Stock Prediction Using Convolutional Neural Networks and Twitter Sentiment

3388A

1 Motivation

Accurately predicting stock prices trend is crucial for investors seeking to optimize returns and for maintaining market stability. However, this task remains challenging due to inherent volatility and noise in financial markets (Wang et al., 2012). While traditional stock prediction models rely on historical data, there is growing evidence that stock markets are influenced not only by past information but also investor sentiment. With social media becoming an inseparable part of our daily life, sentiment expressed by investors on social media platforms can potentially shape and amplify market sentiment, ultimately influencing trading behavior and stock trend.

2 Literature Review

Existing studies have combined sentimental analysis with various deep learning methods for stock market prediction. Among these, the Long Short-Term Memory (LSTM) neural network is the most commonly used technique due to its ability to capture long-term dependencies through its memory function (Jin et al., 2019). In contrast, few studies have explored Convolutional Neural Networks (CNNs) for stock market prediction. While CNNs are primarily used for 2D image processing, they have also been applied to 1D financial data for stock prediction and have achieved high accuracy in Chinese and Thai stock markets (Chen and He, 2018; Sayavong et al., 2019). However, these two studies have not incorporated sentiment analysis into their models. Given the growing influence of online sentiment in stock markets, this project aims to bridge this knowledge gap by leveraging CNNs and sentimental analysis for predicting stock price movement of the world's top companies. Specifically, it seeks to answer the following research questions:

1. Does combining sentiment analysis with historical stock data improve the accuracy of stock price movement prediction?
2. How well do Convolutional Neural Networks perform in predicting stock price movement compared to the more commonly used LSTM models?

3 Datasets

This project utilizes two datasets from Kaggle:

1. **Stock market data:** This dataset contains historical stock market information from NASDAQ for five leading companies: Amazon, Apple, Google, Microsoft, and Tesla. It spans from 2015 to 2020 and includes daily opening and closing prices, high and low values, and trading volume for these stocks. For clarity in this report, variables included in this dataset will be referred to as stock features.
2. **Stock sentiment data:** This dataset consists of more than 3 million unique Twitter posts related to the same five companies from 2015 to 2020. The data includes the date of post, full text, and engagement metrics such as the number of comments, likes, and retweets for each tweet. These variables will be referred to as sentiment features.

4 Methodology

The Methodology of the project consists of four main components: sentiment analysis, data preprocessing, and the CNN and LSTM model architectures. This section details the implementation of each component.

4.1 Sentiment Analysis

Sentiment analysis is the first and a crucial part of this project, as its output is used for model trainings in the next stage. The tweet content in the stock sentiment dataset is analyzed using FinBERT (Araci, 2019), a natural language processing (NLP) model built on BERT and specialized in financial language. The model takes text of each tweet as input and assigns a numerical sentiment score — +1 for positive, 0 for neutral, -1 for negative — based on the highest predicted probability among the three categories. The sentiment score serves as a straightforward indicator of an individual's perception towards the associated stock. The sentiment scores are then aggregated by its date of the post and the associated company ticker to align with the stock market data on a daily level. In addition, preprocessing procedures are applied to each tweet to normalize the text content and reduce noise before processing with FinBERT. These steps include converting text to lowercase, removing URLs, mentions, hashtags, special characters, and punctuation. Moreover, tokenization is performed separately using FinBERT's own tokenizer to ensure consistency with the model's training process.[1]

4.2 Data Preprocessing

The data obtained from the original datasets and sentiment analysis need to be preprocessed to ensure it is suitable for reliable model training in the next stage. Since all numerical stock and sentiment features used for this project exhibit high skewness and significantly varying magnitudes, both log transformation and z-score normalization are applied to all numerical features. However, the sentiment score feature contains negative values, which preclude direct log-transformation. To address this, Yeo-Johnson transformation which accommodates negative values, is specifically applied to sentiment score.

