quick reinforcement learning and forecasting solution. This paper's shortcoming is that it doesn't offer a thorough analysis of COSMOS's performance in actual financial markets.

Mehtab & Sen (2019) suggested a hybrid strategy that improves on current approaches by combining Machine Learning (ML), deep learning, and Natural Language Processing (NLP) to forecast stock prices more accurately. The authors also discussed the limitations of the existing approach. The author collected daily data on NIFTY 50 from 2015 to June 2019, the author used multiple Machine learning and deep learning techniques to predict the stock market. These techniques include a classification-based model, regression-based model, and sentiment analysis model. The author concluded that this hybrid strategy could accurately forecast stock prices. They also suggest that this type of model can have potential implications for the financial industries. A prediction model that employs LSTM and Recurrent Neural Network (RNN) for portfolio management and stock market price prediction was proposed by Pawar et al. (2019). They contrasted the conventional trading machine learning methods with the LSTM RNN model. The LSTM RNN model outperformed traditional machine learning algorithms, according to the authors' conclusion. The difficulties encountered in gathering and preparing the data were also noted by the authors.

Selvamuthu et al. (2019) studied the performance of three algorithms: Scaled Conjugate Gradient, Levenberg-Marquardt, and Bayesian Regularization, and explained that they were compared in terms of accuracy in predicting stock rates based on an Indian company's 15-minute tick data. Further, the author talks about how it can help investors and reduce risk. This study used an Indian company's tick data and 15-minute data for 30 business days. This 15-minute data encompassed price and volume, and the tick data consisted of OHLC (Open, High, Low, Close). Three data sets are created from this data: training, validation, and testing. The data was divided into 70%, 15%, and 15% respectively. The performance was evaluated using two metrics: MSE: Which measures the discrepancy between the asking price and the real one. CC: Calculates how strongly the expected and actual prices follow a linear relationship. The author concluded that the Bayesian Regularization was the most accurate among the three algorithms.

Mehtab et al. (2020) suggested a hybrid strategy that combines deep learning and machine learning techniques to forecast Indian stock market prices. This study trains and tests several models, including CNN based deep learning regression models, machine learning-based regression models, and classification-based models, employing daily stock prices from the NIFTY 50. The Author collects NIFTY 50 index daily stock data from 2015 to 2018. Several models are being trained and tested using this data, which was gathered from the Yahoo Finance website. The author evaluates the performance using some metrics like recall, specification, precision, negative predictive value, and classification accuracy. The author states some limitations, this approach might not apply to other indexes or financial instruments, Additionally, the author acknowledged that a sudden drop or hike in price cannot be accurately predicted.

Jin et al. (2020) a deep learning-based stock market forecast model that takes the emotional inclinations of investors into account. The authors contrasted the traditional LSTM model with the developed LSTM model that included the attention mechanism and sentiment index. The authors compared the evolved LSTM model with the attention mechanism and sentiment index with the traditional LSTM model. The result showed that the proposed approach can improve the prediction accuracy of the stock closing prices than the traditional LSTM model and other benchmark models. The authors also accepted one limitation of this

study was that only the Chinese stock market was considered. Fister et al. (2021) compared the effectiveness of the traditional trading strategies and two LSTM models which were the "Common Model" and the "Unique Model". The author collected 10 different stock price data over 10 years. The stocks were chosen due to their market capitalization and liquidity finally, the author concluded that the common model and unique model outperformed the traditional trading strategies, but the unique model outperformed the standard model by a small margin.

A new framework for intra-day stock market prediction was presented by Kumar & Haider (2021). It blends metaheuristic optimization with the RNN-LSTM network. The authors want to show that, in terms of assessment metrics and accuracy, their suggested framework performs better than current methods. According to the author, the RNN-LSTM model performs better than the current methodology in terms of assessment metrics and accuracy. The author is aware of some of the drawbacks, including the substantial data set and computing power needed for prediction. Jiang (2021) gave a thorough summary of recent advancements in the use of deep learning methods for stock market prediction in 2021. The author systematically reviewed relevant review papers from 2017 to 2019 with a focus on deep learning models and their implementation. The author concluded that there are potential benefits of utilizing deep learning models to forecast the stock market which include higher accuracy and better generalization. the authors note that their survey is limited to studies published between 2017 and 2019 and may not include all relevant research in this field. They also point out that the reproducibility of some studies may be challenging due to differences in data sources, preprocessing techniques, and model architecture.

