Real-Time Stock Market Prediction with Sentiment Analysis

Gopika Rajeendar

ABSTRACT

Harnessing the vast data streams from social media and financial markets, this study demonstrates how real-time sentiment analysis can enhance stock market predictions. By integrating Twitter sentiment with historical and real-time stock data from Yahoo Finance and approximately 9 million tweets, advanced models such as Decision Tree Regressor, LSTM, CNN, and GRU are employed to forecast time series close prices. Utilising VADER for Twitter sentiment and Apache Kafka for real time data streaming, the research delves into the complex interplay between market sentiment and stock performance. Employing PySpark and Hadoop as our distributed computing framework, we managed the vast data streams from social media and financial markets, ensuring efficient processing and analysis.This approach provides valuable insights for investors and sets the stage for innovations in financial forecasting through the application of machine learning techniques.

Keywords: Decision Tree, LSTM, CNN, GRU, Vader, Kafka, Sentiment Analysis, PySpark, Hadoop, Distributed Computing, Time Series, Big Data

1 INTRODUCTION

The stock market is an intricate and dynamic system influenced by various factors, including economic data, market trends, and investor sentiment. Real-time prediction of stock market movements is a critical challenge for traders, investors, and researchers, offering potential opportunities to maximize returns and mitigate risks.

This research focuses on Tesla stock as a case study, exploring the potential of combining sentiment analysis from social media platforms, such as Twitter, with historical and real-time stock market data to improve the accuracy of stock market predictions. Drawing upon historical stock data from Yahoo Finance for a designated period, the study employs both traditional and advanced deep learning models—including Decision Tree Regressor, LSTM, CNN, and GRU—to perform time series predictions of close prices. Concurrently, the study preprocesses tweets from the same time frame, utilizing the VADER tool for sentiment analysis to gauge market sentiment.

By incorporating sentiment data into the models, the research finds significant improvements in prediction accuracy. For instance, the GRU model achieved the lowest Mean Absolute Error (MAE) and loss values compared to other models. The inclusion of sentiment data also led to reduced training times across various models, enhancing the efficiency of the prediction process.

For real-time implementation, the most effective model identified through this research is applied to a randomly generated stock name. This dynamic process is facilitated by streaming real-time

data through Apache Kafka utilizing the Cashtag Piggybacking dataset, setting the stage for adaptive financial forecasting.

By analyzing the intricate relationship between market sentiment and stock performance, the study aims to identify correlations that may offer valuable insights for investors and market participants. The research contributes to the field by evaluating the efficacy of various machine learning models and sentiment analysis techniques, providing a foundation for further innovation in real-time stock market prediction.

2 RELATED WORK

The integration of news sentiment analysis and social media data in stock market prediction has shown promising results in enhancing forecasting accuracy. Prior work in the field has demonstrated the significant impact of incorporating sentiment data from news sources and social media platforms, such as Twitter, into predictive models.

The study "Stock Price Prediction Using News Sentiment Analysis" [4] focused on examining the relationship between financial news sentiment and stock price volatility. By collecting a large dataset of financial news articles related to S&P 500 companies from 2013 to 2017, the research employed machine learning models such as ARIMA, RNN and LSTM to integrate real-time news sentiment into traditional stock price prediction methods. This approach demonstrated a significant improvement in prediction accuracy compared to models that relied solely on historical stock data. The use of deep learning techniques allowed for a more nuanced analysis of news sentiment, showcasing the impact of integrating such data on financial forecasting.

On the other hand, the paper "Social Media and Stock Market Prediction: A Big Data Approach" [1] explored the impact of social media sentiment on stock market predictions. By collecting data from platforms such as Twitter and Yahoo Finance, the study applied machine learning models like Linear Regression, Generalized Linear Regression, and other classification models such as Naive Bayes and Logistic Regression to unstructured social media data. The integration of social media sentiment data led to notable improvements in prediction accuracy. The study showcased the potential of big data and machine learning in harnessing vast volumes of unstructured data from social media platforms for enhanced stock market prediction, offering new avenues for financial forecasting.

Building on these foundational works, our study contributes a novel approach to real-time stock market prediction by combining historical and real-time stock market data with sentiment analysis from Twitter. We utilize VADER for analyzing social media sentiment and employ advanced deep learning models, including LSTM, CNN, and GRU, to predict time series close prices. Our approach leverages Apache Kafka for real-time data streaming, allowing us to process data efficiently and update models continuously.

By integrating these advanced techniques, our study provides a more comprehensive and nuanced understanding of the relationship between market sentiment and stock performance. Our contribution lies in the use of cutting-edge deep learning models combined with real-time data processing to enhance the accuracy and stability of stock market predictions, offering valuable insights for traders, investors, and researchers.