4.3 Convolutional Neural Network

Two sets of CNNs are developed: one using only stock features, and the other integrating both sentiment and stock features. If the model that incorporates sentiment demonstrates better performance, it would provide evidence that investor sentiment contributes to improving stock price movement prediction. For the CNN model using only stock features, the input consists of lagged stock features from date $t-10$ to $t-1$. For the CNN model that combines stock and sentiment features, the input includes the same lagged periods, along with sentiment features at date t. Both models have the same objective: to perform binary classification that predicts whether the stock's closing price will increase or decrease (i.e., whether the difference between close_value at $t+1$ and t is positive or negative).

The baseline CNN begins with a 1D convolutional layer that takes the vector of aforementioned input features and applies 32 filters with a kernel size of 3. This is followed by a ReLU activation function and a max-pooling layer with a kernel size of 2 and stride of 2 to reduce the sequence length. After the convolutional and max-pooling layers, a fully connected layer is applied. Since the task is binary classification, the output of the fully connected layer passes through a sigmoid function, which will return 1 if the model predicts that closing price increase and 0 otherwise. The baseline CNN models are then extended by adding an extra convolutional layer. After the first convolutional layer and max-pooling layer, a second convolutional layer with 64 filters is introduced, followed by another max-pooling layer. The output of the second max-pooling layer is then passed to a fully connected layer and a sigmoid function.

[1] □ For this project, I aggregated all sentiment observations without filtering based on total engagement (i.e., the sum of tweet_num + like_num + comment_num), as proposed in the draft. Experiments were conducted using different total engagement thresholds ranging from 40 to 100. However, these thresholds did not lead to improved model performance compared to models that included all tweets regardless of engagement. Filtering based on engagement thresholds introduced missing data, which ultimately failed to address the initial concern about potential noises caused by including tweets with minimal public attentions. A detailed analysis of this experiment is available in model_training_engagement.ipynb.

4.4 Long Short-Term Memory

For systematic comparison with CNNs, two sets of LSTM models are constructed using identical lagged stock features and sentiment features at date t as inputs. The baseline LSTM implements a single-layer LSTM network that processes sequential stock data with a hidden size of 64, where each unit consists of cell states and hidden states that store and learn temporal patterns in the stock features (Ruke et al., 2024). The sentiment at date t (when applicable) is concatenated with the final hidden state and passed to a fully connected layer. As with the CNN architectures, the output is then passed through a sigmoid function for binary classification. Additionally, both CNN and LSTM models are trained using binary cross-entropy loss and 5-fold cross-validation to ensure robust performance measurement.

5 Results

The trained CNN and LSTM models are tested on the test set, and model performance is evaluated using prediction accuracy, precision, recall, and F1 score. Specifically, accuracy is defined as the number of observations correctly classified divided by the total number of observations. Precision measures the proportion of observations that are correctly predicted as 1 (i.e., stock price increase) among all observations predicted as 1. Recall measures the proportion of observations correctly predicted as 1 among all observations that are actually 1. The F1 score balances precision and recall by computing their harmonic mean, calculated using the following equation:

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

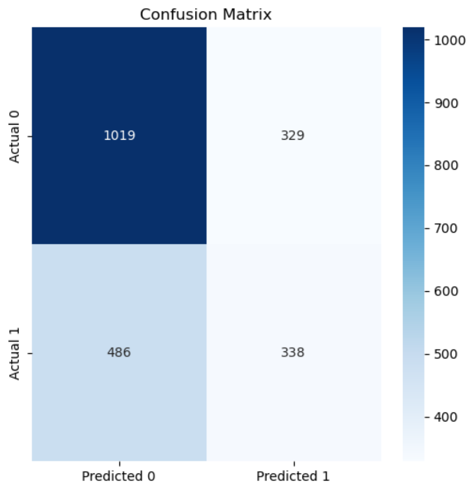
The F1 score is particularly useful for imbalanced datasets, where precision and recall generally cannot be both high or low at the same time, as is the case with the dataset used in this project. The resulting performance metrics for each model on the test set are presented in Table 1.

