A Hybrid LSTM-Transformer Approach for Financial Markets: Forecasting Stock Price Time Series

Shaohong Zheng *

School of Information Science and Technology, Beijing University of Technology, Beijing, 111000, China

* Corresponding Author Email: ericzheng0707@outlook.com

Abstract. Although the most used Long Short-Term Memory (LSTM) networks are now well-suited for short-term data changes, they are not enough to grasp the overall data in financial forecasting. This paper proposes a hybrid approach that combines LSTM and Transformer architectures to leverage both temporal dynamics and global dependencies in financial time series. Specifically, the model employs LSTM layers to extract sequential and local temporal features, while the Transformer's self-attention mechanism captures long-range correlations and the global structure of the data. Experimental results demonstrate that this combined framework achieves higher prediction accuracy and greater robustness compared to traditional models. This suggests that this paper model can better handle the complex, nonlinear, and highly volatile nature of financial data. Overall, this hybrid model provides a more reasonable and reliable basis for financial forecasting, offering valuable insights for investors, analysts, and policymakers seeking to make data-driven decisions in an ever-changing market environment.

Keywords: Stock price prediction; LSTM, Transformer; Hybrid model.

1. Introduction

Predicting stock prices has long been one of the most meaningful and tricky issues in fintech and quantitative investing. High-accuracy stock price predictions with the help of machine learning can directly help investors avoid investment risks, build better portfolios, and promote asset appreciation [1]. However, financial markets are a particularly complex system that always changes in non-linear irregular patterns. Stock price fluctuations are highly random because they are volatile and subject to a lot of uncertainties and noise disturbances. This is why predicting stock prices is particularly difficult, as foundational theories of market efficiency suggest [2].

To address this issue, early research primarily used traditional statistical models such as ARIMA [3]. However, as this method assumes that the data are linear, it's not reasonable and effective to predict complex market changes [4]. Soon after that when machine learning developed, nonlinear models that support vectors and random forests have emerged, like Support Vector Machines (SVM) for classification and regression [5] and Random Forests for ensemble learning [6]. Despite their strengths, these approaches often require substantial manual feature engineering and still exhibit limitations in modeling long-range, complex temporal dependencies in sequential data, a key challenge that has been widely noted in the literature [7].

In recent years, deep learning models have shown significantly better performance than traditional machine learning methods in stock prediction tasks. Among them, long short-term memory (LSTM) networks are the most prominent. [8]. However, LSTM models possess inherent limitations. Their sequential data processing mechanism makes it difficult to grasp the overall relationship well in different time periods, which means they are not paying enough attention to critical global information in making current predictions.

With deep learning techniques growing rapidly, the Transformer architecture has been built and recognized for its revolutionary impact by taking advantage of its self-attention mechanism [9]. The self-attention mechanism is characterized by the ability to pay attention to the relationship between elements in a serial sequence, so as to grasp the global context well. I found this method to be a good

choice for improving the accuracy of financial time series prediction, because it can handle more complex data relationships in a global sense.

Based on this idea, I tried an innovative architecture that combines LSTM with the Transformer model. This architecture aims to synergistically combine the strengths of both models to better discover the overall patterns and characteristics hidden in the data when processing temporal serial data.

For this study, I chose asset data from Amazon, Domino's Pizza, Bitcoin, and Netflix as the experimental dataset. These assets represent four distinct categories in the financial markets: traditional technology stocks, consumer goods, cryptocurrencies, and media companies. The effectiveness and robustness of the hybrid model in predicting complex financial data can be well verified on these four types of data, and it is shown that the method can remain stable under different circumstances.

2. Dataset and Methodology

2.1. Dataset and Preprocessing

The original dataset was obtained from the publicly available "AMZN, DPZ, BTC, NFLX adjusted May 2013–May 2019" collection on Kaggle. The dataset is a 1520-row, 5-column CSV document with one column of date information and four columns of price time series for different stocks. The four stocks included are Amazon (AMZN), Bitcoin (BTC), Domino's Pizza (DPZ), and Netflix (NFLX). The table records the daily closing prices of these four stocks from May 2013 to May 2019. I set the date column as an index and then split each stock into four separate tables, each shaped like (1520, 1). Figure 1 illustrates the general trend of the four stock price series. Table 1 provides detailed statistical characteristics of them.