3 DATA AND PREPARATION

3.1 Historic Data

For this study, stock market data from May to September 2017 was sourced from Yahoo Finance, a comprehensive and reliable provider of financial data. The data provides detailed coverage of various financial indicators including open, close, high, and low prices, as well as volume, dividends, and stock splits.

In preprocessing our dataset, we leveraged PySpark's MLlib for its efficient data scaling and transformation capabilities. Specifically, we utilised the 'MinMaxScaler' from MLlib to normalise the 'Close' prices within a range of 0 to 1, ensuring input standardisation for our models. To construct feature vectors critical for time series forecasting, we applied a 'Window' function, which facilitated the extraction of the last 10 days of scaled 'Close' prices as features for predicting the current day's price. This method utilises a sliding window approach to capture temporal dependencies and trends within the data.

Further, we integrated PySpark's SQL functions to manipulate and prepare the dataset effectively. The 'lag' function along with 'Window', allowed for the creation of lagged features directly within our data pipeline, essential for our time series analysis. These transformations were pivotal in structuring our data appropriately for both training and testing phases, ensuring a robust setup for high-performance machine learning applications performance.

3.2 Sentiment Data

The Twitter dataset used in this study is sourced from the Cashtag Piggybacking dataset, which is enriched with sentiment data. This dataset is composed of approximately 9 million tweets mentioning stocks (cashtags) traded on major U.S. markets, collected between May and September 2017. The data was filtered to include only tweets related to Tesla stock, allowing for focused analysis of sentiment and stock price movements specific to the company. After processing the raw data and computing daily sentiment scores, these scores were integrated with the historical price data. This synthesis enables a thorough examination of the relationship between public sentiment on social media and stock price fluctuations, providing valuable insights into market trends and company perception. The enriched financial data and user information from the tweets add essential context for interpreting their impact on stock performance.

3.3 Real-Time Data

Apache Kafka is a distributed streaming platform that enables real-time data processing and storage, making it ideal for ingesting stock close data from Yahoo Finance. The project uses Kafka to establish a data pipeline that collects stock close data from the past 300 days to enable predictive modeling of future stock prices.

Alphavantage API has been used to collect news data from the last 300 days from the prediction date to calculate news sentiment. The Yahoo Finance API has also been used to scrape the last 300 days stock closing prices to do prediction.

Kafka employs a producer-consumer model where the producer publishes stock close data to a Kafka topic for consumption by

various applications. This architecture is both scalable and fault-tolerant, allowing for efficient and continuous data collection and real-time analysis.

By streaming data from Yahoo Finance, Kafka facilitates the use of machine learning models for predicting future stock prices. Its flexibility supports data processing applications such as transformation, filtering, and aggregation, which are essential for accurate predictive modeling.

4 METHODOLOGY

4.1 Distributed Computing Framework

We have leveraged TensorFlow's MirroredStrategy to harness distributed computing for training our deep learning models — CNN, GRU, and LSTM across multiple GPUs. This approach efficiently allocates the workload of training across the available GPUs, accelerating training times and maximizing computational resources.

Within the scope of MirroredStrategy, the model architecture is defined once and mirrored across all available GPUs. This ensures that each GPU holds an identical copy of the model and processes a portion of the data. During the training phase, 'MirroredStrategy' automatically handles the distribution of data and gradients across the GPUs, effectively synchronizing updates to model weights. The strategy also manages the aggregation of gradients across all devices after each training step, providing consistent updates to the model's parameters. This efficient parallelism speeds up the training process and enhances model performance by allowing the use of larger batch sizes, which can lead to better generalization.

In addition to using MirroredStrategy for distributed computing, we have implemented a distributed computing framework leveraging big data technologies such as Apache Hadoop and Apache Spark to manage and process large volumes of data efficiently. The study begins by downloading the tweets dataset from an external source and uploading it to the Hadoop Distributed File System (HDFS) after unzipping the file. HDFS serves as the storage layer for large-scale data, providing a distributed and scalable solution for accessing and processing data.

Spark, a powerful in-memory distributed computing engine, is employed for data processing and analytics. Reading the CSV file from the specified HDFS path, Spark handles the tweets dataset with its high-speed, parallel processing capabilities. The combination of Hadoop and Spark allows for seamless data handling, transformation, and analysis across multiple nodes in a distributed environment. This architecture is well-suited for processing large-scale data, as it provides high performance and scalability. Additionally, Spark's ease of use and support for various data sources and formats make it an ideal choice for processing and analyzing complex datasets.

Overall, the distributed computing framework, in combination with MirroredStrategy, is essential for efficiently managing large datasets and supporting advanced data analytics. By leveraging the capabilities of Hadoop and Spark, we effectively analyze tweets data and integrate it with other data sources for real-time stock market prediction. This approach demonstrates the advantages of using big data technologies in financial analytics and forecasting.