Bhandari et al. (2022) proposed a reliable predictive model for the stock market index using LSTM and other machine learning algorithms. The author used a combination of LSTM and other machine-learning techniques to develop their predictive model. These models performed reasonably well compared to traditional models if a good combination of features is considered. The author acknowledged some limitations. They only considered a limited set of variables and did not explore the impact of external factors such as political events or natural disasters. Ghosh et al. (2022) used random forest and LSTM models to forecast industry developments. S&P500 stocks covering initial prices and adjusted closing prices from 1990 to December 2018 were used in this analysis. The data set was split into an in-sample and an out-of-sample portion. Ten equities with the highest likelihood were brought and ten stocks with the lowest probability were sold on each trading day. Before taxes, the authors found that random forest produced a daily return of 0.54% while LSTM produced a daily return of 0.64%.

Wu (2022) describes a unique approach for improving trading strategy performance that combines a Generative Adversarial Network (GAN) and a Piecewise Linear Representation (PLR). It uses supervised learning to address the complexity of trading methods and overfitting concerns, to produce different trading action sequences for better results. The methods section gives a thorough description of the framework, covering data preprocessing, model training, and performance evaluation measures. While no constraints are expressly indicated, potential obstacles such as model complexity and data requirements may have an impact on implementation. The conclusion is most likely a summary of major findings, emphasizing the GAN-based framework's superior performance with PLR and suggesting future research areas. Overall, the article provides great insights on using advanced methodologies to optimize trading tactics.

4. Experimental Setup

LSTM (Long Short-Term Memory)
Recurrent Neural Network (RNN) architecture in the form of Long Short-Term Memory (LSTM) networks solves the vanishing gradient issue that ordinary RNNs have. The long-term dependencies within sequential data may be retained in LSTMs, which employ gated units to control information flow.

Common Architecture:
LSTM networks are made up of specialized memory cells, each with a cell state and three gating mechanisms: input, forget, and output gates. These gates, controlled by sigmoid and hyperbolic tangent activation functions, control the information that enters, leaves, and travels through the cell.

The Input Gate (I): determines how much fresh information should be absorbed into the cell state.

Forget Gate (f): Allows the network to discard unnecessary prior data by controlling the retention or removal of information from the cell state.

Output Gate (o): Regulates how much the LSTM unit's output is impacted by the current cell state.

Implemented architecture

- The Model Architecture is defined using Keras Sequential API.
- Two LSTM layers are stacked successively, each having 64 units.
- The first LSTM layer returns sequences (i.e., outputs at each time step), but the second simply delivers the final output.
- Both LSTM layers employ L2 regularization with a strength of 0.01.
- The definition of the input shape is (input dim, feature size), with input dim representing the length of the input sequences and feature size representing the number of characteristics for each time step.
- A Leaky ReLU activation function and a Dense layer with a single unit make up the output layer.

GRU (Gated Recurrent Units)

Recurrent Neural Network (RNN) architecture in the form of Gated Recurrent Units (GRUs) seeks to address the shortfalls of standard RNNs in identifying long-term relationships in sequential input. GRU uses gating techniques to control the flow of information through the network, allowing for enhanced gradient flow and minimizing the vanishing gradient problem.

Common architecture

Gated structures, such as an update and reset gate, are present in GRUs. By selectively updating the hidden state in response to input and the previous hidden state, these gates control the flow of data throughout the network.

Update Gate (z): Controls the extent to which the prior hidden state is integrated with the candidate activation, hence managing the amount of information kept from the past.

Reset Gate (r): Regulates the amount that the prior hidden state influences the candidate activation calculation, enabling the network to adaptively discard unnecessary data.

Implemented architecture

- The model is defined using Keras Sequential API.
- Two GRU layers are consecutively added to the model.
- The first GRU layer has 64 units and is configured to return sequences (outputs at each time step).
- The input shape is defined as (input dim, feature size), with input dim representing the length of the input sequences and feature size denoting the number of features per time step.
- The first GRU layer's kernel weights undergo L2 regularization with a strength of 0.01.
- The second GRU layer contains 32 units and does not return sequences, implying that it just produces the final prediction.

