Predicting Stock Market Trends from News Articles And Price Trends using Transformers

Stanford CS224N Custom Project, Mentor: Kaylee Burns

Kasra Naftchi-Ardebili
Department of Bioengineering
Stanford University
knaftchi@stanford.edu

Karanpartap Singh
Department of Electrical Engineering
Stanford University
karanps@stanford.edu

Team Contributions

Both members of the group contributed equally. Kasra designed most of the dataset pre-processing and filtering pipeline, while Karan focused on the model design, training, and evaluation. Both contributed to project conception and writing the final report, and all code, with the exception of the pretrained FinBERT and TinyBERT models, was written entirely from scratch.

Abstract

In this study, we introduce a novel approach for predicting stock market closing price on day 10, by integrating 9-day technical stock data with extensive financial news analysis using a transformer-based model, specifically leveraging the pretrained TinyBERT enhanced with Low-Rank Adaptation (LoRA). Unlike traditional methods that mainly utilize manual feature engineering and sentiment analysis of news headlines, our model processes the full content of news articles alongside corresponding raw stock market metrics. Specifically, 10 randomly selected news articles per stock ticker per day were tokenized and processed with FinBERT, and 6 financial metrics per stock ticker per day were analyzed using group-wise one-dimensional convolutional kernels. We show that this integration harnesses a more comprehensive set of data-driven insights for improved accuracy in market trend predictions. Our results indicate a notable enhancement in predictive performance compared to the baseline StockFormer model, achieving higher precision in directional accuracy (62.5% vs 57.66%) and improved metrics in normalized mean squared error (MSE) and R^2 values (0.9986 vs. 0.9956). This research demonstrates the potential of applying advanced NLP techniques to financial analytics and highlights the diminishing returns of traditional feature engineering techniques. With more powerful models, larger datasets, and longer news articles, it is not unreasonable to expect these models to predict stock market trends with very high precision.

Introduction

Predicting stock market trends has always been a challenging yet lucrative endeavor for investors and analysis. Common methods for modeling the stock market include technical analysis, or forecasting the direction of prices based on the study of past data such as price or volume, and fundamental analysis, which focuses on information surrounding the company underlying the stock. However, with the advent of large-scale machine learning and fast-moving NLP techniques, these techniques can now be automated and applied in tandem with the hope of producing better predictions.

Technical analysis relies on historical price and volume data to identify patterns and trends that can indicate future movements. Indicators such as moving averages, relative strength indices (RSI), and Bollinger Bands are often used to interpret market conditions. While these methods have proven effective to some extent, they primarily depend on past data, which may not always capture the full picture of market dynamics. Moreover, technical indicators are sometimes criticized for being overly simplistic and not accounting for the broader context that influences market behavior. The "efficient

market hypothesis" even suggests stock prices are entirely based on currently available information, and any price changes based on new information are inherently unpredictable.

On the other hand, fundamental analysis delves into the intrinsic value of a stock by examining factors such as company earnings, revenue, economic conditions, and industry trends. This approach aims to determine whether a company, and therefore its underlying stock, is overvalued or undervalued based on its financial health and growth prospects. Although fundamental analysis provides a deeper understanding of a company's potential, it can be time-consuming and subjective, relying heavily on the analyst's expertise and interpretation of qualitative information.

In this paper, we propose a novel approach that integrates technical stock data with comprehensive financial news analysis using a transformer-based model. Our method aims to address the limitations of traditional approaches by leveraging the full content of news articles rather than just headlines, utilizing a richer set of information to make more accurate predictions. Primarily, we expand on prior work in three ways: 1) leveraging full news articles rather than headlines, 2) learning features in these news articles predictive of stock trends rather than relying on simple sentiment analysis, and 3) using raw time-series quantitative metrics rather than human interpretations like running averages to allow the model to abstract its own relevant features.

Related Work

Past work in this area has largely relied on the aforementioned manual feature engineering on historical financial metrics, and/or sentiment analysis on news headlines. For instance, Kaeley et al. presented a transformer-based model that used finBERT-based sentiment analysis of news headlines to enhance prediction accuracy over extended time windows. The core of their work detailed the creation of a unique dataset (which they did not share) comprising manually-engineered daily technical indicators and news headlines for major stocks over nearly three years. Their model, termed StockFormer, achieved a directional accuracy (whether the model accurately predicted a rise or fall in stock value) of 57.6% with a lag time of n=9 (predicting the stock price on the 10th day given 9 days worth of articles and metrics), with a normalized MSE of 0.00466. They showed an increase in directional accuracy but a worsening of prediction accuracy / MSE as the lag time was increased.