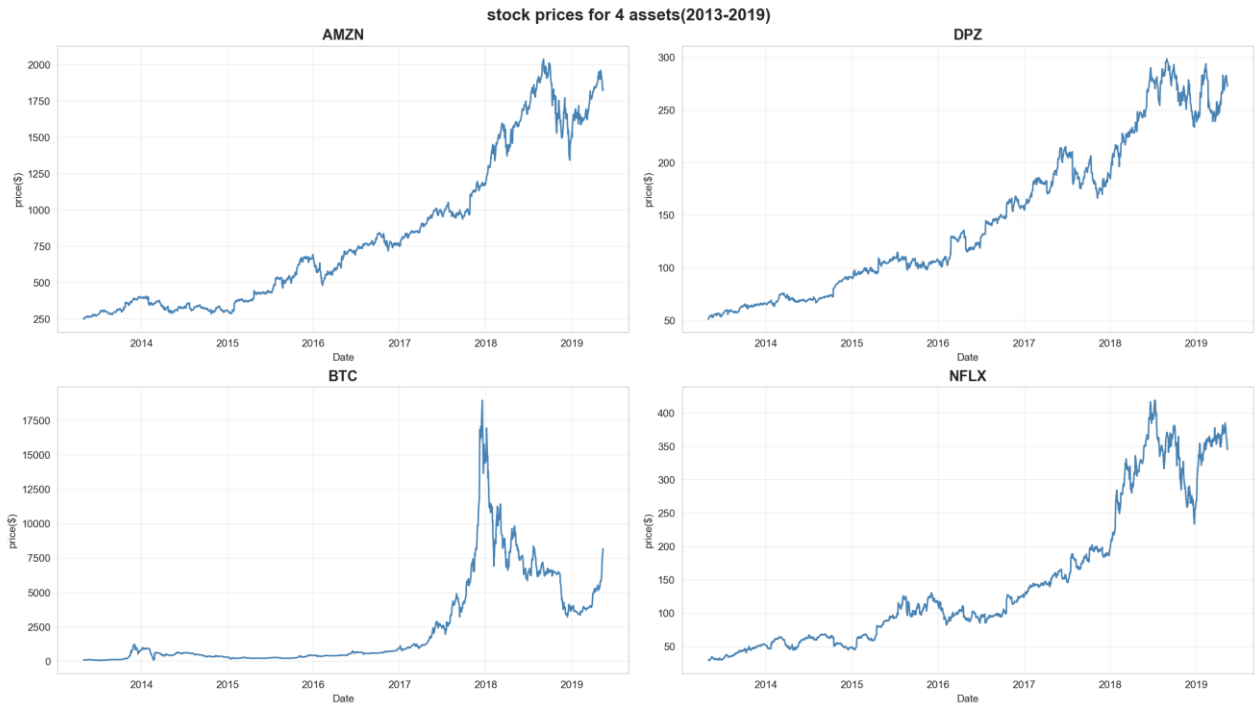

Fig.1 Visualization of Raw Data (Data from: AMZN, DPZ, BTC, NFLX adjusted May 2013–May 2019)

Table 1. Statistical Characteristics of the Data

	AMZN	DPZ	BTC	NFLX
count	1520	1520	1520	1520
mean	821.542	146.772	2421.466	147.668
std	518.443	72.192	3310.894	107.641
min	248.230	51.191	69.660	29.464
max	2039.510	298.636	18972.320	418.970

This study employs LSTM as the primary forecasting model; hence, the data will be fed into the model in chronological order. Since stock markets are closed on weekends, each calendar week is represented by five timesteps in this dataset. To maintain the authenticity of market trading patterns, no imputation will be performed for missing data on Saturdays and Sundays. Therefore, the time series is configured to consist of 25 timesteps, equivalent to five consecutive calendar weeks. The dataset for each stock is reshaped from an original size of (1520, 1) to a new dimension of (1495, 26), which means instead of containing only the closing price of a single day, each row now incorporates the past 25 timesteps of historical data, arranged in chronological order. The data is then normalized to the range (-1, 1) using the MinMaxScaler from the sklearn library.