4.2 Sentiment Analysis

Sentiment analysis is a crucial component of this study as it enables the extraction of opinions and emotions expressed in twitter text data, which can provide valuable insights into market sentiment and trends. By analyzing social media content and news articles mentioning specific stocks, the study can gain a better understanding of public perception and its potential impact on stock market movements. In our project, we obtained the sentiment scores using VADER and concatenated them with other features to train the models. This approach allowed us to evaluate whether integrating sentiment data with other inputs improved model performance compared to using only traditional features without sentiment analysis.

4.2.1 Vader. One tool commonly used for sentiment analysis is VADER (Valence Aware Dictionary and Sentiment Reasoner). VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to social media text. It calculates sentiment scores based on positive, negative, and neutral word frequencies in a given text. The overall sentiment score is calculated as the compound score, which represents the polarity of the text as a whole. VADER is particularly well-suited for analyzing Twitter data due to its ability to handle slang, emoticons, and other informal language.

4.3 Stock Prediction

In this study, four distinct models — three deep learning models (LSTM, CNN, and GRU) and a traditional machine learning model (Decision Tree) — are chosen for time series prediction, each offering unique advantages tailored to the task at hand. Employing past 10 days' closing values as the input variable (X) to predict today's closing value (Y) aligns with the nature of time series forecasting, where historical data plays a crucial role in predicting future trends. By leveraging these four diverse models and utilizing past data as predictors, this study aims to explore various approaches to enhance the accuracy and robustness of stock market predictions. The dataset was partitioned into a training set comprising the majority of the data and a test set consisting of the final 30 days to assess the models.

4.3.1 Decision Tree. The Decision Tree model is a simple yet powerful machine learning algorithm used as a baseline model in this study. It is a non-linear model that uses a tree-like structure to represent decision rules based on the input features, facilitating clear and interpretable predictions. For time series forecasting, such as stock market prediction, the Decision Tree model can identify patterns and relationships in the historical data by splitting the input features based on certain thresholds. The model learns to split the data into different branches based on conditions that maximize the separation between different target values. This approach enables the model to handle complex interactions and nonlinear relationships in the data effectively.

The Decision Tree model is implemented using Spark MLlib for efficient training and processing. The model uses the mean squared error as the loss function and mean absolute error as the evaluation metric to assess its predictive performance. The decision tree serves as a reference point for comparing the performance of the advanced deep learning models (LSTM, CNN, and GRU) in this

study. By setting a baseline, it helps us to evaluate the improvement and added value that deep learning models can provide in stock market prediction.

4.3.2 LSTM. The LSTM (Long Short-Term Memory) model is a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data. It is particularly effective for time series prediction tasks like stock market forecasting due to its ability to retain and utilize historical information over extended time periods. The model architecture implemented in this study consists of two LSTM layers, each with 64 units, followed by a densely connected layer with 32 units and ReLU activation function. A dropout layer with a dropout rate of 0.5 is applied for regularization purposes to prevent overfitting. Finally, a single neuron output layer is utilized to produce the predicted value. The model is compiled using the Adam optimizer and the mean squared error loss function, with mean absolute error as the evaluation metric. The LSTM model's robust architecture, coupled with its adaptability to capture temporal dependencies, positions it as a formidable tool for accurate and reliable time series forecasting in domains such as stock market analysis.

4.3.3 CNN. The CNN (Convolutional Neural Network) model, traditionally used in image processing, is adapted here through identifying and extracting relevant features to exploit temporal patterns in sequential data. The model comprises several convolutional layers followed by max-pooling layers to extract and prioritize key features from the input data. Specifically, the CNN architecture includes two Conv1D layers with 32 and 64 filters, respectively, each followed by a max-pooling layer to downsample the extracted features. The flattened output is then passed through densely connected layers with 64 units, applying ReLU activation functions to capture non-linear relationships within the data. Finally, a single neuron output layer produces the predicted value. The model is trained using the Adam optimizer, optimizing the mean squared error loss function with mean absolute error as the evaluation metric. This CNN architecture is tailored to effectively capture temporal patterns in sequential data, making it a valuable tool for accurate time series forecasting tasks.

