Stock Market Prediction using Deep Learning Techniques

Project report submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Technology
in
Computer Science and Engineering

by

Utkarsh Singh Ashok - 21UCS221
Utkarsh Gupta - 21UCS222

Under Guidance of
Dr. Navneet Garg

Department of Computer Science and Engineering
The LNM Institute of Information Technology, Jaipur

November 2024

©The LNM INSTITUTE OF INFORMATION TECHNOLOGY Jaipur-2022.
All rights reserved.

The LNM Institute of Information Technology
Jaipur, India

CERTIFICATE

This is to certify that the project entitled "Stock Market Prediction Using Deep Learning Techniques" , submitted by Utkarsh Singh Ashok (21UCS221) and Utkarsh Gupta (21UCS222) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Computer Science and Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2024-2025 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my/our opinion, this report is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

Date: 22/11/24

Adviser: Dr. Navneet Garg

Acknowledgments

We would like to express our heartfelt gratitude to all those who contributed to the successful completion of this project on stock market prediction using deep learning techniques.

First and foremost, we would like to extend our deepest appreciation to our project guide, Dr. Navneet Garg, for their constant support, invaluable insights, and guidance throughout the course of this research. Their expertise in the field and willingness to help at every step played a crucial role in shaping the outcome of this project.

We would also like to thank our professors, for their encouragement and for providing us with the foundational knowledge necessary to undertake this project. Their teachings in machine learning, data science, and deep learning have been instrumental in understanding the core concepts behind stock market prediction.

Our sincere thanks to the technical staff and the library team for their assistance in providing the necessary resources, tools, and infrastructure to conduct our research. Without their support, we would not have been able to execute our project with the desired efficiency.

We also wish to acknowledge the open-source datasets, research papers, and online resources that made it possible for us to delve deeper into the realm of stock market prediction and deep learning techniques. Their availability greatly enhanced the scope and depth of our work.

Finally, we would like to express our gratitude to our family and friends for their patience, understanding, and encouragement, which provided us with the motivation to complete this project.

We are grateful for the opportunity to work on this project and hope that the results contribute meaningfully to the field of stock market prediction and the application of deep learning techniques.

Abstract

The stock market, characterized by its high volatility, has been a popular topic of interest in recent years, particularly as inflation rates have prompted individuals to explore investments in stocks, commodities, and other markets rather than traditional saving methods. This report delves into the application of advanced deep learning models for stock market prediction, leveraging their proven ability to handle time-series data effectively.

We explore a range of architectures, including Long Short-Term Memory (LSTM), Vanilla LSTM, Recurrent Neural Networks (RNN), Gated Recurrent Units (GRU), hybrid GRU-LSTM models, Stacked LSTM, and Generative Adversarial Networks (GANs). Our study incorporates datasets specifically designed for intraday trading and simple trading scenarios, enabling us to evaluate these models' capabilities in predicting future stock prices.

The research also integrates technical analysis, a widely-used practice among traders and investors, by utilizing technical indicators to enhance the prediction process. Through rigorous experimentation, we analyze the comparative performance of these models, focusing on their ability to capture the complex temporal dependencies and non-linear patterns characteristic of stock market data.

Our findings demonstrate the potential of deep learning models to enhance decision-making in the dynamic stock market environment, providing insights into their strengths, limitations, and practical applications in real-world trading strategies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The Area of Work

The stock market is a highly dynamic and unpredictable financial system that plays a critical role in the global economy. Accurate stock price prediction has been a long-standing challenge due to the inherent complexity and volatility of market behavior. Recent advancements in machine learning and deep learning have provided powerful tools for analyzing historical stock data and predicting future trends. Time-series data, such as stock prices, requires specialized models capable of processing sequential patterns, making techniques like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) particularly effective. LSTMs, a variant of Recurrent Neural Networks (RNNs), excel in capturing temporal dependencies. GRUs, on the other hand, offer computational efficiency and reduced training time due to their simplified gating mechanisms. In this project, we explore various deep learning models, including LSTMs and GRU-based architectures, to forecast stock prices. Additionally, Generative Adversarial Networks (GANs), originally designed for image synthesis, have been adapted for sequence generation in this study, incorporating GRUs as generators and 1-dimensional Convolutional Neural Networks (CNNs) as discriminators.

1.2 Problem Addressed

The financial market's volatility and non-linear behavior make predicting stock prices a formidable task. Traditional time-series analysis methods often fall short in capturing the intricate patterns and dependencies present in financial data. The vanishing and exploding gradient issues of RNNs hinder their efficiency, leading to the development of LSTMs and GRUs.

Stock market data is time-series in nature, consisting of time-ordered data points associated with various variables such as opening, closing, high, and low prices, as well as traded volumes. Effective stock price prediction necessitates the development of robust models that can handle this data's sequential nature while minimizing errors in training and prediction. This study investigates and compares

the performance of advanced deep learning models on datasets of prominent Indian stocks like SBI, HDFC, ICICI, TCS, and Reliance Industries Limited.