Then, the input (X) and the target output (y) are split. In this study, the LSTM model predicts the current day's price based on the previous 25 days' prices. Therefore, the first column of the processed data table, which represents the current day's price, is designated as y, while the subsequent 25 columns are used as the input features X. At this stage, the shape of y is (1495, 1), while the processed input X has been reshaped into dimensions of (1495, 25, 1). This transformation ensures that X satisfies the input shape requirement of the LSTM model, which follows the format (batch size, timesteps, input dimension). The segmented inputs and outputs are loaded into a DataLoader, with the first 90% allocated to the training set and the remaining 10% to the test set. A batch size of 128 was selected for this study based on hardware constraints.

2.2. Hybrid LSTM-Transformer Model

In the initial part of this study, employ the Long Short-Term Memory (LSTM) network, a gated recurrent architecture renowned for its ability to model temporal dependencies, to tackle the challenge of stock price forecasting. As shown in Figure 2, x_t is the input vector at time step t, h_{t-1} is the previous hidden state, and h_t is the current output. The input gate i_t controls how much new information from x_t enters the cell. The forget gate f_t decides how much of the previous cell state c_{t-1} to retain. The cell state c_t carries long-term memory through the network. The output gate o_t determines which part of the cell state is output as h_t [10]. In financial scenarios, the forgetting gate dynamically discards outdated market information, the input gate incorporates fresh market data, and the output gate generates predictions based on the processed market state. This adaptive memory control allows the LSTM model to filter market noise and capture the underlying nonlinear trends in historical price sequences, a capability where traditional statistical models often fail [11]. This study utilizes the PyTorch library to construct the LSTM model in a Python environment.

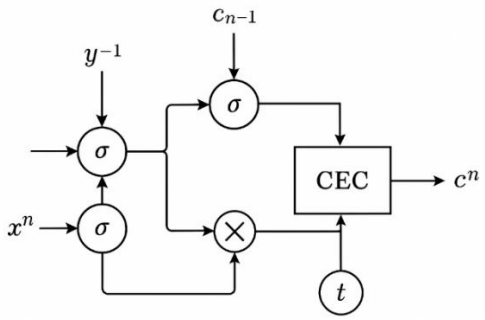

Fig.2 Architecture of LSTM memory cell [10]

The structure of the model consists of a nn.LSTM module as the foundation, followed by an nn.LayerNorm layer is used to standardize the output of hidden state. Then, connect a Dropout layer in front of the final linear output layer to reduce overfitting. The model has only one input dimension and one output dimension, and this nn.LSTM module consists of two layers, each containing 128 neurons.

Initially, the model's performance wasn't satisfying. This is because the model only passes the last time step of the LSTM layer output directly to the linear output layer. This does not take into account all other outputs of the entire time series, limiting its ability to effectively learn from past patterns. Therefore, the model incorporates MeanMax pooling into the optimization, allowing it to retain general historical information while effectively capturing the current salient features. Of course, correspondingly, the input dimension of the linear output layer is doubled to accommodate the processed output.

However, after a new round of training, the model's predictions are still not good enough, showing significant lag. Especially, for stocks with high noise and strong volatility such as AMZN, the R-squared metric occasionally drops below 0.65, reflecting limited predictive accuracy. Therefore, during further optimization, a self-attention mechanism from the Transformer architecture was integrated into the model. This enables the model to establish direct and precise dependencies on information from any timestep. In other words, the enhanced LSTM with Transformer gains a global perspective over the entire time series. The architecture of transformer model is shown in Figure 3, which demonstrates how the self-attention mechanism works.