4.3.4 GRU. Similar to LSTM, GRU (Gated Recurrent Units) is a variant of the traditional RNN architecture designed to address the vanishing gradient problem and capture long-term dependencies in sequential data. It simplifies the architecture of LSTM by merging the forget and input gates into a single "update gate," making it computationally more efficient while retaining similar predictive capabilities. GRU models are often used for tasks like speech recognition, language modeling, and time series forecasting. Comprising multiple GRU layers, the model effectively learns and retains information over extended sequences. Specifically, the GRU architecture consists of two GRU layers, each containing 64 units, facilitating the extraction of intricate patterns from the input data. A densely connected layer with 32 units, coupled with a dropout layer for regularization, enhances the model's generalization capabilities and mitigates overfitting. Finally, a single neuron output layer generates the predicted value. The model is trained using the Adam optimizer, optimizing the mean squared error loss function with mean absolute error as the evaluation metric.

4.4 Real-Time Prediction

The real-time stock prediction system described in this research leverages sentiment analysis on news data and financial data to predict close price of a stock. The system processes data from multiple sources, including news articles and real-time stock prices, using data scraping and API interfaces. The system architecture consists of three producers and three consumers, interacting through three Kafka topics for data transmission. Each producer and consumer component plays a specific role in the data processing pipeline.

The first producer randomly selects one of the top ten stocks every three minutes and sends the stock name to the next Kafka topic, effectively generating stock names. By receiving the stock name, the next component acts as both a consumer and a producer. It reads the stock name from the previous producer, and scrapes the stock's news data using the Alpha Vantage API and stock price data from Yahoo! Finance. The sentiment score of the news articles scraped is then calculated using VADER sentiment scoring. The sentiment score is then combined with the financial data to form a comprehensive dataset for prediction. The system acts as a producer when it sends these combined prices and sentiment data to the next consumer through another Kafka topic.

The next component is where machine learning models and time series forecasting techniques are employed to generate real-time predictions for stock price movements. As a consumer, it reads the combined dataset from the Kafka topic and then utilizes the best model to train the data and predict the close price. The prediction is then sent to the final consumer through a Kafka topic.

The final component reads the prediction from the Kafka topic and displays the final result acting as only a consumer.

5 NUMERICAL RESULTS

5.1 Goals

Our primary objective is to evaluate various neural network models within a distributed computing framework for predicting stock prices. We aim to investigate the extent to which sentiment analysis influences stock market predictions. Ultimately, we plan to leverage the most effective model to forecast real-time stock prices for randomly selected stocks using Apache Kafka.

5.2 Metrics

5.2.1 Mean Absolute Error (MAE). : Mean Absolute Error (MAE) is a metric used to evaluate the performance of a machine learning model in regression tasks. It calculates the average absolute difference between the predicted values and the actual values. The formula for MAE is:

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

Where: - n is the number of samples in the dataset. - y_i is the actual value of the target variable for the i^{th} sample. - $\hat{y}_i$ is the predicted value of the target variable for the i^{th} sample.

5.2.2 Loss Function: Mean Squared Error (MSE). We are using Mean Squared Error (MSE) as our loss function. MSE is a commonly used loss function in regression problems. It calculates the average of the squares of the differences between predicted and actual values.

The formula for MSE is:

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

Where: - n is the number of samples in the dataset. - y_i is the actual value of the target variable for the i^{th} sample. - $\hat{y}_i$ is the predicted value of the target variable for the i^{th} sample.

5.2.3 Computational time: Computational time measures the duration required for a system to process and analyze data, a crucial factor in machine learning and big data applications. It affects efficiency, influencing both the scalability of data operations and the cost-effectiveness of cloud computing resources. This metric is evaluated by timing both the model training and the prediction phases, as each impacts the performance of the machine learning pipeline.

5.3 Baseline Methods for Comparison

While Decision Tree regressors are not commonly used for time series predictions due to their limited ability to capture temporal dependencies, they can still be employed when data exhibits clear, non-linear patterns, serving as a simple baseline for comparison with more sophisticated time series models.

We are using the Decision Tree Regressor as our baseline method due to its simplicity in implementation and interpretation. Additionally, its low computational requirements make it efficient for training and testing, especially with large datasets or limited computational resources. Moreover, it serves as a benchmark to evaluate the performance of more complex regression models, helping to determine if the added complexity yields significant improvements in predictive accuracy.

5.4 Numeric Results and Dicussion

5.4.1 Stock Prediction without Sentiment. Below are two tables presenting the training and prediction times, as well as the Mean Absolute Error (MAE) and loss values for each model without sentiment.

Model	Training Time (s)	Prediction Time (s)
DT	5.93	0.38
LSTM	14.81	0.90
CNN	10.99	1.18
GRU	20.72	1.07

Table 1: Training and prediction times for each model.

Model	MAE	Loss
DT	0.23	0.069
LSTM	0.07	0.0077
CNN	0.09	0.0142
GRU	0.06	0.0067

Table 2: Mean Absolute Error (MAE) and loss for each model.