GAN (Generative Adversarial Network)

Generative Adversarial Networks uses two neural networks, the discriminator and the generator, compete in a minimax game to form GANs. Whereas the discriminator looks for differences between actual and fake samples, the generator aims to produce realistic data samples.

Common Architecture

- **Generator network:**

a. The generator network uses random noise vectors as input to generate synthetic data samples.
b. It is often composed of numerous layers of neural networks, which are frequently implemented using convolutional or deconvolutional architectures in the case of picture-generating applications.
c. The generator's goal is to produce data samples that are indistinguishable from real ones.

- **Discriminator network:**

a. The discriminator network has been programmed to differentiate between actual training set samples of data and counterfeit ones produced by the generator.
b. It also has numerous layers of neural networks to correctly identify authentic and fraudulent samples.
c. The discriminator is trained alongside the generator and offers input on the realism of the generated samples.

- **Training process:**

a. During training, the generator and discriminator networks are iteratively trained using a minimax game.
b. The generator's goal is to minimize the objective function, tricking the discriminator into identifying its generated samples as real.
c. In contrast, the discriminator seeks to maximize the objective function by properly categorizing real samples as real and fraudulent samples as fake.
d. The training approach entails updating the weights of both networks using backpropagation and gradient descent techniques.

Implemented architecture

- **Generator model:**

o The generator is made up of a sequential stack of GRU layers.
o It starts with a GRU layer of 1024 units and returns sequences.
o Following that, a 512-unit GRU layer returns sequences as well.
o Finally, a GRU layer of 256 units is added, which does not return sequences.
o Following the GRU layers, two thick layers are added: one with 128 units and the other with 64 units.
o The final dense layer produces data with the specified output dimension.

- **Discriminator model:**

o The discriminator architecture commences with a collection of convolutional layers which have one dimension in size.
o Using a Leaky ReLU activation function, the first convolutional layer has 32 filters and a kernel size of three.
o The next step involves applying a convolutional layer with 64 filters and a kernel size of 5.
o An extra convolutional layer with 128 filters and a kernel size of 5 is part of the architecture.
o Following the convolutional layers, the feature maps are flattened, and two dense layers are used.
o The first dense layer uses a Leaky ReLU activation function and has 220 units.
o Next, 220 units of another thick layer with ReLU activation is placed.
o To complete the binary classification process, a single dense output layer with a sigmoid activation function is added.

Dataset Description

The dataset, consisting of 2447 rows and 50 columns, includes a wide range of assets sourced from Yahoo Finance. These assets, represented by correlated assets, currencies, index funds etc. Along with the asset data, the dataset includes a range of technical indicators calculated for each asset, including metrics such as Moving Averages (MA5, MA21, MA200), Relative Strength Index (RSI), Exponential Moving Averages (26EMA, 12EMA), Moving Average Convergence Divergence (MACD), standard deviation (20SD), Bollinger Bands (upper band, lower band), Exponential Moving Average (EMA), momentum, predicted close estimated from ARIMA model. Furthermore, the dataset includes the NMDC stock's open, high, and low prices, which provide further information about its market performance. With such detailed data, analysts and researchers may conduct in-depth financial analysis and create predictive models to help them make investment decisions. 30% of the dataset was put aside for testing, while the remaining 70% was used for training. This partitioning guarantees that models can be trained on a sufficient quantity of data and permits robust assessment on data that hasn't been seen before in order to gauge generalization performance.

Feature Selection

Using the dataset, an XGBoost model was trained, and each feature's relevance was assessed according to how well it contributed to the model's overall performance. XGBoost has a built-in feature significance metric that ranks features according to their impact on the model's predicted accuracy. After training the XGBoost model, the top 20 features with the greatest significance scores were chosen for additional analysis and model refinement. These characteristics were deemed the most informative and influential in predicting the target variable or outcome of interest.

5. RESULTS AND DISCUSSION

Error measures can be helpful for assessing the efficacy of machine learning models since they reveal how well the model's forecasts match the actual values. Here's a technical overview of a few common error metrics.