1.3 Existing System

Recurrent Neural Networks (RNNs) have been extensively used for time-series predictions but are hindered by vanishing and exploding gradient problems, limiting their capacity to model long-term dependencies. LSTMs address these issues with sophisticated gating mechanisms, but their increased complexity leads to higher training times. GRUs, with a simplified structure, provide faster training while retaining efficiency.

Generative Adversarial Networks (GANs), although traditionally used for generating synthetic images, have been adapted for sequence generation in recent studies. A GAN architecture combining GRUs as generators and CNNs as discriminators has shown promise in generating accurate stock price sequences. By leveraging these advanced techniques and integrating technical indicators like Exponential Moving Average (EMA) and Volume-Weighted Average Price (VWAP), this study builds a comprehensive system to address the shortcomings of traditional methods and provide more accurate stock market predictions.

1.4 Creation of bibliography

Use bibch1.bib file to save your bib format citations. Use the command [**?**] for referring to a particular article [**?**]. and a new citation [**?**].

Chapter 2

Literature Review

The stock market has long been a subject of extensive research, with early studies rooted in the Random Walk Theory and the Efficient Market Hypothesis (EMH). According to EMH, stock prices are inherently unpredictable due to their random movement. However, subsequent studies demonstrated deviations from this theory, suggesting that stock prices do not strictly follow a random walk and can exhibit patterns under certain conditions [1].

Traditional stock market prediction methods relied heavily on statistical models like Autoregressive Moving Average (ARMA) and Autoregressive Integrated Moving Average (ARIMA). While effective for linear time-series data, these methods struggled with the non-linearities and complexities inherent in financial markets. This prompted a shift towards machine learning and neural network-based approaches.

In 2012, Mittal and Goel [2] pioneered the integration of machine learning with sentiment analysis to explore the correlation between market trends and public sentiment. Their approach achieved an accuracy of 75.56

Neural network-based approaches, including Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), marked the next evolution in stock market prediction. These models demonstrated significant improvements over traditional methods, particularly in their ability to capture complex patterns and features. However, they required considerable computational resources and faced challenges in managing highly volatile and large-scale stock market data.

Deep learning models like Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU) further advanced the field by effectively modeling time-series data. Nabipour et al. [3] conducted an extensive study using technical indicators such as n-day moving averages, Relative Strength Index (RSI), and Commodity Channel Index (CCI) across various sectors. They tested ANN, RNN, LSTM, and tree-based models, concluding that LSTM performed best based on root-mean-squared error (RMSE) metrics. The computational efficiency of GRU models was also noted as an advantage for practical applications.

Recent studies emphasize the integration of

technical indicators and multivariate analysis to improve prediction accuracy. Unlike traditional univariate models, multivariate approaches incorporate indicators such as moving averages, Bollinger Bands, and RSI to capture a comprehensive view of market dynamics. This is particularly useful for high-frequency trading, where stock data is produced in massive volumes and characterized by significant volatility.

Despite their potential, deep learning models face limitations, including long training times and challenges in adapting to new data in real time. Kumar et al. [4] highlighted the strengths of deep learning models in detecting non-linearities and extracting significant features without manual intervention. These strengths motivate ongoing efforts to optimize deep learning frameworks for real-time forecasting and decision-making in high-frequency trading scenarios.

This review underscores the evolution of stock market prediction methodologies, highlighting the transition from traditional statistical models to advanced deep learning techniques. While these methods have demonstrated significant improvements, challenges related to training efficiency and real-time adaptability remain key areas for further research and development.

Chapter 3

Proposed Work

3.1 DATASET

The datasets used in this study were obtained from the Yahoo Finance library, a reliable source for financial market data. We selected stock data to analyze both single-stock prediction and intraday trading scenarios, aiming to evaluate the performance of our models across varying use cases.

For single-stock prediction, we utilized the historical data of **Reliance Industries Limited**, one of India's leading corporations. This dataset includes daily trading information such as the opening price, closing price, high, low, volume, and adjusted close values, covering an extensive time period. This data was instrumental in training and testing our models to predict long-term stock trends.

For intraday trading analysis, we collected datasets for five major stocks: **HDFC Bank**, **ICICI Bank**, **Larsen & Toubro (L&T)**, **State Bank of India (SBI)**, and **Tata Consultancy Services (TCS)**. These datasets include minute-level stock price movements sampled at **15-minute intervals**, capturing essential features such as timestamp, open, high, low, close, and volume. This high-frequency data provided detailed insights into short-term market trends, which are crucial for intraday trading strategies.

The inclusion of both daily data for Reliance Industries and 15-minute interval intraday data for the five stocks ensures a comprehensive evaluation of our deep learning models. The selected stocks are significant players in the Indian stock market, known for their high trading volumes and volatility, making them ideal for assessing the applicability of our predictive techniques. This diverse dataset forms the backbone of our study, facilitating a robust examination of the proposed models across different trading scenarios.