In the analysis of stock prediction models without sentiment, four models were evaluated: Decision Tree (DT), Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), and Gated Recurrent Units (GRU). GRU demonstrated the best performance with a low MAE of 0.0625 and loss of 0.0067, but at the cost of a longer training time (20.7 seconds) and prediction time (1.07 seconds). LSTM offered a slightly higher MAE of 0.0701 and loss of 0.0077, but with a moderate training time of 14.8 seconds and prediction time of 0.90 seconds. CNN balanced accuracy and speed with a MAE of 0.0922 and loss of 0.0142, taking 11 seconds for training and 1.18 seconds for prediction. DT was the fastest in both training (5.93 seconds) and prediction (0.38 seconds), but its accuracy was lower, with an MAE of 0.2329 and loss of 0.0690. Overall, GRU and LSTM stand out as the most accurate models for stock prediction, though they require longer computation times. CNN offers a reasonable compromise between speed and accuracy, while DT provides rapid results at the expense of precision.

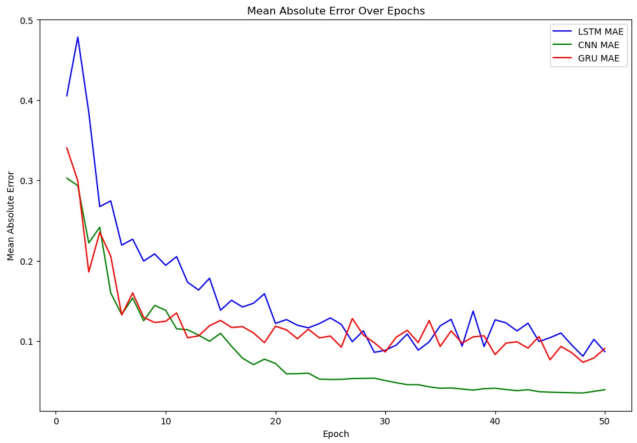

Figure 1: MAE for each model per epoch (without sentiment)

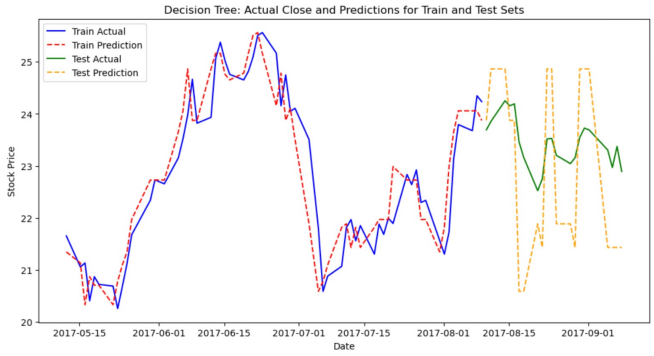

Figure 2: Decision Tree Prediction

5.4.2 Comparison Between Baseline(Decision Tree Regressor) and the Best (CNN) Models' predictions: While both models effectively capture temporal patterns within the training set, their performance diverges noticeably when predicting Tesla's stock prices. The Decision Tree, while quick and straightforward, struggles with accuracy, showing a marked discrepancy between its predictions and the actual stock prices. This suggests that it may be overfitting the training data or failing to capture the underlying trends effectively. On the other hand, the CNN model aligns more closely with the actual price movements, particularly in the test set, indicating its superior ability to model the temporal dependencies and volatility associated with stock prices. The CNN's layered architecture

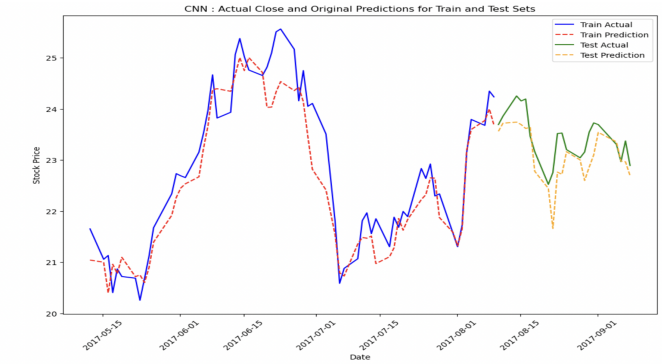

Figure 3: CNN Prediction

allows it to discern more complex patterns in the data, making it a more effective tool for financial time series forecasting than the simpler Decision Tree approach.

5.4.3 Stock Prediction with Sentiment. Below are two tables presenting the training and prediction times, as well as the Mean Absolute Error (MAE) and loss values for each model with Sentiment.

Model	Training Time (s)	Prediction Time (s)
DT	54.1361	0.1996
LSTM	11.6958	1.0258
CNN	9.1042	0.4760
GRU	13.6928	2.1265

Table 3: Training and Prediction Time for different models.