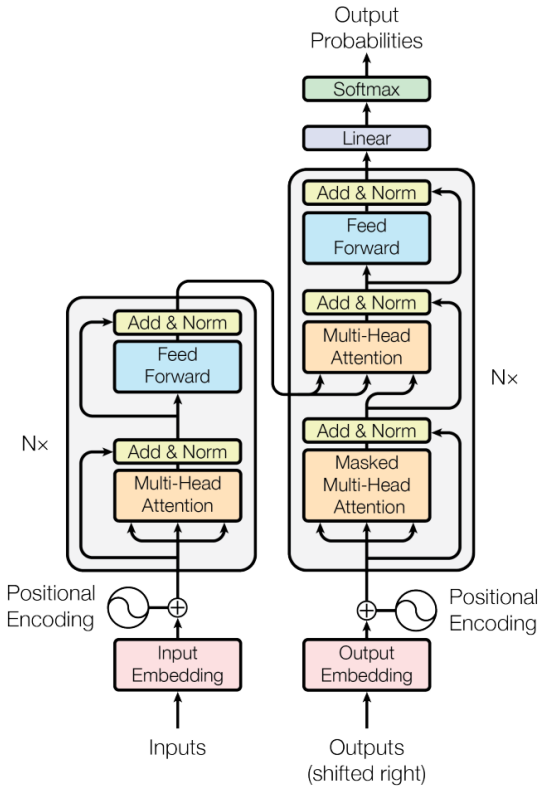

Fig.3 Transformer model architecture [9]

The optimized model incorporates an nn.Transformer layer, which takes as input the normalized outputs from the LSTM layer. This transformer layer is configured with 4 attention heads, 128 neurons, and consists of 2 encoder layers and 2 decoder layers. Thus, the original LSTM layer functions similarly to an encoder. In this hybrid LSTM-Transformer architecture, the LSTM module extracts temporal features from the input sequence and encodes them into a hidden representation, which is subsequently passed to the Transformer for further processing [12].

When the feature at the last timestep of the LSTM output may fail to capture comprehensive information from the entire sequence, pooling operations can help aggregate global patterns. However, in this implementation, incorporating the nn.Transformer module, the self-attention mechanism inherently captures contextual relationships across the full sequence. As a result, the model automatically focuses on the most relevant information throughout the input sequence without relying on hand-designed pooling. Therefore, the model no longer requires explicit Mean-Max pooling; it is sufficient to take the output from the final timestep of the Transformer.

In summary, the final model architecture is structured as Figure 4: the input first passes through an LSTM layer to capture temporal features, which are then normalized and fed into a Transformer layer to obtain contextually enriched representations through self-attention. Finally, after dropout regularization is applied, the output is produced by a linear layer.

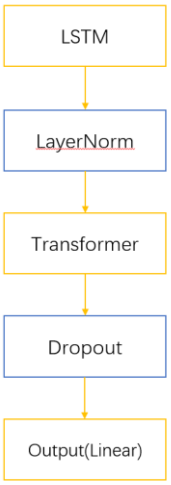

Fig. 4 Final model structure (Picture credit: Original)

3. Implementation and Analysis

3.1. Experimental Setup

The experiment was conducted in a Python 3.13.5 environment for model construction, training, and evaluation. The PyTorch library (version 2.9.0. dev20250802+cu129) was used for model implementation, with CUDA version 12.9 enabled for GPU acceleration. Additional libraries including pandas and numpy were employed for data reading and manipulation, matplotlib.pyplot for visualization, sklearn.preprocessing for data normalization, and pickle for saving scaling parameters.

All code was developed, compiled, and executed in Visual Studio Code (version 1.103.2). Model training was performed on an NVIDIA GeForce RTX 5080 GPU.

3.2. Results and Comparison

This study conducted ablation experiments by applying two model architectures—a pure LSTM model and a hybrid LSTM-Transformer model—to predict the four selected stocks on the test set. Results of the experiments are shown in Table 2.