3.2 Technical Indicators

To predict stock prices effectively, traders often rely on charts to analyze price movements and technical indicators. Among the many tools available, Moving Average Convergence Divergence (MACD) and Relative Strength Index (RSI) are commonly used for determining buy and sell signals. However, to capture current market trends more accurately, indicators such as the Exponential Moving Average

(EMA) and Volume Weighted Average Price (VWAP) are particularly effective.

Exponential Moving Average (EMA) is the exponentially weighted average price of a stock over the previous d data points, placing greater weight on recent prices. This property allows EMA to react quickly to price changes, making it a valuable tool for identifying recent trends. The smoothing coefficient α, which determines the weight of recent data, is calculated as:

$$\alpha = \frac{2}{d+1}$$

The EMA at time t (EMA_t) is computed as:

$$EMA_t = \alpha Y_t + (1-\alpha)EMA_{t-1}$$

Here, Y_t represents the closing price at time t. For intraday trading, it is considered effective to use a value for d between 5 and 20 to capture short-term price movements.

Volume Weighted Average Price (VWAP) is another essential indicator that calculates the average price of a stock during a trading session, weighted by its trading volume. VWAP is computed as:

$$VWAP = \frac{\sum_{i=1}^{n} p_i v_i}{\sum_{i=1}^{n} v_i}$$

In this formula, p_i and v_i represent the price and volume of the stock at the i-th time interval, respectively, and n is the total number of intervals in the trading session. VWAP resets at the start of each new session and provides insights into whether the stock price is trading above or below its average for that session, aiding in intraday trading strategies.

To construct a time series for historical analysis, EMA(10) and VWAP are calculated and combined with stock prices. These indicators are integrated into the dataset to enhance its predictive power, allowing models to better capture trends and trading behaviors. Together, EMA and VWAP offer traders a comprehensive view of both price trends and trading volume, supporting more informed decision-making in stock trading.

3.3 Deep Learning techniques

3.3.1 RNN

Recurrent Neural Networks (RNNs) are specialized neural networks designed for processing sequential data by leveraging their recurrent structure to handle input sequences of variable lengths and produce outputs of variable lengths. They achieve this by sharing weights across time steps and maintaining a hidden state that captures information from previous steps, making them ideal for tasks like natural language processing, speech recognition, and time-series prediction. RNNs are trained using backpropagation through time (BPTT), where gradients are propagated through the sequence to update parameters. However, they face challenges with long-term dependencies due to vanishing or exploding gradients, which can be addressed by advanced architectures like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU). RNNs are foundational for sequence-to-sequence tasks such as machine translation and summarization, enabling them to model temporal dependencies effectively.

The Recurrent Neural Network (RNN) processes sequential data by maintaining a hidden state that captures information from previous time steps. The key equations are as follows:

Hidden State Update:

$$h_t = \tanh(W_h \cdot [h_{t-1}, x_t] + b_h)$$

where: - h_t: Hidden state at the current time step. - h_{t-1}: Hidden state from the previous time step. - x_t: Input at the current time step. - W_h: Weight matrix for the hidden layer. - b_h: Bias term for the hidden layer. - $\tanh$: Activation function (hyperbolic tangent).

$$y_t = W_y \cdot h_t + b_y$$

where: - y_t: Output at the current time step. - W_y: Weight matrix for the output layer. - b_y: Bias term for the output layer.

In an RNN, the hidden state h_t serves as a memory, enabling the model to capture information from earlier in the sequence. The recursive nature of the hidden state update allows the RNN to process sequences of arbitrary length.

3.3.2 LSTM

Long Short-Term Memory (LSTM) networks, introduced in 1997, address the vanishing gradient problem inherent in traditional RNNs, enabling them to handle long-term dependencies in sequence data effectively. The core of an LSTM is its cell, which maintains two states: the cell state, serving as long-term memory, and the hidden state, functioning as short-term memory. The architecture uses three gates—input, forget, and output—to regulate the flow of information. The input gate determines what new information to store, the forget gate decides what to discard from the cell state, and the output gate controls what information to expose as output. This gating mechanism ensures that LSTMs can retain essential information over extended sequences while discarding irrelevant data. By mitigating issues like vanishing gradients, LSTMs are ideal for tasks requiring the modeling of long-term dependencies, such as language modeling, machine translation, and time-series forecasting.

The key equations are as follows:

Forget Gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

where f_t is the forget gate vector, σ is the sigmoid activation function, W_f and b_f are the weight matrix and bias for the forget gate, h_{t-1} is the hidden state from the previous time step, and x_t is the input at the current time step.

Input Gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

where i_t is the input gate vector, $\tilde{C}_t$ is the candidate cell state, W_i, W_C are the weight matrices, and b_i, b_C are the biases for the input gate and candidate cell state.

Cell State Update:

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

where C_t is the updated cell state, C_{t-1} is the previous cell state, and $\odot$ denotes element-wise multiplication.