RMSE

- The average size of errors between predicted and actual values is measured by the Root Mean Squared Error (RMSE).
- The square root of the average of the squared differences between the predicted and actual values is used to calculate it.
- RMSE is sensitive to large errors and penalizes them more severely than smaller ones.

MAPE

- The average percentage difference between the expected and actual values is computed using Mean Average Percentage Error (MAPE).
- It is particularly helpful when comparing the amount of real data to the accuracy of projections.
- MAPE is computed by averaging the absolute percent errors and is shown as a percentage.

MSE

- The Mean Square Error (MSE) measures the difference between expected and actual values.
- It calculates the average squared error and is commonly used in regression situations.
- MSE is less interpretable than RMSE since it does not use the same units as the target variable.

MAE:

- The average absolute difference between the expected and actual values is determined by Mean Absolute Error (MAE).
- It determines the average error magnitude without taking direction into account. MAE is less sensitive to outliers than RMSE and MSE.

LSTM Performance

The stacked LSTM architecture consists of two LSTM layers: the first layer returns sequences, while the second layer aggregates these sequences to produce a single output. This output is then processed by a single unit in a dense layer, and the Leaky ReLU activation function to provide the final prediction. Meanwhile, the bidirectional LSTM design includes bidirectional LSTM layers, which enable the model to interpret input sequences in both forward and backward orientations. The first bidirectional LSTM layer generates sequences, whereas the second layer produces a single sequence. Additional dense layers with Leaky ReLU activation functions are added to help with feature extraction and final output prediction. Both designs are evaluated using the RMSE, MAE, MSE, and MAPE after being trained for 50 epochs with a certain batch size. Figure 1 shows the training behaviour of the stacked LSTM model using the NMDC intraday dataset. The training loss between epochs is decreasing steadily, which suggests a stable convergence and good learning of temporal price patterns, which proves that the model is able to learn sequential dependencies without overfitting a lot in the training process.

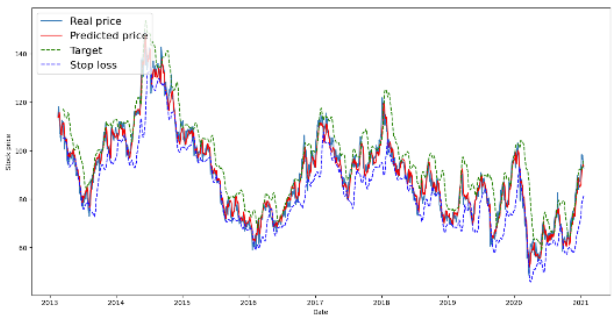


Figure 1. Training loss convergence of the stacked LSTM model

Figure 2 shows the test performances of the stacked LSTM model on unseen NMDC data. The findings indicate that there is a reasonable correspondence between the forecasted and the observed price dynamics, which proves the ability of the model to be generalized. Nevertheless, we have slight off course which means that there is room to work on capturing the intraday market variability.

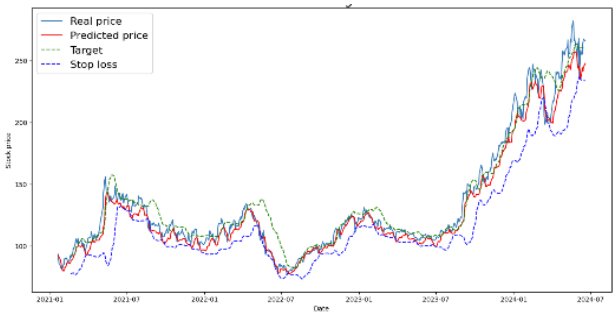


Figure 2. Testing Performance of the Stacked LSTM model

The training performance of the bidirectional LSTM model is presented in Figure 3, where the information is processed both forwards and backwards. This two-channel learning allows better extraction of features of historical price data with a faster rate of convergence and lower training error in comparison with the single stacked LSTM architecture. The testing results regarding the NMDC dataset on the bidirectional LSTM model are presented in Figure 4. The high similarity between the predicted price and the actual price shows that the generalization performance is better attained, and that bidirectional temporal learning is a better model to predict intraday stock price changes more precisely.

