Application of Transformer-Based Model in Stock Price Prediction

Runqi Chen, Luyi Ge
New York University
rc5235@nyu.edu, lg3846@nyu.edu

Introduction

Deep learning has significantly impacted numerous fields, such as computer vision, NLP etc. However, its applications in the field of computational finance remain comparatively undeveloped (Arevalo et al. 2016). Our interest lies in deploying deep learning models, specifically Transformer-based models, for time-series prediction, which is a domain traditionally dominated by statistical methods and more recently by various machine learning models. The Transformer's capacity to capture long-range dependencies and interactions makes it particularly appealing for time series challenges in stock price prediction (Wen et al. 2023). Consequently, various modifications of the basic Transformer models have been developed to make it more suitable for time series modeling.

Throughout our project, we have experimented with multiple Transformer-based models including the vanilla Transformer, the Autoformer etc. and selected the best model based on performance metrics on our stock price dataset. We refined the model structure and fine-tuned the hyperparameters to finalize our model structure and training techniques. Then, given the trained networks, we explored the application of it by implementing an algorithmic trading strategy based on the prediction from the model and examine its profitability.

The rest of our report unfolds as follows: we would discuss existing papers relevant to our topic and emphasize the aspects incorporated into our project, then we would detail our dataset, the architecture of our finalized model, and the implementation details, in the end we would demonstrate the application of our model's prediction by executing our trading strategies to illustrate the potential practical deployment of our model.

Literature Survey

We conducted thorough literature survey around topics relevant to our project. Specifically, we focused on getting a deeper understanding about Transformer-based models, and time series Transformer models, applications on stock price predictions and methods and metrics suitable for this project. We initiated our survey with a fundamental review of the paper we learned from the course "Attention is All You Need" by Vaswani et al. (2017), a cornerstone text that introduced the Transformer model architecture. This paper was instrumental in understanding the structural and operational mechanisms of Transformers, including encoding, multi-head attention, and the integration of feed-forward and residual networks, which are pivotal for advancing machine learning applications in time series analysis. Building upon this foundational knowledge, we explored more recent advancements and applications of Transformer models in the field of time series analysis.

A significant contribution to our understanding came from Wen et al. (2023), who in their paper "Transformers in Time Series: A Survey," discuss enhancements to both the module and overall architecture of Transformers. Their research provides a critical analysis of the strengths and limitations of various Transformer applications in popular time series prediction scenarios. This study not only enriched our theoretical knowledge but also offered practical insights into optimizing performance for time series modeling.

Further enriching our survey, Wang and Yuan (2023) demonstrated the practical application of Hidden Markov Models (HMM), Long Short-Term Memory (LSTM) networks, and Transformer models on daily stock price predictions. Their findings highlighted the superior performance of Transformer structures compared to HMM and LSTM in this context, reinforcing the potential of Transformer-based models in handling complex financial datasets.

Additionally, Muhammad et al. (2023) provided an extensive overview of various Transformer models applied to stock price predictions across several markets. Their research outlined a robust pipeline for conducting stock price predictions using Transformer-based models, which includes sophisticated data processing techniques and metrics selection. The insights from this study were particularly valuable in enhancing our understanding of how Transformers can be tailored to solve specific issues in time series prediction and daily stock price movements. Leveraging methodologies from these studies, we experimented with our Transformer models, aiming to validate their effectiveness and optimize their application in predicting stock price trends. This literature survey has not only grounded our research in robust academic theory but also guided the practical aspects of our model development and testing.

Data and Input Features

Our raw dataset is the historical daily close price of Apple Inc (AAPL) fetched from Yahoo Finance from 01/01/2017 to 12/31/2023. The dataset is split into training set from 01/01/2017 to 06/30/2022 and the test set is from 06/30/2022 to 12/31/2023, an 80/20 split. The historical data is shown in Fig.1 below. Our input features are the previous 7 days' adjusted closing price. The features would be scaled before fed into the models, which would be described in detail in the later section.