In a similar vein, Phuoc et al. (2024) applied LSTMs to technical analysis indicators such as simple moving average and relative strength index to predicting stock prices for the Vietnamese market. Feeding in 60 days of data to forecast the stock price at the next day, they achieved an average accuracy of 93%. This lower accuracy could be attributed to the lack of news articles or external indicators outside of historical stock data in their approach.

Approach

Data Acquisition and Preprocessing

We sourced a large dataset of 306,242 financial news articles spanning approximately 5 months from reputable news outlets such as Bloomberg and CNBC (USA). Each article included metadata such as the publication time and the full article text. We filtered the articles for Meta, Apple, Amazon, Netflix, and Google to remove duplicates or headline-only articles and, using finBERT (Goo), removed any articles that had *neutral* sentiments. Our hypothesis was that neutral articles should not play a role in stock trends and therefore should be safely removed. All of this was done to extract the most high-quality data possible from the dataset and avoid spurious correlations. Next, we procured stock trends corresponding to these articles' timestamps from Yahoo Finance (Dow).

Over a batch of D consecutive days, we randomly selected 10 articles for each day, each padded to a maximum of 512 tokens, before passing them to finBERT to obtain the last hidden state, with size 512×768. Unlike the StockFormer model, we refrained from relying on sentiment analysis over news headlines only and instead abstracted the last hidden layer of finBERT over 512-token long articles. We had observed that headlines often included nothing substantive and a quick sentiment analysis over them would likely be superficial. In addition to these text embeddings of 10 financial articles per day, each day contained 6 raw quantitative metrics for that stock: open, high, low, close, adjusted close, and the total traded volume. Unlike StockFormer, we did not manually engineer any

features over these financial metrics and instead relied on series of group-wise (6 groups, one per metric) 1-dimensional convolutions with kernel size of 3. Our goal was to allow the model to learn the most meaningful embeddings rather than feeding it with limited feature engineering that made sense to humans. The final dataset was separated with an $80/10/10$ split into training, validation, and testing sets. These conditions resulted in 348 10-day training samples, 48 10-day validation samples, and 48 10-day test samples. All the code used in extracting, filtering, parsing, and loading the dataset was written from scratch.

Model Architecture

Whereas StockFormer, introduced in Kaeley et al., used a full transformer architecture with encoder and decoder models, we used a bidirectional encoder only. Specifically, we used the pretrained TinyBERT model, `huawei-noah/TinyBERT_General_4L_312D` (Goo). The StockFormer decoder received as input the closing price for the pat n days, so that in tandem with its encoder receiving news headline sentiments, feature-engineered metrics, and closing prices for the past n days, it could predict the closing price on day $n+1$. However, the past n-day closing price was being utilized both in its encoder and decoder networks. We found this unnecessarily costly and likely redundant, and were thus motivated to instead use a relatively lean bidirectional encoder model. We stacked linear layers and a `tanh` nonlinearity on the output of this efficient BERT model to predict the closing price for the 10th / last day. Moreover, we wrote a simple wrapper to apply Low-Rank Adaptation (LoRA) layers (Hu et al., 2021) over the TinyBERT parameters such that we could efficiently fine-tune them to our specific dataset. The details of our data preprocessing and model training are presented in Figure 1.