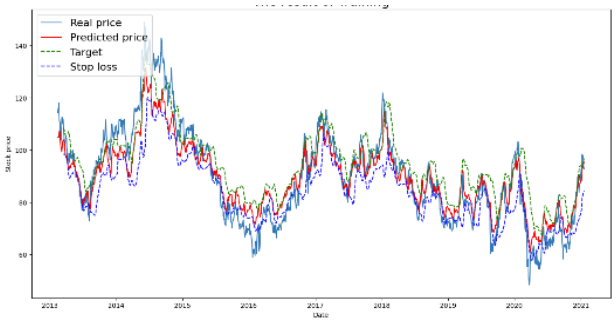


Figure 3. Training Performance of the Bi-directional LSTM model

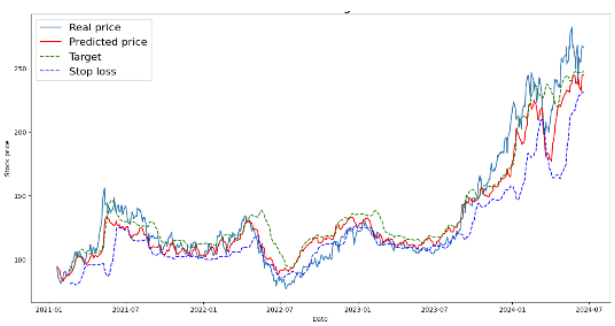


Figure 4. Testing Performance of the Bi-directional LSTM model

GRU performance

With information from the NMDC dataset, the designed architecture employs a GRU (Gated Recurrent Unit) neural network for time series prediction. GRU networks can capture long-term relationships without experiencing the vanishing gradient problem, which makes them useful for modelling sequential data. In this architecture, the model receives input data from the NMDC dataset, with the initial GRU layer set to output sequences, allowing it to capture temporal patterns across numerous time steps. Following that, another GRU layer combines data from the full input sequence to create a final prediction. Two dense layers follow, helping with feature extraction and prediction refinement. Training progress is tracked while the model is trained over 50 epochs with the Adam optimizer and the mean squared error (MSE) loss function. In Figure 5, it is shown how the training of the GRU model based on the NMDC dataset proceeds. The flattened loss of training indicates effective learning through simplified gating operation that allows the model to induce important temporal reliance at lower computational complexity than standard LSTM designs. Figure 6 shows how the GRU model performs on the unseen intraday data. The findings show a reasonable predictive accuracy, and the variations are moderate when the markets are volatile. This proves the fact that within dynamic market environments, the GRU model is capable of balancing the efficiency of learning and predictive performance.

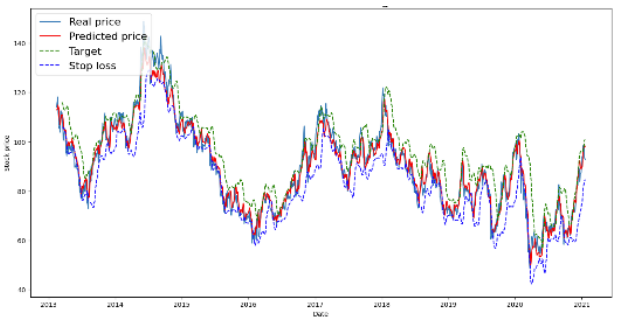


Figure 5. Training results of the GRU model

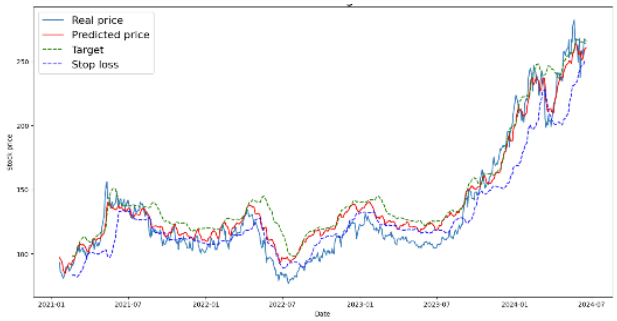


Figure 6. Testing results of the GRU model

GAN Performance

The developed GAN architecture uses adversarial training to generate synthetic data samples that closely mimic actual data from the NMDC dataset. The two main components of the architecture are the generator and the discriminator. The generator seeks to generate realistic data samples while the discriminator gains the ability to discern between actual and false data. Iterative training increases the generator's ability to create data that is indistinguishable from genuine samples while also enhancing the discriminator's ability to recognize synthetic data. The adversarial architecture enhances the quality and diversity of the dataset by enabling the GAN to generate realistic data samples. The testing and training of the GAN are based on the NMDC dataset, which makes it possible to generate synthetic data samples. As shown in Figure 7, the GAN-based model has performance capabilities of producing real-looking synthetic data that has a close correlation with the real NMDC price trends. The training process of adversarial training leads to the increase in the diversity of data and its strength, which helps to increase the reliability of prediction in different intraday market conditions.