Output Gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

where o_t is the output gate vector, W_o and b_o are the weight matrix and bias for the output gate.

Hidden State Update:

$$h_t = o_t \odot \tanh(C_t)$$

where h_t is the updated hidden state, which serves as the output of the LSTM unit for the current time step.

In summary, the LSTM unit uses the forget gate f_t, input gate i_t, and output gate o_t to control the flow of information, ensuring that long-term dependencies can be captured effectively.

3.3.2.1 Vanilla LSTM

Vanilla LSTM, the basic form of Long Short-Term Memory networks, is well-suited for stock market prediction due to its ability to model long-term dependencies in sequential data. It uses a cell structure with three gates—input, forget, and output—to manage information flow effectively. The input gate decides which new information from the stock price data to store, the forget gate determines which past data to discard, and the output gate selects what to predict or output at each time step. This architecture allows Vanilla LSTMs to learn patterns in historical stock prices, such as trends and seasonality, making them highly effective for forecasting future price movements.

In stock prediction, Vanilla LSTMs can handle noisy and complex financial time-series data, capturing both short-term fluctuations and long-term trends. Applications include predicting daily stock closing prices, identifying trends for investment strategies, and assisting in automated trading systems. By learning temporal dependencies, LSTMs help analysts and investors make informed decisions in the volatile stock market.

3.3.2.2 Stacked LSTM

Stacked LSTM refers to a multi-layer architecture where multiple LSTM layers are stacked on top of each other, enabling the model to learn more complex and hierarchical patterns in sequential data. This architecture is particularly effective for stock market prediction, as it captures both low-level features, like daily price fluctuations, and high-level patterns, such as long-term trends or market cycles. Each LSTM layer processes the sequential data and passes its output to the next layer, allowing deeper feature extraction.

For stock prediction, Stacked LSTMs excel at analyzing historical price data to forecast future stock movements with greater accuracy. They are especially beneficial when working with large datasets, as their hierarchical learning structure can uncover nuanced relationships in financial time series. Applications include predicting stock prices, volatility analysis, and identifying buy/sell signals, making Stacked LSTMs a powerful tool for developing robust trading algorithms and investment strategies.

3.3.2.3 Bidirectional LSTM

Bidirectional LSTM (BiLSTM) extends the capabilities of standard LSTMs by processing

sequential data in both forward and backward directions. This dual perspective allows BiLSTM to capture context from both past and future time steps, making it particularly advantageous for stock market prediction, where understanding patterns in historical data and anticipating trends are equally important.

In stock prediction, BiLSTM models analyze historical price movements while also considering how future trends might influence the interpretation of past data. This bidirectional processing enables the model to uncover deeper temporal relationships and dependencies in stock prices, leading to more accurate predictions. Applications include forecasting stock prices, analyzing sentiment-driven market fluctuations, and enhancing algorithmic trading strategies. By leveraging insights from both directions in time, BiLSTM offers a powerful approach for making informed decisions in volatile financial markets.

3.3.3 GRU

Gated Recurrent Units (GRUs), introduced by Kyunghyun Cho et al. in 2014, are a streamlined type of recurrent neural network designed to address exploding and vanishing gradient issues. They are simpler than Long Short-Term Memory (LSTM) units, using fewer parameters due to the absence of an output gate, while maintaining comparable performance in tasks like natural language processing and time-series prediction. GRUs utilize two gates: a reset gate, which controls the retention of past information for short-term dependencies, and an update gate, which manages the integration of old and new states for long-term dependencies. This efficiency and adaptability make GRUs particularly well-suited for applications like stock market prediction, where capturing sequential and temporal patterns is critical.

The key equations are as follows:

1. Update Gate:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$$

where: - z_t: Update gate vector. - σ: Sigmoid activation function. - W_z, b_z: Weight matrix and bias for the update gate. - h_{t-1}: Previous hidden state. - x_t: Input at the current time step.

2. Reset Gate:

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$$

where: - r_t: Reset gate vector. - W_r, b_r: Weight matrix and bias for the reset gate.

3. Candidate Hidden State:

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h)$$

where: - $\tilde{h}_t$: Candidate hidden state. - $\odot$: Element-wise multiplication. - W_h, b_h: Weight matrix and bias for the candidate hidden state.

4. Final Hidden State:

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

where: - h_t: Updated hidden state.

The GRU uses the update gate z_t to determine how much of the past hidden state h_{t-1} should be retained and how much of the candidate hidden state $\tilde{h}_t$ should be incorporated. The

reset gate r_t controls how much of the previous hidden state h_{t-1} is used in the computation of the candidate hidden state $\tilde{h}_t$.