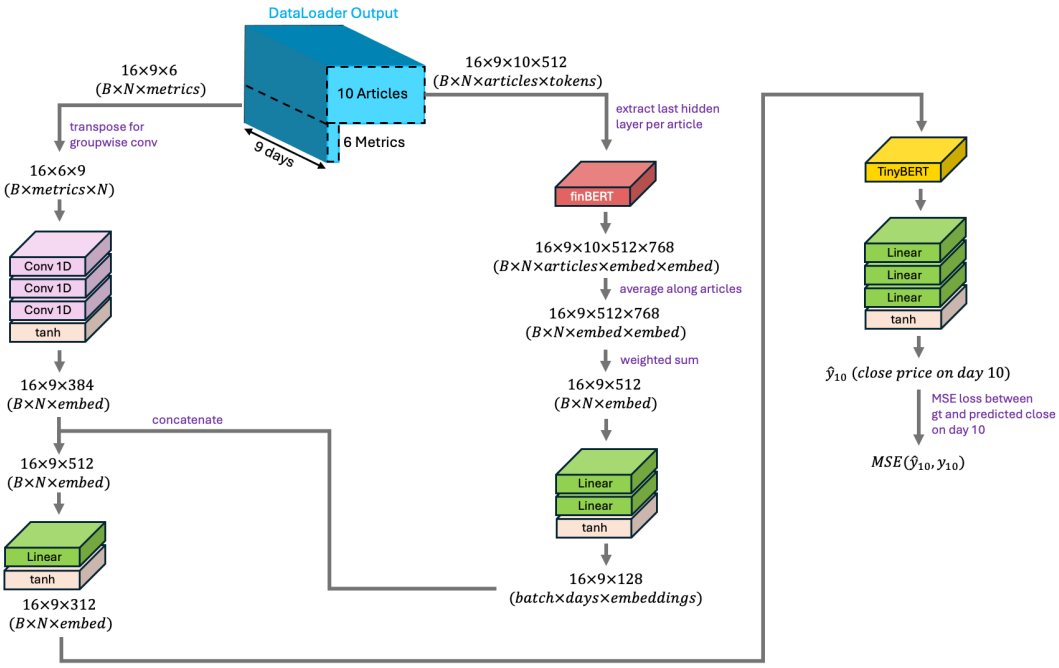

Figure 1: **Data Preprocessing and Training.** For each day, the 10 news articles are processed with finBERT and the last hidden layer is saved. The news output per day is a vector of length 128. The 6 financial metrics are independently convolved (setting `groups=6`) along the days dimension, and then concatenated with the news component to yield a final tensor of shape $16 \times 9 \times 512$, where 16 is the batch size, 9 is the number of days we are considering, and 512 is the final embedding dimension. This tensor is then passed through linear and nonlinear layers to output a tensor of shape $16 \times 9 \times 312$, compatible with TinyBERT input requirements. TinyBERT output was further processed through three linear layers and one `tanh` nonlinear layer, so to predict the closing price on day 10 ($\hat{y}_{10}$).

Baseline

The StockFormer model presented in Kaeley et al. was our primary baseline. Specifically, we aimed to beat this model's performance in normalized MSE, R^2, and directional accuracy, over a lag period of 9 days.

Experiments

Data

The final preprocessed **input** at each iteration of training was a tensor of shape $16 \times 9 \times 512$, corresponding to batch size, lag period in days, and embedding dimension. The embedding dimension captured the information on 10 financial news articles per day, plus abstracted information over 6 financial metrics per day. This tensor was passed through a linear layer as well as a `tanh` layer to reduce its last dimension to 312 so that it could be input to TinyBERT. We processed TinyBERT's output through three linear layers and one `tanh` nonlinear layer to predict a singular number for the closing price on day 10 ($\hat{y}_{10}$. Therefore, at every iteration, the final **output** was of the shape 16×1, which corresponded to the closing prices for a batch of 16 samples.

Evaluation Method

In order to compare our model against the StockFormer baseline, we reported MSE, R^2, and directional accuracy (did we accurately capture whether the price increased or decreased from the previous day). Additionally, we reported our average dollar amount difference, but since StockFormer does not report their error in dollar amounts, we could not compare the absolute accuracy of our model against StockFormer.

Experimental Details

We wrapped our models in LoRA, so that we could efficiently fine-tune the TinyBERT parameters to our task. Considering the additional layers and convolutional kernels added onto the model, our best performing model had 10,345,697 learnable parameters. The optimum learning rate started at 5×10^{-5}, and was halved at plateaus. We adopted 300 warm-up steps and 75 epochs, and used the Adam optimizer to update the model weights. In order to further increase efficiency, we used mixed precision in our custom `Trainer` class. Training time was about 40 minutes on a single NVIDIA A100 40GB GPU.

Results

Our best model was able to beat the baseline StockFormer model on every metric. Although we can confidently compare our R^2 and directional accuracies, we are not entirely certain our normalized MSE is directly comparable, as the baseline paper did not explain how they had normalized their MSE values. Our second best model, one that used a Huber loss to minimize sensitivity to outliers, also beat the baseline in normalized MSE and directional accuracy. Our results are presented in Table 1.