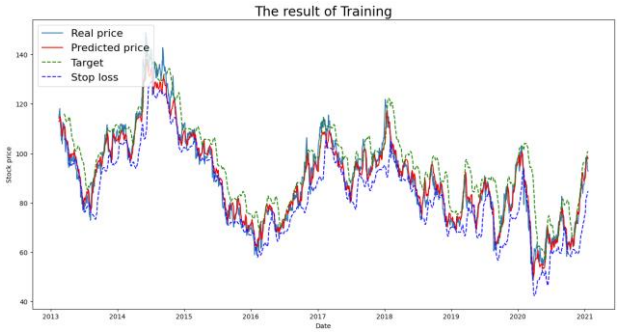


Figure 7. Prediction performance results of GAN model

Table 1 shows a comparison of the standard error of training performance of LSTM, bidirectional LSTM, GRU, and GAN. The highest values of RMSE, MAE, and MSE are obtained with the bidirectional LSTM, which suggests the best learning ability and is able to fit best to historical intraday stock price behavior.

Table 1. The error metric analysis on the training data set

Model	*LSTM (BASIC)*	*BI-LSTM*	*GRU*	*GAN*
RMSE	*6.44*	*3.561*	*5.053*	*4.269*
MAE	*5.09*	*2.686*	*4.221*	*3.390*
MSE	*41.52*	*12.68*	*25.53*	*18.22*
MAPE	*5.82*	*2.994*	*4.996*	*4.052*

Table 2 gives a summary of the testing performance of the evaluated models on the unseen data. Bidirectional LSTM has shown superiority over the simple LSTM and GRU models in all error measures, which validates its better generalization power and strength in the intraday target and stop-loss forecasting tasks.

Table 2. The error metric analysis on test data set

Model	*LSTM (BASIC)*	*LSTM (BI)*	*GRU*	*GAN*
RMSE	*6.44*	*4.123*	*5.655*	*N/A*
MAE	*5.09*	*3.72*	*4.031*	*N/A*
MSE	*41.52*	*20.87*	*31.98*	*N/A*
MAPE	*5.82*	*2.781*	*3.133*	*N/A*

6. CONCLUSION

To summarize, while the bidirectional LSTM model surpasses the GAN and GRU models in terms of generating more accurate closure price predictions for the NMDC dataset, it is crucial to highlight that the performance of each model varies based on the individual dataset characteristics. While the bidirectional LSTM performed better in this case, it is worth noting that there may be circumstances in which the GAN exceeds the LSTM, or where the LSTM outperforms both the GAN and the GRU. As a result, it is critical to analyse each dataset's unique properties before selecting the suitable model. Furthermore, while predicting price movements, a target is set at 2 standard deviations above the mean and a stop-loss at 1.5 standard deviations below the mean. This technique maintains a balanced strategy for chasing profitable opportunities while avoiding prospective losses, hence improving the predictive model's effectiveness in financial forecasting scenarios.

Acknowledgement: We would like to sincerely thank our colleagues for their thoughtful criticism and constant support during the project. We would also want to express our gratitude to our mentors and advisers for their crucial advice that helped us stay on course. We also thank the management of VIT, Chennai for their supporting role and the technical personnel for their help.

7. FUTURE WORK

Future work could include automating the feature engineering process and discovering the most correlated assets. Automated feature engineering strategies, such as feature selection algorithms and automatic feature creation methods, could be used to speed up the feature engineering process and increase the model's prediction performance. Furthermore, advanced machine learning and data mining techniques could be used to automatically identify the assets with the highest correlation to the target variable, eliminating the need for manual selection and potentially uncovering hidden relationships that contribute to better forecasting accuracy. Future research that incorporates these automated methodologies into the modelling pipeline can improve the efficiency and effectiveness of financial forecasting models, resulting in better informed investment decisions.