3.3.4 GANs

Generative Adversarial Networks (GANs), introduced by Ian Goodfellow et al. in 2014, are a class of neural networks designed for generating data that mimics a given dataset. GANs consist of two networks: a generator, which creates synthetic data, and a discriminator, which evaluates the authenticity of the data. The two networks are trained simultaneously in a competitive process, where the generator aims to improve its outputs to deceive the discriminator, and the discriminator enhances its ability to identify real data. In the context of stock market prediction, GANs can be used to generate realistic financial time series data, enabling better training of predictive models by addressing data scarcity or imbalances. Additionally, GANs can uncover patterns and anomalies in stock market trends, enhancing the accuracy of predictions by simulating diverse market scenarios.

Generative Adversarial Networks (GANs) consist of two neural networks, a Generator (G) and a Discriminator (D), which compete with each other in a zero-sum game. The key equations are as follows:

GAN Objective Function: The objective of the GAN is to optimize the following value function $V(G, D)$:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)} \big[\log D(x) \big] + \mathbb{E}_{z \sim p_z(z)} \big[\log \big(1 - D(G(z)) \big) \big]. \quad (3.1)$$

where: - $p_{data}(x)$: The true data distribution. - $D(x)$: The discriminator's prediction for real data x. - $D(G(z))$: The discriminator's prediction for synthetic data $G(z)$.

Training Objectives: The discriminator D is trained to maximize the probability of correctly distinguishing between real and synthetic data:

$$\max_D \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

The generator G is trained to minimize the discriminator's ability to distinguish synthetic data from real data:

$$\min_G \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

Alternatively, the generator can be trained by maximizing:

$$\max_G \mathbb{E}_{z \sim p_z(z)} [\log D(G(z))]$$

In GANs, G and D iteratively update their parameters in a competitive manner until G produces data that D cannot distinguish from the real data.

3.4 EVALUATION METRIC

The accuracy of the forecasting models was evaluated using two key metrics: **Mean Absolute Error (MAE)** and **Root Mean Square Error (RMSE)**. These metrics are widely used in regression models to assess prediction performance.

MAE calculates the average magnitude of prediction errors by taking the absolute difference between the actual and predicted values. It is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

where y_j represents the actual values, $\hat{y}_j$ denotes the predicted values, and n is the total number of predictions. Unlike other metrics, MAE assigns equal weight to all errors.

RMSE, on the other hand, is a quadratic metric that assigns greater weight to larger errors by squaring them before averaging. This makes RMSE more sensitive to significant deviations between actual and predicted values. It is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

Together, these metrics provide complementary insights into model performance, with RMSE penalizing larger errors more heavily. They were used to compare the accuracy of the forecasting models and evaluate their predictive capabilities.

Chapter 4

Simulation and Results

4.1 Prediction of stock price for Reliance dataset

Here we have used various types of deep learning models and methodologies on the stock price of the Reliance dataset. This is a comparison study of these models to see which one gives us the best accuracy.

4.1.1 Recurrent Neural Networks(RNN)

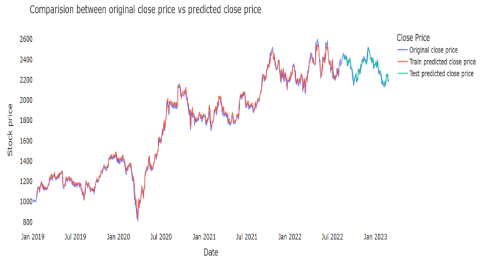

FIGURE 4.1: caption

4.1.2 Stacked LSTM

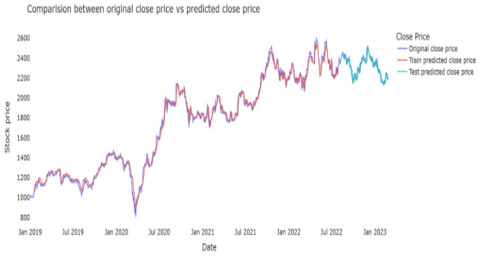

FIGURE 4.2: caption

4.1.3 Gated Recurrent Units(GRU)

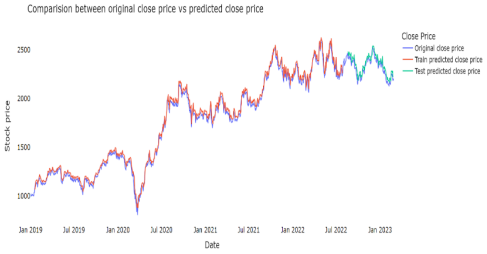

FIGURE 4.3: caption

4.1.4 LSTM-GRU

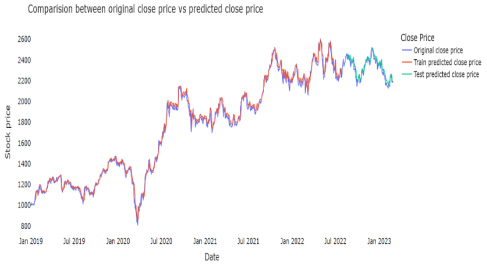

FIGURE 4.4: caption

4.1.5 Generative Adversarial Networks(GANs)

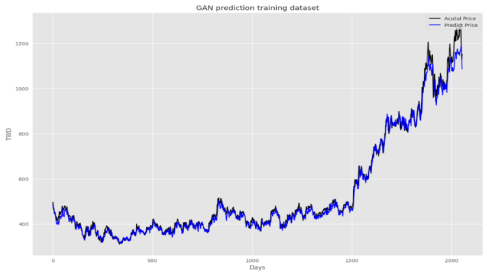

FIGURE 4.5: caption

Architecture	RMSE (Training Dataset)	RMSE (Testing Dataset)
RNN	38.539	43.143
Stacked LSTM	44.779	36.261
GRU	34.761	30.701
Combined LSTM-GRU	34.840	28.858