We had mixed anticipations regarding our model performance. On one hand, the baseline model used 3 years worth of data during training whereas our dataset only spanned 5 months. Therefore, we were concerned we had a very limited dataset for meaningful pattern recognition. On the other hand, we were confident in the advantages of our approach over the baseline, in terms of relying on convolutional kernels rather than feature engineering, as well as utilizing full articles as opposed to only the headlines. It appears that the strengths of our model outweighed the modest size of our dataset. This suggests that if we had a much larger dataset on par with the one StockFormer was trained on, we would quite likely beat the baseline by much wider margins.

Analysis

A major component of our architecture was the 1D convolutional layers that we applied to the six financial metrics in our dataset. To test the contribution of these layers, we first tried an ablation

Model	Parameter Count	Lag Period (days)	R^2	Normalized MSE	Mean Error (USD)	Directional Accuracy
StockFormer (baseline)	NA	9	0.9956	0.004659	NA	57.66%
Our Model with MSE Loss	10,345,697	9	0.9986	0.000113	$2.65	62.50%
Our Model with Huber Loss	10,345,697	9	0.9931	0.000520	$5.07	62.50%

Table 1: **Model Performance**. Our best model utilizing an MSE loss function beat the baseline on every metric. Our second best model using Huber loss function also beat StockFormer in directional accuracy as well as MSE error. Although the baseline model did not report any absolute dollar amounts, based on the other metrics, it is safe to assume their absolute dollar amount errors were worse than our models'.

study where we replaced them with linear layers. This model's performance was much worse than one including the convolutions, only achieving a 50% directional accuracy with a mean USD error over $6.

Next, to further examine what features these convolutional layers were learning, we visualized the convolutional kernels of the first layer for each financial metric as shown in Figure 2. We were able to isolate the effect of each financial metric because we had applied our convolutions on a group-wise manner, convolving each metric separately. Through such investigation, we found that these convolutional layers gave more weight to the open, close, and adjusted close price for each day, while the other metrics were considered to a lesser degree. This was both intuitive and informative. It was intuitive in that given the model was tasked with predicting the close price on day 10, it put more emphasis on examining the close price on days 1-9. It was informative in that analytical metrics such as True Range, Average True Range (Wilder Jr., 1978), Pivot Points (Kirkpatrick and Dahlquist, 2010), Donchian Channels (Covel and Ritholtz, 2017), and Stochastic Oscillator (Achelis, 2013) that rely on high and low price points are likely less significant in predicting the closing price, given the weight distributions we observe in Figure 2.

An additional ablation we ran involved freezing all parameters in TinyBERT, to evaluate the impact of attention and the transformer architecture on our model. The resulting model with only 554,681 parameters did even worse, producing a directional accuracy of only 45.8% (worse than random), with a USD error of $7.34. This result suggested that the addition of a learnable transformer model was crucial to the performance of our model.

Conclusion

In this study on predicting stock market trends using transformers and an extensive dataset of financial news articles, we have demonstrated advancements over traditional methods that largely depend on manual feature engineering and sentiment analysis of headlines. By integrating comprehensive data from full news articles and machine-abstracted financial metrics through convolutional kernels, we have addressed some of the inherent limitations found in previous methodologies.

Our approach diverges from conventional techniques by utilizing raw time-series financial data and the rich contextual information available in full-length news articles, rather than just headline sentiments. This allows our model to capture a broader spectrum of influential factors that might impact stock prices. The use of a transformer architecture, specifically the bidirectional-encoder-employing TinyBERT, enhanced the predictive capabilities of our system compared to the baseline model, StockFormer, as we showed though our ablation studies. Moreover, breaking free from traditional feature engineering methods such as computing relative strength index and moving averages, and instead relying on convolutional kernels, allowed the model to learn the most relevant information without the bias of pre-engineered features. These innovations were reflected in our results, where we achieved higher accuracy in terms of normalized mean squared error (MSE), R^2 values, and directional accuracy compared the the baseline.

One limitation is that we were working with a very small dataset spanning only about 5 months. As a result, we were unable to test our model performance over the larger lag times, such as $n = 29$, that our baseline had reported optimal for their setup. For the same reason, we were unable to replicate the StockFormer model and apply it to our specific dataset, as some of their feature-engineered metrics required long lag times that were not feasible with our small dataset.