Model	MAE	Loss
DL	0.1503	0.0346
LSTM	0.0780	0.0106
CNN	0.0969	0.0137
GRU	0.0588	0.0056

Table 4: MAE and Loss for different models.

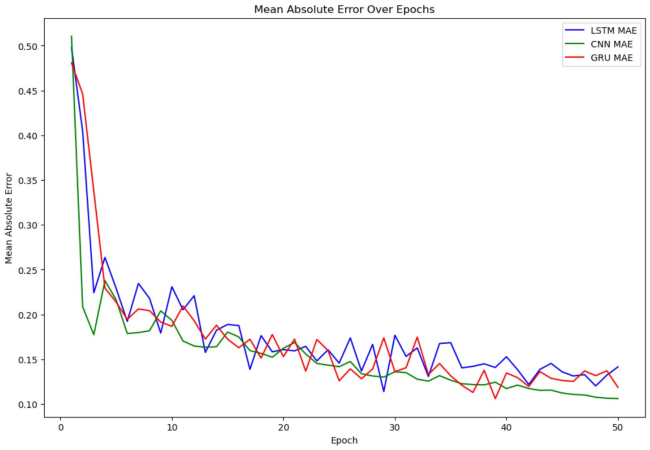

Figure 4: MAE for each model per epoch (with sentiment)

Incorporating sentiment analysis into stock prediction models provides additional insights and contextual information that can potentially enhance the accuracy of predictions. Training and prediction times vary across different models. Decision Tree (DT) takes the longest training time at 54.1361 seconds and has the shortest prediction time at 0.1996 seconds. LSTM, CNN, and GRU all demonstrate shorter training times—11.6958 seconds, 9.1042 seconds, and 13.6928 seconds, respectively—while their prediction times are longer than DT. The models' Mean Absolute Error (MAE) and loss values reveal that GRU achieves the best performance with the lowest MAE (0.0588) and loss (0.0056) values. LSTM and CNN also perform well, with MAE and loss values within a reasonable range. The Decision Tree model exhibits higher MAE (0.1503) and loss (0.0346) values compared to the deep learning models.

A comparison between models with and without sentiment analysis reveals that incorporating sentiment can significantly impact model performance. When sentiment is included, the training times for LSTM, CNN, and GRU are lower—around 11.7, 9.1, and 13.7 seconds, respectively—compared to longer times without sentiment. Prediction times with sentiment for these models are generally higher, with GRU showing the longest prediction time at 2.1265 seconds. In terms of accuracy, incorporating sentiment leads to improvements in MAE and loss across the board. For instance, GRU with sentiment demonstrates a lower MAE (0.0588) and loss (0.0056) compared to without sentiment (MAE of 0.06 and loss of 0.0067). Similarly, LSTM and CNN also show improvements in MAE and loss when sentiment analysis is considered. Overall, while sentiment analysis enhances predictive accuracy and reduces training times, it may also lead to increased prediction times for deep learning models, indicating a trade-off between efficiency and speed.

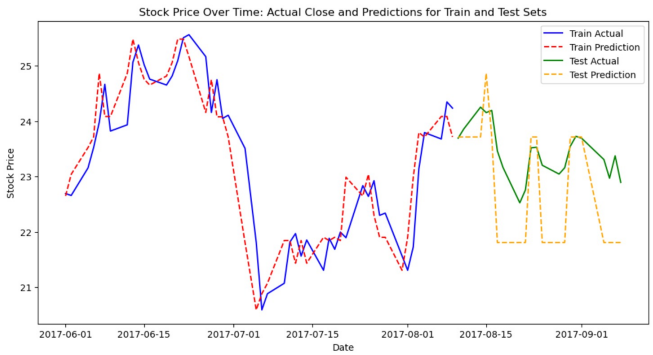

Figure 5: Decision Tree Prediction with Sentiments

Incorporating tweet sentiment analysis markedly improves the predictive performance of both the Baseline (Decision Tree) and the best(GRU model) for Tesla's stock prices, though the impact varies between the models. The Decision Tree, a simpler model, still shows some volatility in its predictions, but with sentiment data, its alignment with actual price trends is noticeably enhanced, suggesting better handling of market sentiment fluctuations. Meanwhile, the GRU model, more sophisticated and adept at processing sequential data, demonstrates even finer accuracy and smoother predictions with the addition of sentiment, capturing subtle shifts