TABLE 4.1: RMSE values of different architectures on training and testing datasets

4.2 Intraday trading prediction

Here there is a use of multivariate analysis used on multiple stocks' datasets namely SBI, HDFC, LT, TCS and ICICI. Deep Learning models which gave higher accuracy are used for these datasets named as Vanilla, Stacked and Bidirectional LSTM. The data is preprocessed by calculating technical indicators like EMA and VWAP, and splitting the sequences into training and testing datasets. Models are trained to minimize MSE, and their performance is evaluated using RMSE and visualized by comparing actual vs. predicted stock prices. Univariate time-series were derived from stock historical prices, whereas multivariate time-series include the EMA and VWAP along with the historical prices.

4.2.1 SBI Dataset

4.2.1.1 Vanilla LSTM

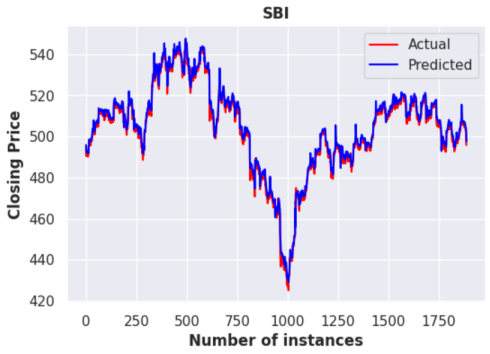

FIGURE 4.6: caption

4.2.1.2 Stacked LSTM

FIGURE 4.7: caption

4.2.1.3 Bidirectional LSTM

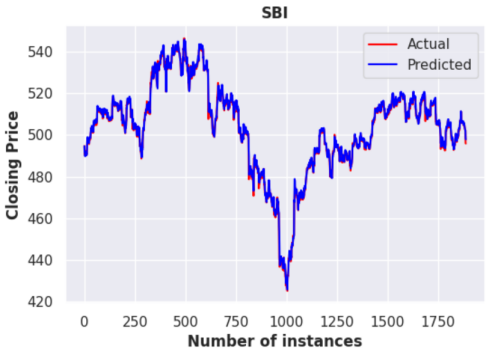

FIGURE 4.8: caption

4.2.2 ICICI Dataset

4.2.2.1 Vanilla LSTM

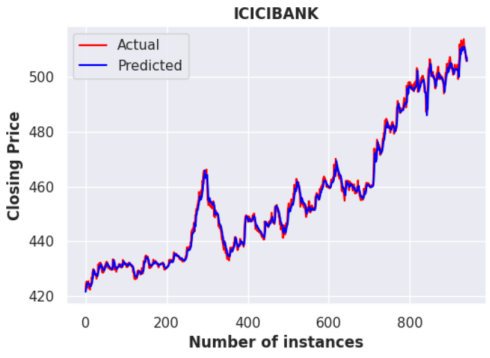

FIGURE 4.9: caption

4.2.2.2 Stacked LSTM

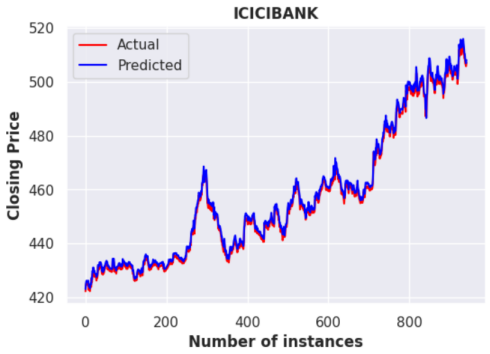

FIGURE 4.10: caption

4.2.2.3 Bidirectional LSTM

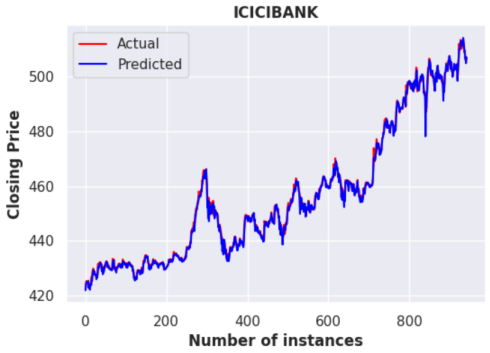

FIGURE 4.11: caption

4.2.3 HDFC Dataset

4.2.3.1 Vanilla LSTM

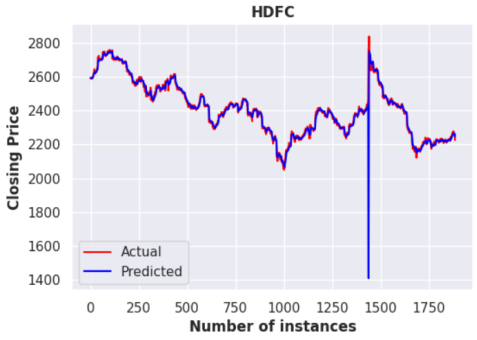

FIGURE 4.12: caption

4.2.3.2 Stacked LSTM

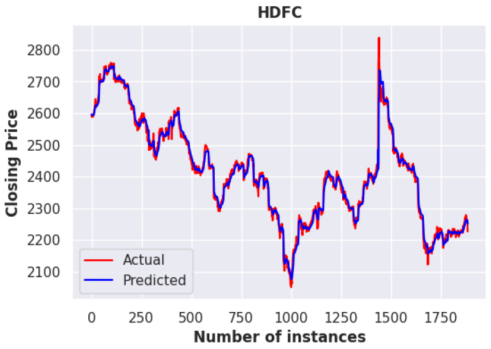

FIGURE 4.13: caption

4.2.3.3 Bidirectional LSTM

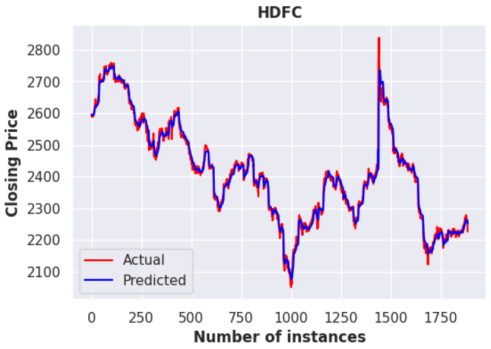

FIGURE 4.14: caption

4.2.4 LT Dataset

4.2.4.1 Vanilla LSTM

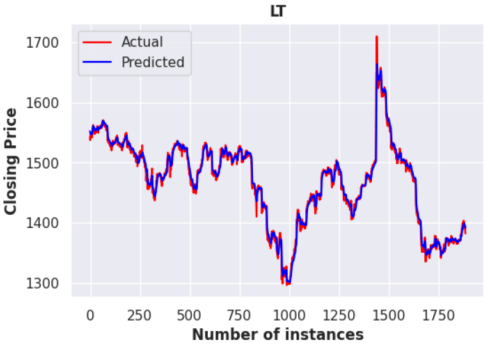

FIGURE 4.15: caption

4.2.4.2 Stacked LSTM

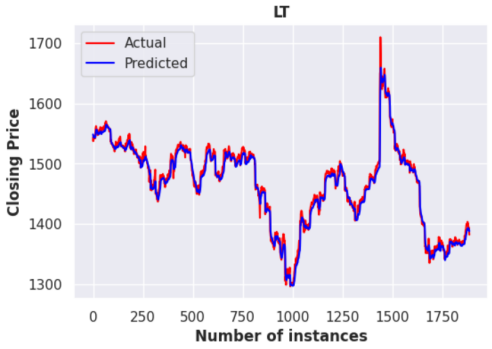

FIGURE 4.16: caption

4.2.4.3 Bidirectional LSTM

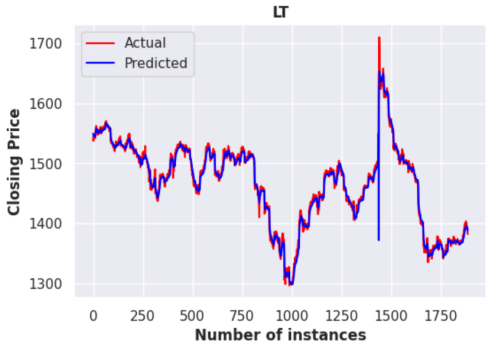

FIGURE 4.17: caption