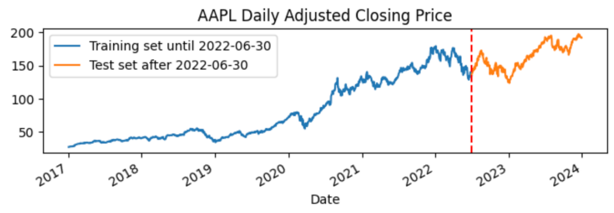


Figure 1: AAPL Adjusted Closing Price from 01/01/2017 to 12/31/2023.

Methodology

Model Structure and Parameters

The basic mechanism behind Transformer is introduced in "Attention is All You Need" by Vaswani et. al in 2017 as we learned in the course. The architecture can be generally summarized as encoder and decoder, multi-head attention, feed-forward networks, embedding and positional encoding etc., making it effectively capture the stock's volatile nature and the price's dependencies and interactions over timescales. Our finalized model structure is the modification of a time series Transformer model from Ntakouris (2021). We have tried various ways of modification proposed in the research from Wen et. al (2023) to make the model handle the times series data and regression task properly and to achieve desired performance in predicting future stock prices.

Our final architecture comprises a series of 5 Transformer encoder blocks, within each with a multi-head mechanism with 60 attention heads, and each attention head processes the data with an embedding size of 46 to facilitate feature extraction at each step of the sequence. The feed-forward networks within each Transformer block have an inner dimension of 55 to help the model to learn non-linear combinations of features through ReLU-activated convolutional layers.

Regularization techniques are employed to enhance the model's generalizability and prevent overfitting, crucial for maintaining robust performance across varying market conditions. Specifically, dropout rates are set at 14% within the Transformer blocks at these complex layers, and 40% in the multi-layer perceptron (MLP) that follows, together to prevent overfitting by omitting some of the feature detectors.

The MLP itself consists of a single dense layer with 256 units, providing capacity to integrate and synthesize the learned features into a predictive output. The model culminates in a linear activation function in the output layer, which is suitable for predicting stock prices, where the output needs to be a continuous variable reflecting the expected price. In total, we have 99760 trainable parameters.
Table 1 is a detailed model parameters summary:

Transformer Model Parameters	Choice
Embedding Size	46
The Number of Heads	60
Hidden Layer Size	55
Multi-Layer Perceptron Units	256
The Number of Transformer Blocks	5
Dropout Rates within Transformer Blocks	0.14
Dropout Rates within Multi-Layer Perceptron	0.4

Table 1: Transformer Model Details and Parameters

Loss Functions

Since MSE is sensitive to large errors and can be easily converted to the same units as stock price's, it is a common choice when evaluating the stock price prediction models. As suggested by the paper from Muhammad et al. (2023), due to the volatility of the stock price's daily change, large discrepancies need to be penalized when training the model. Since MSE can also be easily converted to the same unit as the stock price's and thus provide a straightforward interpretation of the loss, and it can also lead to a smooth gradient descent calculation, we select it as our loss function:

$$MSE = \frac{1}{m} \sum_{t=0}^{m} (\hat{y}_t - y_t)^2$$

Application of Trading Strategy

After experimenting with different parameters and model structures, we would implement an algorithmic trading strategy as an application of our trained model and evaluate its performances on the test dataset. The strategy largely depends on the predicted price and the actual price and simulate the real-world trading by incorporating factors such as the transaction cost and risk-free rate. The strategy is described as follows:

If the predicted price is higher than the actual price, we consider that as a signal that the stock is undervalued and the investor would buy one share; if the predicted price is lower than the actual price, it is overvalued, and the investor would sell one share.
The daily returns at day $t + 1$ is defined as:

$$r_{t+1} = s_t \times log(\frac{y_{t+1}}{y_t}) - c$$

where s_t denotes the action of buying or selling (+1/-1), y_t, y_{t+1} denotes the actual price at day t and day $t+1$ separately, and c denotes the transaction costs. We use the cumulative returns and the Sharpe ratios to evaluate the performance of this strategy where the cumulative return is just the sum of daily returns over the entire test time span. The Sharpe ratios is defined as follows:

$$SR = \frac{\mathrm{E[r]} - r_f}{\sigma(\mathrm{r})}$$

where E[r] is the expected return of the stock, r_f is the risk-free rate, and $\sigma(\mathrm{r})$ is the standard deviation of the return.