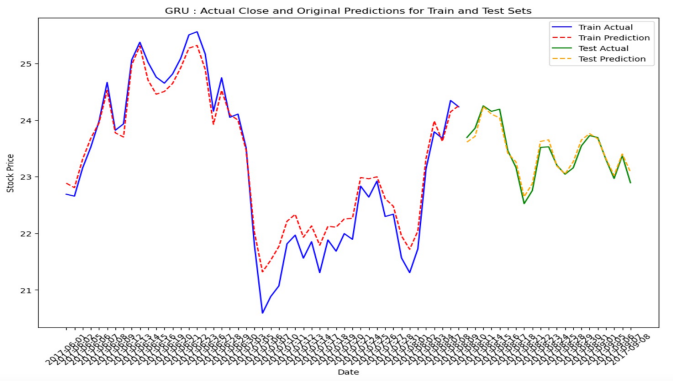

Figure 6: GRU Prediction with Sentiments

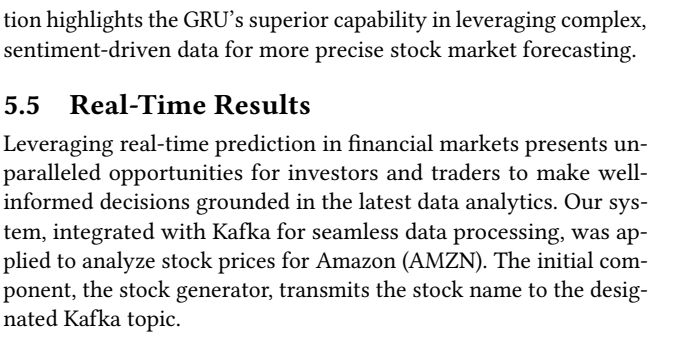

Figure 7: Component 1 - Stock Generator

The succeeding module retrieves the stock name from the Kafka topic and proceeds to extract Amazon's stock prices from Yahoo Finance!, alongside news articles pertaining to Amazon sourced via the AlphaVantage API, covering the preceding 300 days. Utilizing the VADER sentiment analysis tool, sentiment scores for these articles are computed, culminating a comprehensive dataset comprising average sentiment scores, dates, and close prices. This dataset is then transmitted to the next component through an additional Kafka topic.

The modeling component, receiving the dataset from its predecessor, employs the GRU model to forecast the close price for the present day. The GRU model, renowned for its adeptness in utilizing sentiment-driven data for precise stock price predictions, is selected for its superior capabilities. Upon generating the prediction, it is relayed to the subsequent component where further analysis occurs. Here, the change in Amazon's stock price is computed, and a prediction is made regarding the stock price which is anticipated to fall. Through this meticulously orchestrated process, our system offers investors and traders invaluable insights into potential market movements, enhancing their ability to navigate financial markets with confidence and foresight.

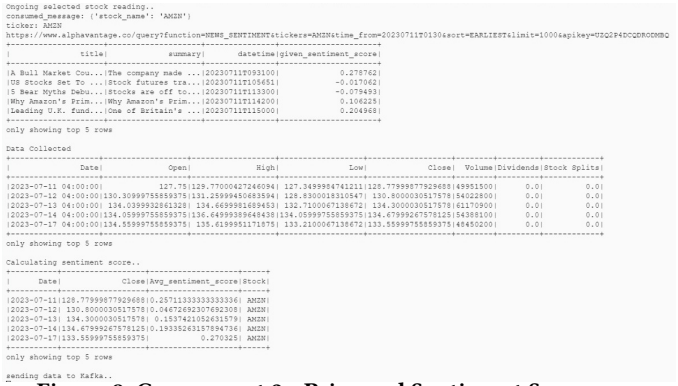

Figure 8: Component 2 - Price and Sentiment Scraper

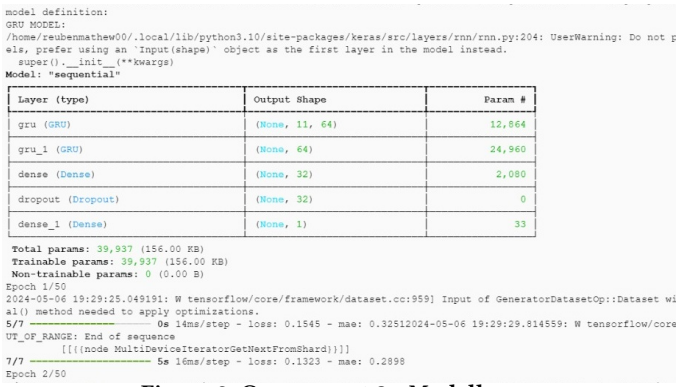

Figure 9: Component 3 - Modeller

Figure 10: Component 4 - Final Output

6 CONCLUSION

This study effectively demonstrates the potential of combining sentiment analysis with advanced machine learning models for real-time stock market prediction. By integrating Twitter sentiment data with historical and real-time stock market data, deep learning models such as LSTM, CNN, and GRU outperform traditional methods, achieving lower Mean Absolute Error (MAE) and loss values. The inclusion of sentiment data leads to enhanced predictive accuracy and reduced training times across the board, though it may result in slightly longer prediction times for deep learning models.