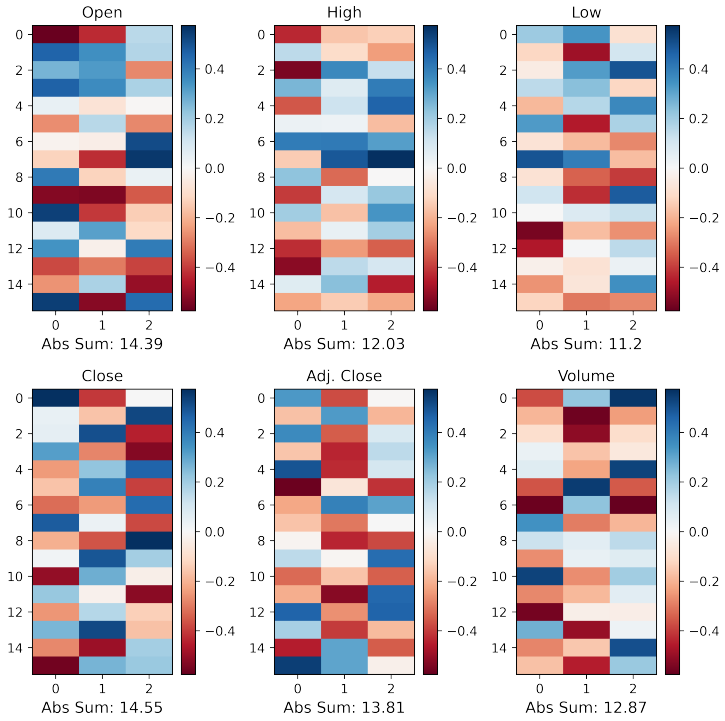

Figure 2: **1D Convolutional Kernel Weights.** Sum of absolute values of the weights for each of the 16 channels of the 1×3 convolutional kernel is shown for its corresponding financial metric. It is evident that *close* price, *open* price, and *adjusted close* price are upweighted compared to metrics such as *high* and *low* price.

Looking ahead, continuous improvement of our model's architecture and training processes is essential. Further research could explore the integration of other AI techniques, such as reinforcement learning, to enhance decision-making processes under dynamic market conditions. Additionally, expanding our dataset to include a more extensive range of financial products and longer time spans could further validate the robustness and applicability of our model.

Ethical Considerations

Our work merits the consideration of several ethical points. Firstly, our model's use of news articles relies on real-time data from reputable sources like Bloomberg and CNBC. However, we did not procure any licenses to use this data, relying on open source datasets that contained the articles we needed. This brings up the topics of copyright law and fair use, and larger scale work in this field might require data usage permission from the publishing sources.

Secondly, the application of NLP techniques that require large amounts of data and compute resources to predict stock market trends could potentially lead to unfair advantages for certain market participants who have greater access to these technologies. Ethical use implies that such technologies should not contribute to market manipulation or unethical trading practices.

Specifically on the last point, we would consider democratizing access by developing APIs that allow smaller market participants to access predictions at a reasonable cost. We would also aim to work closely with financial regulators to ensure that the use of AI in financial markets adheres to existing laws and ethical guidelines. If these efforts don't mitigate the risk of conferring unfair advantages to big hedge funds against other market participants, we may have no choice but to keep our model architecture restricted to a research setting, and require licensing agreements of those who wish to use it in their research and development teams.

References

Download historical data in Yahoo Finance | Yahoo Help - SLN2311.

google-bert/bert-base-uncased · Hugging Face.

US Financial News Articles.

Steven B Achelis. 2013. *Technical Analysis from A to Z, 2nd Edition*, volume 77. McGraw-Hill Education.

Michael Covel and Barry Ritholtz. 2017. Trend Following, 5th Edition. page 688.

Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *ICLR 2022 - 10th International Conference on Learning Representations*.

Harsimrat Kaeley, Ye Qiao, and Nader Bagherzadeh. Support for Stock Trend Prediction Using Transformers and Sentiment Analysis.

Charles Kirkpatrick and Julie. Dahlquist. 2010. Technical Analysis The Complete Resource for Financial Market Technicians.

Tran Phuoc, Pham Thi Kim Anh, Phan Huy Tam, and Chien V. Nguyen. 2024. Applying machine learning algorithms to predict the stock price trend in the stock market – The case of Vietnam. *Humanities and Social Sciences Communications 2024 11:1*, 11(1):1–18.

J. Welles Wilder Jr. 1978. New Concepts in Technical Trading Systems. *New Concepts in Technical Trading Systems*, page 130.