4.2.5 TCS Dataset

4.2.5.1 Vanilla LSTM

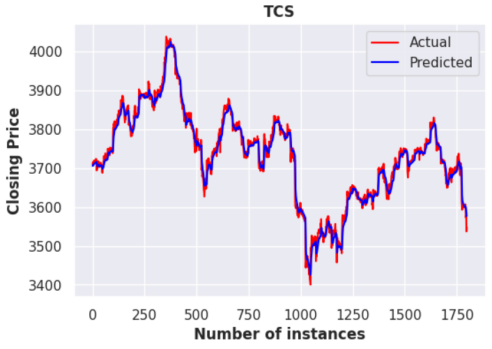

FIGURE 4.18: caption

4.2.5.2 Stacked LSTM

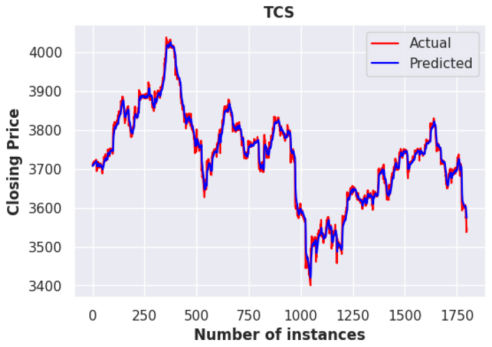

FIGURE 4.19: caption

4.2.5.3 Bidirectional LSTM

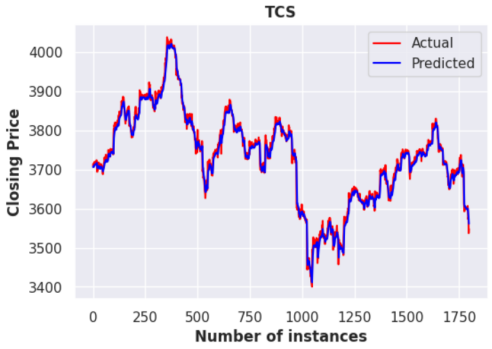

FIGURE 4.20: caption

4.2.6 Comparison of RMSE values of all the datasets

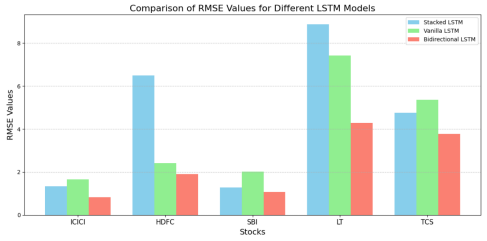

FIGURE 4.21: caption

Chapter 5

Conclusions and Future Work

5.1 Conclusion

This project investigated and compared various deep learning models for stock market prediction, including Stacked LSTM, Vanilla LSTM, Bidirectional LSTM, GRU, RNN, LSTM-GRU combinations, and GANs. The analysis spanned across five major stocks: ICICI, HDFC, SBI, LT, and TCS, leveraging key features such as EMA (Exponential Moving Average) and VWAP (Volume Weighted Average Price) to enhance model performance.

5.1.1 Key Findings

- **Model Comparison:**
  - **Bidirectional LSTM** consistently demonstrated the best accuracy, effectively capturing temporal dependencies in both forward and backward directions. For instance, it achieved the lowest RMSE values, such as 0.83 for ICICI, outperforming Stacked LSTM (1.33) and Vanilla LSTM (1.66).
  - **Stacked LSTM** showed competitive results due to its deeper architecture but was less effective than Bidirectional LSTM.
  - **Vanilla LSTM** performed well on simpler datasets but struggled with complex data dynamics.
  - **GRU and Stacked LSTM** were effective in capturing temporal dependencies but were computationally intensive.
  - **RNNs** faced challenges with long-term dependencies.
  - **GANs** introduced innovative prediction approaches but required meticulous tuning to avoid instability.

- **Stock-Specific Insights:**
  - Prediction accuracy varied across stocks, with stable stocks (e.g., ICICI, SBI) yielding lower RMSE values compared to volatile ones (e.g., LT, TCS).