GRU stands out as the top-performing model with the lowest MAE and loss when sentiment is incorporated. LSTM and CNN also benefit from the integration of sentiment data, showing improved performance. However, the Decision Tree model falls short in predictive accuracy, serving as a benchmark for the more sophisticated deep learning models. In conclusion, this study provides compelling evidence that real-time sentiment analysis, when integrated with state-of-the-art machine learning techniques, can

significantly enhance the accuracy and efficiency of stock market prediction. This approach offers valuable insights for traders, investors, and researchers seeking to leverage real-time data and advanced analytics for financial forecasting.

6.1 Future Work

Future work in real-time stock market prediction with sentiment analysis can focus on enhancing the quality and timeliness of data sources to improve model accuracy and robustness. Currently, our real-time system's news data and sentiment analysis rely on an API that may not provide the most recent news updates, limiting its ability to capture up-to-date sentiment for the latest market events.

To overcome this limitation, incorporating more current and diverse news sources from around the world would offer a richer, more immediate understanding of market sentiment and trends. This expansion could include sources across various industries and regions, providing a broader context for analysis and allowing the models to better capture nuances and dynamics in the global financial markets.

Enhancing sentiment analysis techniques by incorporating advanced natural language processing (NLP) models like transformer-based architectures (e.g., BERT, GPT) would allow for a more nuanced understanding of social media language and sentiment. Additionally, integrating other sources of sentiment data, such as financial news and reports, could complement social media insights and provide a more holistic view of market sentiment.

Another promising direction is the development of ensemble modeling approaches, combining different model architectures such as LSTM, CNN, and GRU to improve prediction stability and accuracy. Alternatively, adaptive models that update their parameters in real-time based on new data could enhance the responsiveness of predictions.

A key limitation of the current project is the focus on analyzing sentiment data for a short time frame, which may not fully capture long-term trends. To address this, future work could involve expanding the time frame of sentiment data analysis and exploring scalable approaches to handle larger datasets, enabling continuous model training and real-time adaptation.

Finally, leveraging explainable AI techniques can increase transparency and trust in model predictions by identifying key features and data sources that drive forecasts. This would facilitate informed decision-making and support the adoption of predictive models in real-world financial applications.

REFERENCES

[1] Mazhar Javed Awan, Mohd. Shafry Mohd. Rahim, Haitham Nobanee, Ashna Munawar, Awais Yasin, and Azlan Mohd Zain. 2021. Social media and stock Market Prediction: A big data approach. *Computers, materials continua/Computers, materials continua (Print)* 67, 2 (1 2021), 2569–2583. https://doi.org/10.32604/cmc.2021.014253

[2] Lakshmi Ramani Burra, Bhanu Prasanna Koppolu, Baduganti Deva Karthik, Bolem Lasya Naga Sai Priya, Allam Prasanthi, and Praveen Tumuluru. 2023. Stock Price Prediction using Zero-Shot Sentiment Classification. In *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*. 741–746. https://doi.org/10.1109/ICSSIT55814.2023.10060957

[3] Narayana Darapaneni, Anwesh Reddy Paduri, Himank Sharma, Milind Manjrekar, Nutan Hindlekar, Pranali Bhagat, Usha Aiyer, and Yogesh Agarwal. 2022. Stock Price Prediction using Sentiment Analysis and Deep Learning for Indian Markets. (2022). arXiv:2204.05783

[4] Joshi Kalyani, Prof. H. N. Bharathi, and Prof. Rao Jyothi. 2016. Stock trend prediction using news sentiment analysis. *arXiv* abs/1607.01958 (2016). arXiv:1607.01958

[5] Saloni Mohan, Sahitya Mullapudi, Sudheer Sammeta, Parag Vijayvergia, and David C. Anastasiu. 2019. Stock Price Prediction Using News Sentiment Analysis. In *2019 IEEE Fifth International Conference on Big Data Computing Service and Applications (BigDataService)*. 205–208. https://doi.org/10.1109/BigDataService.2019.00035

[6] Zhihao Peng. 2019. Stocks Analysis and Prediction Using Big Data Analytics. In *2019 International Conference on Intelligent Transportation, Big Data Smart City (ICITBS)*. 309–312. https://doi.org/10.1109/ICITBS.2019.00081

[7] Jashanjit Singh and John Jenq. 2023. A Stock Recommendation System. In *Proceedings of 38th International Conference on Computers and Their Applications (EPiC Series in Computing, Vol. 91)*, Ajay Bandi, Mohammad Hossain, and Ying Jin (Eds.). EasyChair, 164–171. https://doi.org/10.29007/p27x