Table 2. Comparison of the Predicting Ability of the Two Models

Stock name	Adjusted R^2 with pure LSTM model	Adjusted R^2 with LSTM-Transformer model
AMZN	0.52782	0.81018
DPZ	0.80254	0.83298
BTC	0.93866	0.94315
NFLX	0.92270	0.91482

These results in Table 2 will be analyzed according to data characteristics.

Category 1 includes data with high noise and strong volatility, such as AMZN and DPZ. On this data, the optimized LSTM-Transformer model shows significantly stronger prediction ability than the pure LSTM model. The adjusted R^2 results show that the LSTM-Transformer model reaches 0.81018, while the pure LSTM model is only 0.52782. DPZ stock data shows relatively low volatility compared to AMZN. At this time, the adjusted R^2 value of the pure LSTM model is 0.80254, and the optimized LSTM-Transformer model reaches a relatively high 0.83298.

These results further suggest that the optimized hybrid model can provide more accurate and reasonable predictions in scenarios where pure LSTM does not perform well.

Category 2 includes relatively low-noise financial data, such as BTC and NFLX. For Bitcoin and Netflix stock price predictions, the pure LSTM model has already shown strong performance, with adjusted R^2 values of 0.93866 and 0.92270, respectively. Still, on this type of data, the LSTM-Transformer model still delivers excellent performance, with adjusted R^2 values of 0.94315 and 0.91482, respectively, which is satisfactory enough.

To sum up, based on the prediction results of these four stocks, the LSTM-Transformer architecture not only maintains the strong prediction ability of the pure LSTM model on low volatility and low noise data, but also has a better and more convincing performance on high volatility and high noise data. It can be concluded that the LSTM-Transformer model improves the performance of the LSTM model and enhances its versatility, providing greater practical value for financial data forecasting.

3.3. Visualized Analysis

The predictive results of both models will now be visualized to corroborate the preceding conclusions. In the following figures, the actual stock price movement on the test set is represented by a black line, the predictions from the pure LSTM model are shown in blue, and the LSTM-Transformer model's predictions are displayed in red.

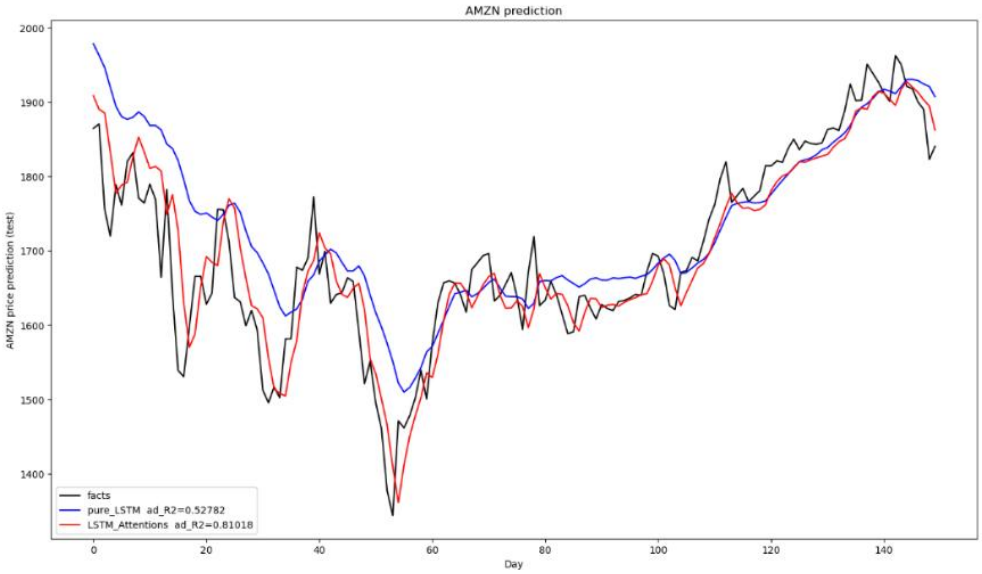

Fig.5 Predictive Performance of Amazon Stock Price on the Test Set (Picture credit: Original)

Figure 5 clearly demonstrates the superior predictive performance in terms of AMZN stock data of the LSTM-Transformer model (red line) compared to the pure LSTM model (blue line), as reflected by the R^2 metric.