Table 1: Test Set Performance Comparison of CNN and LSTM Models

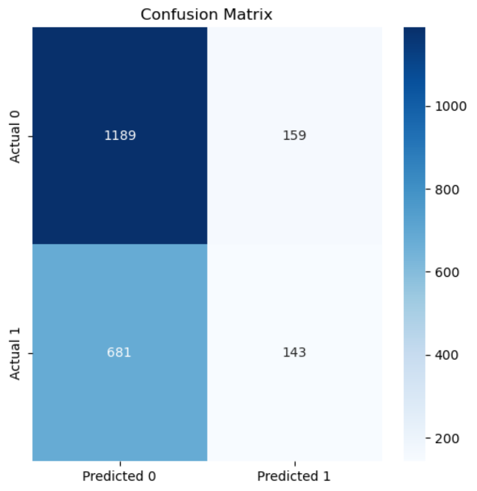
Model Type	Architecture	Features	Accuracy	Precision	Recall	F1 Score
CNN	Baseline	Stock-Only	0.6257	0.5873	0.0449	0.0834
	Baseline	Stock+Sentiment	0.6225	0.5244	0.0522	0.0949
	Two-Layer	Stock-Only	0.6248	0.5067	0.4102	0.4534
	Two-Layer	Stock+Sentiment	0.6133	0.4735	0.1735	0.2540
LSTM	Baseline	Stock-Only	0.6206	0.0000	0.0000	0.0000
	Baseline	Stock+Sentiment	0.6174	0.3704	0.0121	0.0235
	Two-Layer	Stock-Only	0.6206	0.0000	0.0000	0.0000
	Two-Layer	Stock+Sentiment	0.6215	0.5020	0.3095	0.3829

Note: All metrics reported on the test set.

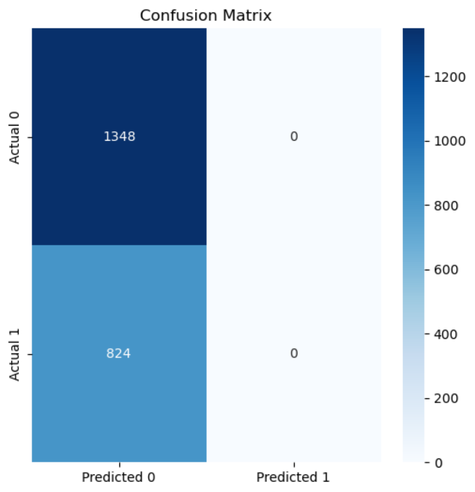
While both CNN and LSTM models achieve comparable accuracy around 62%, they generate different precisions, recalls, and F1 scores. The two-layer LSTM, when using only stock features, fails completely (precision = recall = F1 = 0). However, its sentiment-augmented counterpart achieves the highest F1 score of 0.3829 among all LSTMs. In contrast, the F1 score of the two-layer CNN drops by 44% (from 0.4534 to 0.2540) when sentiment features are incorporated. A similar pattern can be observed in the confusion matrices in Figure 1. In the imbalanced data setting, where the observations labeled as 0 (i.e., stock price decrease) is significantly greater than those labeled as 1 (i.e., stock price increase), CNN models are able to correctly predict more price increase cases when sentiment features are not included, compared to their sentiment-augmented counterparts. LSTM models, on the other hand, are able to correctly predict more price increase cases when sentiment features are included, compared to their stock-only counterpart. These patterns suggest that sentiment features help LSTMs recover meaningful signals from sequential stock data by providing complementary information about the stock market. However, the inclusion of sentiment features appears to distract CNNs from extracting stock price patterns, thereby not improving their performance in predicting stock movement.