Implementation

Data Processing and Augmentation

As described before, we used a sequence of past 7 days' historical adjusted closing price as our parameters to predict next day's price following the sequence. Our choice of past 7 values is mainly from the consideration that it strikes a balance between detect meaningful patterns while making the model not sensitive to old, less relevant trends. Plus, as indicated in the case study from Muhammad et al. (2023), this number is closer to the real-world fact that stock market exhibits weekly cycles due to factors such as trading behaviors, institutional activities, and weekly economic reports.

Through experiments with normalized and raw data, we also decided to use *MinMaxScaler* from *sklearn* to normalize our price from 0 to 1, for it can preserve the the relative difference between data which could be an important indicator for financial data, and also for its special trait that for the unseen data which exceed the range during training, the scaler would output values outside [0,1] range, suitable for stock price prediction which might show great volatility, while unscaled data did not give desirable results.

We also randomly shuffled the data and use *validation_split = 0.2* to prevent overfitting. Final training data parameters has the shape of (1376,7,1) and test data parameters (378,7,1).

Selection of Hyperparameters

During the hyperparameters tuning process, we found that using Optimizer itself would not lead to smooth training process, and thus we selected parameters paired with warm-up and decaying learning rate scheduler which would be introduced later. Our final choice of hyperparameters applied in the training of both models is listed in Table 2:

Categories	**Choice**
Optimizer	Adam
Learning Rate	1.00E-04
Batch Size	20
The Number of Maximum Epochs	100
Early Stopping Patience	10

Table 2: Selection of Hyperparameters.

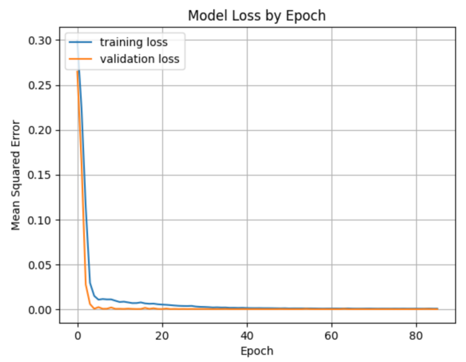
We used Adam optimizer paired with functions for warm up and controlled decay learning rate scheduler since Adam is suitable for deep learning problems on large datasets. Our learning rate scheduler for gradually increasing the learning rate and controlled decaying, making the training to stabilize given the complicated model structure during the start process and making finer adjustments to weights after the model starts to converge. We also forced early stopping after the validation loss stopped improving after 10 epochs to avoid overfitting problems. The initial learning rate is 1e-6, the base learning rate is 1e-3, and the minimize learning rate is 5e-5, with 30 as warm epochs.

Finally, the selection of batch size of 20 provides enough noise and leads to better generalization without losing efficiency. Although smaller batch sizes could potentially extend the training time per epoch and introduce instability, our use of a learning rate scheduler can help counterbalance this issue.

Combined these parameters, as shown in the training and test metrics in the results section, our choice of parameters led to a smooth training process with no sign of overfitting.

Result

Our training takes 86 epochs to stabilize and upon the completion of our training process, our model achieved a final training MSE loss of 0.00075 and validation MSE loss of 0.00015, which indicates an adept model design and parameter selection. Fig. 2 below shows the train and validation losses across epochs.



```
Final training loss: 0.0007470913697034121, validation loss: 0.0001524905819678679
```

Figure 2: Model Train and Validation Losses Over Epochs

From these figures, we can see that the training losses and the validation losses smoothly decreases over epochs, indicating that the model is effectively learning from the training data and converging towards an optimal solution and has a healthy degree of generalization capacity without substantial overfitting, indicating that the model has been effectively trained and the training hyperparameters work well.