  - EMA and VWAP contributed significantly to capturing price trends and variations, improving predictive accuracy.

5.1.2 Limitations

- Models are sensitive to market volatility and risk overfitting.
- GANs, while innovative, need more robust tuning and evaluation.

5.1.3 Future Scope

- Incorporating additional features, such as macroeconomic indicators or sentiment analysis from financial news.
- Developing ensemble methods or hybrid architectures combining LSTMs with CNNs or attention mechanisms.
- Extending predictions to multi-step forecasting or longer time horizons.
- Creating a user-friendly platform or website to enable investors to leverage these models for informed decision-making.

In conclusion, the project underscores the potential of advanced deep learning models in tackling the challenges of stock market prediction. Among these, Bidirectional LSTM emerged as a robust solution, demonstrating superior predictive power across datasets. This work provides a strong foundation for further exploration of cutting-edge techniques, offering valuable tools for investors and financial analysts.

Bibliography

[1] D. R. Gallagher, M. Kavussanos, and R. Butler, "Random walk theory and evidence in financial markets: A review," in *Proceedings of the Financial Markets Conference, 2005*, pp. 45–60, University of Sydney Business School, 2005.

[2] A. Mittal and A. Goel, "Stock prediction using twitter sentiment analysis," *Stanford University CS229*, 2012.

[3] M. Nabipour, P. Nayyeri, H. Jabani, A. Mosavi, and E. Salwana, "Deep learning for stock market prediction," *IEEE Access*, vol. 8, pp. 185816–185831, 2020.

[4] A. Kumar, Y. Kumar, A. Kumar, and R. Prasad, "Deep learning-based stock price prediction using lstm and technical analysis indicators," *International Journal of Business Forecasting and Marketing Intelligence*, vol. 6, no. 1, pp. 1–12, 2021.