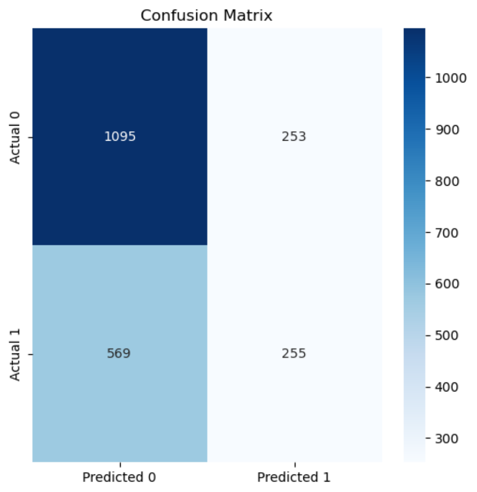
(a) Stock-Only CNN



(b) Stock + Sentiment CNN



(c) Stock-Only LSTM



(d) Stock + Sentiment LSTM

Figure 1: Confusion Matrix on Test Set

6 Discussion

The performance difference between CNNs and LSTMs may stem from their different approaches to processing noisy stock data. CNNs tend to outperform LSTMs on stock-only data because they treat lagged stock features as independent local patterns, rather than modeling explicit time-series relationships like LSTMs do. This can be advantageous in situations where stock market volatility disrupts temporal correlations between days. By ignoring these noisy dependencies, CNNs can more effectively extract stable short-term trends. However, LSTMs regain predictive power when sentiment features are added. Their gating mechanisms allow them to selectively amplify strong sentiment signals at time t, enabling them to override irrelevant volatility when sentiment provides a clear indication for price movement. As a result, LSTMs perform better than CNNs when sentiment features are incorporated.

The noisiness of stock data also highlights the first limitation of this project. Given the inherent volatility of stock data, models that incorporate volatility measures, such as standard deviation and average true range (ATR) of price fluctuations, may provide more valuable information for predicting stock price movement. Another important limitation is the imbalanced dataset. As observed in the results, both CNNs and LSTMs tend to bias predictions towards 0 (i.e., stock price decrease) rather than 1 (i.e., stock price increase), since the dataset contains more price decrease observations. While reweighting the loss function for model training is a potential approach to address this issue, experiments conducted indicated that it did not improve model performance compared to an unweighted loss function. However, future research can explore other balancing techniques, such as the Synthetic Minority Over-sampling Technique (SMOTE), or test the models in this project with more balanced datasets. Additionally, this project relied solely on the FinBERT model for sentiment analysis, without evaluating its performance in comparison to other NLP models. Future work can experiment with more NLP models, such as VADER (Gilbert and Hutto, 2014), which is designed for social media content like tweets, and assess whether sentiment predictions are consistent with those of FinBERT.

References

[1] Araci, D. (2019). FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. *ArXiv*, abs/1908.10063.

[2] Chen, S., & He, H. (2018). Stock prediction using Convolutional Neural Network. *IOP Conference Series: Materials Science and Engineering*, 435, 012026. `https://doi.org/10.1088/1757-899x/435/1/012026`

[3] Hutto, C., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1), 216–225. `https://doi.org/10.1609/icwsm.v8i1.14550`

[4] Jin, Z., Yang, Y., & Liu, Y. (2019). Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications*, 32(13), 9713–9729. `https://doi.org/10.1007/s00521-019-04504-2`

[5] Omer Metin. (2020, October). Tweets about the Top Companies from 2015 to 2020, Version 1.0. Retrieved March 6, 2025, from `https://www.kaggle.com/datasets/omermetinn/tweets-about-the-top-companies-from-2015-to-2020/data?select=Tweet.csv`

[6] Omer Metin. (2020, October). Values of Top NASDAQ Companies from 2010 to 2020, Version 1.0. Retrieved March 6, 2025, from `https://www.kaggle.com/datasets/omermetinn/values-of-top-nasdaq-copanies-from-2010-to-2020/data`

[7] Ruke, A., Gaikwad, S., Yadav, G., Buchade, A., Nimbarkar, S., & Sonawane, A. (2024). Predictive analysis of stock market trends: A machine learning approach. *2024 4th International Conference on Data Engineering and Communication Systems (ICDECS)*, 1–6. `https://doi.org/10.1109/icdecs59733.2023.10503557`

[8] Sayavong, L., Wu, Z., & Chalita, S. (2019). Research on stock price prediction method based on Convolutional Neural Network. *2019 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*, 173–176. `https://doi.org/10.1109/icvris.2019.00050`

[9] Wang, B., Huang, H., & Wang, X. (2012). A novel text mining approach to Financial Time Series forecasting. *Neurocomputing*, 83, 136–145. `https://doi.org/10.1016/j.neucom.2011.12.013`