Next, we evaluate the model's performance on the test dataset. Our model achieved a root mean squared error of 2.88 and consider our data fluctuates between a range of around 70, our model has a good learning and prediction capability, especially in capturing the movements within the given range without being too disrupted by the inherent noise and volatility. In Fig. 3 we present the actual price and predicted price on the test dataset.

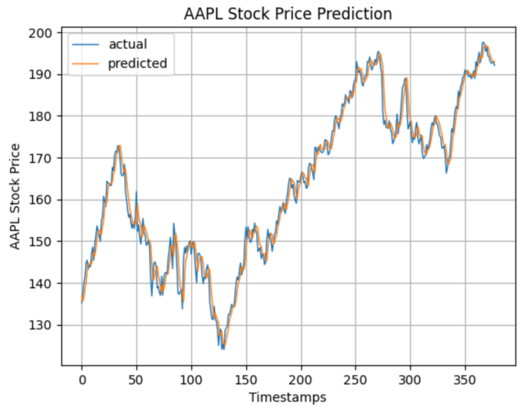


Figure 3: Actual And Predicted Stock Price Over Time On The Test Set

The final step in our project is to implement a trading strategy on our dataset using the predicted price and the actual price. Fig.4 shows the daily returns of our strategy. We achieved a mean daily return of 0.22 with a Sharpe ratio of 0.48, which further shows the model's predictive power and serves as an example of application on using the Transformer-based model in stock price prediction problems.

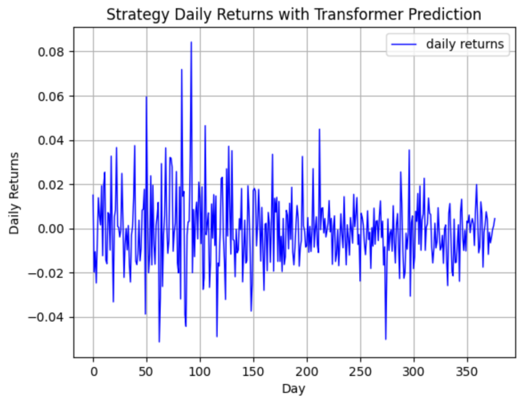


Figure 4: Actual And Predicted Stock Price Over Time on The Test Set

Conclusion

In this project, we have explored the utility of Transformer-based models in the domain of stock price prediction. Our project includes experimenting and modifying different variants of Transformer models and forming our own model structure and parameters, together with the fine-tuning of hyperparameters and training techniques, our finalized model showed smooth training curves and achieved desirable results. Through the process, we have enhanced our understanding about different Transformer models and time series prediction problems. We also demonstrated the practical application of our optimized model through the trading strategy example. The results underscore the potential of Transformer-based models to not only predict stock prices with high accuracy but also to drive profitable trading decisions in a real-world financial environment.

For this project, we originally intended to compare Transformer-based model structures' performances specifically in time series prediction and the application of model's prediction in the field of trading, and we generally achieved this goal. Should our project continue, we would incorporate other factors such as the trading volumes, market, and economic indicators etc. into our model's input features to make our model more delicate. We would also like to experiment with some novel but not widely used Transformer models in this field.

GitHub Repository

https://github.com/LobbyBear/DL_spring24_final_project

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I., 2023, "Attention Is All You Need," *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, arXiv:1706.03762v7 [cs.CL], August 2.
[2] Wang, Q., and Yuan, Y., 2023, "Stock Price Forecast: Comparison of LSTM, HMM, and Transformer," International Conference on Business Information Systems (ICBIS 2023), Advances in Human-Computer Systems, Vol. 14, pp. 126-136
[3] Muhammad, T., Aftab, A. B., Ibrahim, M.*, Ahsan, M. M., Muhu, M. M., Khan, S. I., and Alam, M. S., 2023, "Transformer-Based Deep Learning Model for Stock Price Prediction: A Case Study on Bangladesh Stock Market," International Journal of Computational Intelligence and Applications, Vol. 22, No. 3, pp. 2350013 (24 pages), World Scientific Publishing Europe Ltd.