Fig.6 Predictive Performance of Domino's Pizza Stock Price on the Test Set (Picture credit: Original)

Figure 6 reflects a slight advantage of the LSTM-Transformer model on DPZ stock data predicting. Although the improvement is less pronounced than in the AMZN prediction case, it still represents a measurable enhancement in forecasting capability. By visualizing the predictive capacity, it can be further concluded that the advantage of the LSTM-Transformer architecture becomes more pronounced as the forecasting difficulty of the stock data increases.

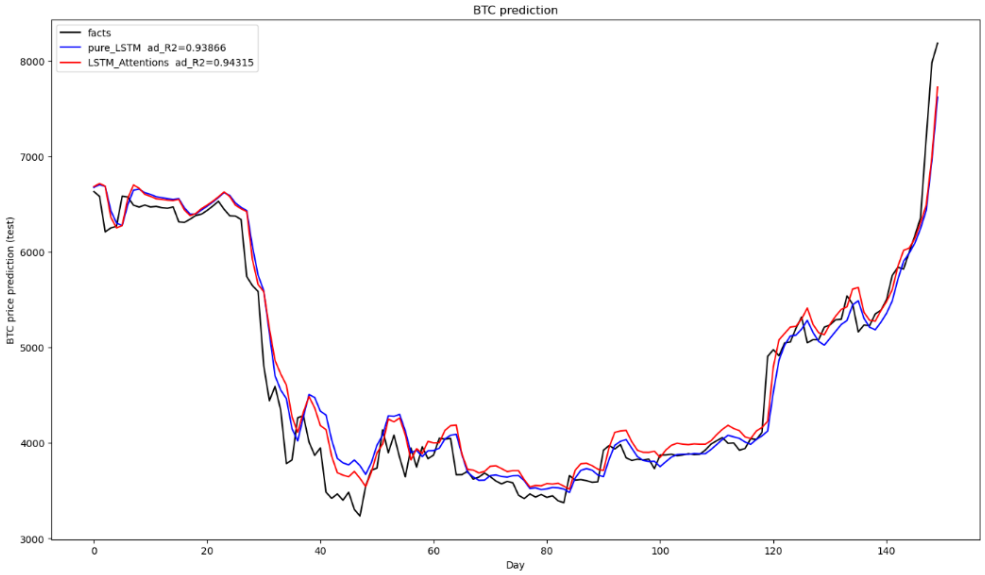

Fig.7 Predictive Performance of Bitcoin Stock Price on the Test Set (Picture credit: Original)

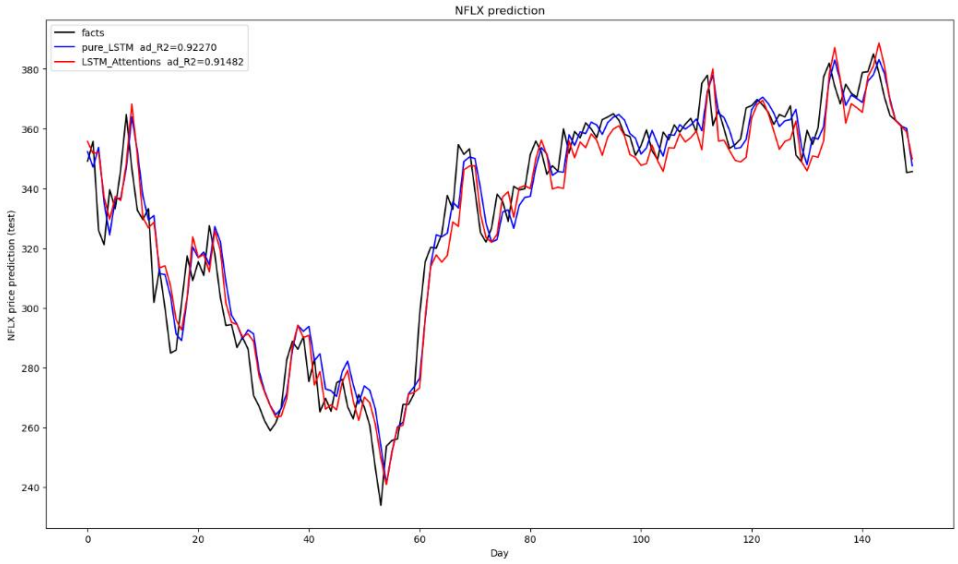

Fig.8 Predictive Performance of Netflix Stock Price on the Test Set (Picture credit: Original)