References:

Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018, April). Evaluation of bidirectional LSTM for short-and long-term stock market prediction. *In 2018 9th International Conference on information and communication systems (ICICS) (pp. 151-156). IEEE.*

Bhandari, H. N., Rimal, B., Pokhrel, N. R., Rimal, R., Dahal, K. R., & Khatri, R. K. (2022). Predicting stock market index using LSTM. *Machine Learning with Applications, 9, 100320.*

Fister, D., Perc, M., & Jagrič, T. (2021). Two robust long short-term memory frameworks for trading stocks. *Applied Intelligence, 51*(10), *7177-7195*

Ghosh, A., Bose, S., Maji, G., Debnath, N., & Sen, S. (2019, September). Stock price prediction using LSTM on Indian Share Market. *In Proceedings of the 32nd International Conference (Vol. 63, pp. 101-110).*

Ghosh, P., Neufeld, A., & Sahoo, J. K. (2022). Forecasting directional movements of stock prices for intraday trading using LSTM and random forests. *Finance Research Letters, 46, 102280.*

Idrees, S. M., Alam, M. A., & Agarwal, P. (2019). A prediction approach for stock market volatility based on time series data. *IEEE Access, 7, 17287-17298.*

Jiang, W. (2021). Applications of deep learning in stock market prediction: Recent progress. *Expert Systems with Applications, 184, 115537.*

Jin, Z., Yang, Y., & Liu, Y. (2020). Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications, 32, 9713-9729.*

Kumar, K., & Haider, M. T. U. (2021). Enhanced prediction of intra-day stock market using metaheuristic optimization on RNN–LSTM network. *New Generation Computing, 39, 231-272.*

Lee, R. S. (2019). COSMOS trader–Chaotic Neuro-oscillatory multiagent financial prediction and trading system. *The Journal of Finance and Data Science, 5*(2), 61-82.

Mehtab, S., & Sen, J. (2019). A robust predictive model for stock price prediction using deep learning and natural language processing. *arXiv preprint arXiv:1912.07700.*

Mehtab, S., Sen, J., & Dutta, A. (2020). Stock price prediction using machine learning and LSTM-based deep learning models. *In Machine Learning and Metaheuristics Algorithms, and Applications: Second Symposium, SoMMA 2020, Chennai, India, October 14–17, 2020, Revised Selected Papers 2 (pp. 88-106). Springer Singapore.*

Pawar, K., Jalem, R. S., & Tiwari, V. (2019). Stock market price prediction using LSTM RNN. *In Emerging Trends in Expert Applications and Security: Proceedings of ICETEAS 2018 (pp. 493-503). Springer Singapore.*

Selvamuthu, D., Kumar, V., & Mishra, A. (2019). Indian stock market prediction using artificial neural networks on tick data. *Financial Innovation, 5*(1), *1-12.*

Vezeris, D., Kyrgos, T., & Schinas, C. (2018). Take profit and stop loss trading strategies comparison in combination with an MACD trading system. *Journal of Risk and Financial Management, 11*(3), *56.*

Wu, J. L., Tang, X. R., & Hsu, C. H. (2022). A prediction model of stock market trading actions using generative adversarial network and piecewise linear representation approaches. *Soft Computing, 27*(12), *8209-8222.*

Zhang, K., Zhong, G., Dong, J., Wang, S., & Wang, Y. (2019). Stock market prediction based on generative adversarial network. *Procedia Computer Science, 147, 400-406.*

Zhou, X., Pan, Z., Hu, G., Tang, S., & Zhao, C. (2018). Stock market prediction on high-frequency data using generative adversarial nets. *Mathematical Problems in Engineering.*

Kaja Mohideen A
School of Computer Science and Engineering,
Vellore Institute Of Technology,
Chennai, India
kajamohideen.a@vit.ac.in
ORCID 0000-0003-1329-3925

Raj Gupta
School of Computer Science and Engineering,
Vellore Institute Of Technology,
Chennai, India
raj.gupta2023@vitstudent.ac.in
ORCID 0009-0002-8703-8265