Figures 7 and 8 verify that the LSTM-Transformer model essentially retains the predictive performance of the pure LSTM approach.

4. Conclusion

This paper mainly examines a hybrid model that combines LSTM and Transformer to predict time series changes in financial data. Experiments have found that the predictive effect of this hybrid model is much better than using the LSTM model alone, especially in the evaluation index of R^2. The reason for this improvement is that the model plays out the advantages of both structures, LSTM is good at processing time-dependent relationships and long-term memory, and Transformer's self-attention mechanism is good at capturing the overall characteristics of the entire sequence. The model's stability in the complex market environment with lower predictability, higher noise levels, and greater volatility is also verified.

While this study has yielded positive outcomes, several limitations remain to be addressed. First, the current model relies solely on historical price data (closing prices) for prediction. This may limit its generalizability and ability to fully respond to fluctuations driven by other market factors. Second, the incorporation of the Transformer module increases model complexity, posing challenges to computational efficiency. This will be a more serious problem when the scale becomes larger and more diverse.

To address these limitations, more types of input data can be added. In addition to those now used, some technical indicators such as RSI and MACD can be added, as well as macroeconomic data such as interest rate and inflation, and even the views of the market on the news and social media can be added to the market situation, so that you can get a more complete picture of the market situation. In terms of model structure, some improved and specially designed versions of the Transformer model should be considered, such as Informer or FEDformer, which may probably be more efficient when processing long-term sequence financial prediction. What's more, expanding the model's predictive dimension beyond single-point closing price forecasts could surely be helpful to the predicting ability.

Ultimately, this study confirms the significant potential of hybrid deep learning models in handling high-noise, non-stationary financial data. In summary, the LSTM-Transformer model significantly improves forecasting performance by synergistically combining temporal memory encoding with self-attention mechanisms. This hybrid approach not only provides an effective solution for financial

time series prediction but also establishes a solid foundation for developing more accurate and robust financial forecasting tools in future research.

References

[1] Gu S, Kelly B, Xiu D. Empirical asset pricing via machine learning. The Review of Financial Studies, 2020, 33(5): 2223-2273.

[2] Fama E F. Efficient capital markets: a review of theory and empirical work. The Journal of Finance, 1970, 25(2): 383-417.

[3] Box G, Jenkins G M. Analysis: forecasting and control. San Francisco, 1976.

[4] Cont R. Empirical properties of asset returns: stylized facts and statistical issues. Quantitative Finance, 2001, 1(2): 223.

[5] Cortes C, Vapnik V. Support-vector networks. Machine Learning, 1995, 20(3): 273-297.

[6] Breiman L. Random forests. Machine Learning, 2001, 45(1): 5-32.

[7] Lim B, Zohren S. Time-series forecasting with deep learning: a survey. Philosophical Transactions of the Royal Society A, 2021, 379(2194): 20200209.

[8] Sezer O B, Gudelek M U, Ozbayoglu A M. Financial time series forecasting with deep learning: a systematic literature review: 2005–2019. Applied Soft Computing, 2020, 90: 106181.

[9] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. Advances in Neural Information Processing Systems, 2017, 30.

[10] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735-1780.

[11] Fischer T, Krauss C. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 2018, 270(2): 654-669.

[12] Kabir M R, Bhadra D, Ridoy M, et al. LSTM–transformer-based robust hybrid deep learning model for financial time series forecasting. Science, 2025, 7(1): 7.