Enhancing Financial Time Series Prediction with LSTM Neural Networks

Authors: Xi Zheng, Peiyan Zou, Guanhao Chen

Instructors: Antoine Jacquier, Lukas Gonon

Date: September 26, 2024

Abstract

This study tries to examine and refine the usage of ensemble Long Short-Term Memory (LSTM) networks to predict stock price movements in the Dow Jones Industrial Average (DJIA) constituent stocks from January 1996 to August 2024. Daily returns in DJIA are classified for binary prediction. Ensemble bidirectional LSTM model is used to train for the dataset. The portfolio return based on the refined model is calculated to display the profitability. The results imply difficulties in model adaptation and underscore the need for further refinements in feature selection and data utilization.

1 Introduction

In financial forecasting, accurately predicting market trends and asset prices remains a significant challenge due to financial time series data's inherent complexity and volatility. As a specialized type of Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) neural networks have emerged as a powerful tool in this domain. Unlike traditional methods, LSTM networks are designed to capture long-term dependencies in sequential data, making them particularly well-suited for modeling time series where historical information is crucial for predicting future values. By leveraging LSTM's ability to maintain and utilize long-term memory, financial analysts can improve the accuracy of their forecasts, allowing for more informed investment decisions and risk management. This report explores the application of LSTM networks in enhancing financial time series predictions, focusing on methodological improvements and comparative analyses with other models.

1.1 Research Problem

Building upon the ensemble LSTM approach outlined by Fjellström (2022), this study aims to determine whether the ensemble LSTM method can be effectively applied to the DJIA (Dow Jones Industrial Average), a stock index similar in number of components to the OMXS30 index used in the original study. This research is going to figure out whether the ensemble LSTM methodology can be adapted and improved for this new dataset.

1.2 Purpose

The purpose of this research is to assess the applicability of the ensemble LSTM method to a different stock index and to seek for potential refinements to the original methodology to enhance predictive performance. Additionally, this research will adopt an investment strategy based on the LSTM model to evaluate its profitability.

1.3 Literature Review

RNNs are able to process past and present data to learn and predict future patterns, but RNNs have a big drawback, the gradient descent process RNN loses its long-term memory. To solve this problem, Hochreiter and Schmidhuber (1997) proposed the LSTM networks.

The study by Fjellström (2022) investigates the use of Long Short-Term Memory (LSTM) networks for predicting stock price movements. Fjellström's study employs an ensemble LSTM approach, which is inspired by Barra et al. (2020), to improve prediction accuracy, and reports that the accuracy of stock movement predictions ranges between 49% and 55%. He compares an LSTM-based portfolio against randomly chosen and market-wide portfolios to assess model performance, evaluating profitability and Sharpe ratios. Despite moderate prediction accuracy, the ensemble LSTM method demonstrates potential advantages in portfolio management and risk-adjusted returns. Our research will build on this by applying and refining the ensemble LSTM model using the components of DJIA index and evaluating the profitability of the refined model.

2 Long Short-Term Memory

Following the explanation in "Detailed explanation of LSTM parameters in PyTorch" (2021), Figure 1 shows an LSTM network consists of an input feature x, a hidden layer containing multiple units, and an output y. Each LSTM unit contains three gates: a forgetting gate, an input gate, and an output gate, which control how much of the memory is retained, forgotten, or output, respectively.

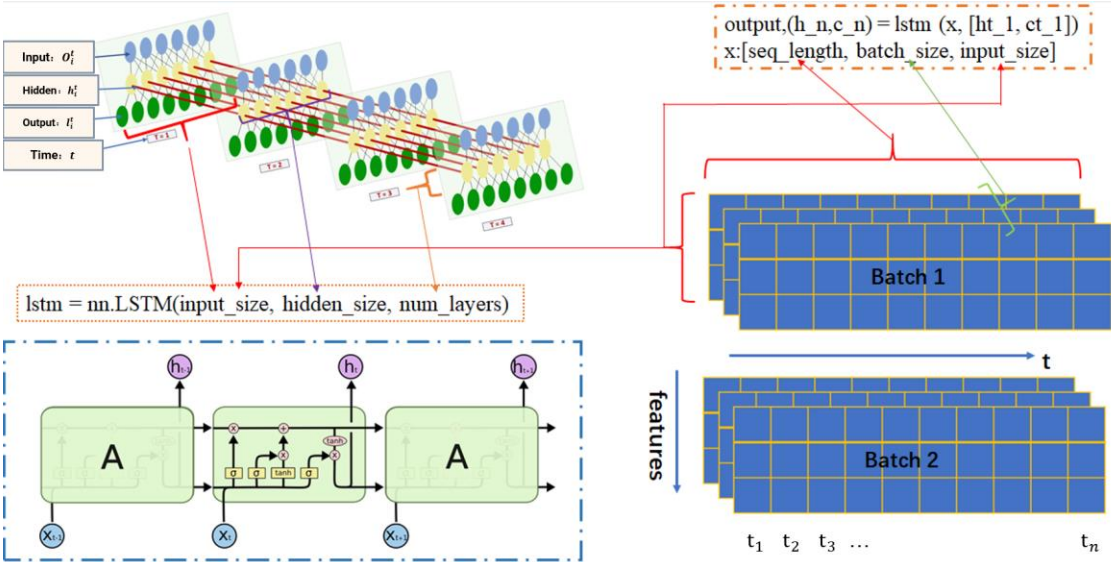

Figure 1: LSTM network with a magnified LSTM unit (memory cell)

In figure 2, parameter and symbol definitions:

x = (x1, x2, ... , xn) is the input vector, denotes the input data with sequence length n. st is the cell state, the memory of the cell at time t. s̃t is the candidate state at time t. ht is the hidden state, also the output of the cell. Forgetting gate ft and input gate It determines how much old information is retained and new input information updates in memory. Output gate ot controls the hidden state ht of the output of the current time step. All gates use a Sigmoid activation function σ, whose output value ranges from 0 to 1, with 0 indicating that no information passes and 1 indicating that all information passes.

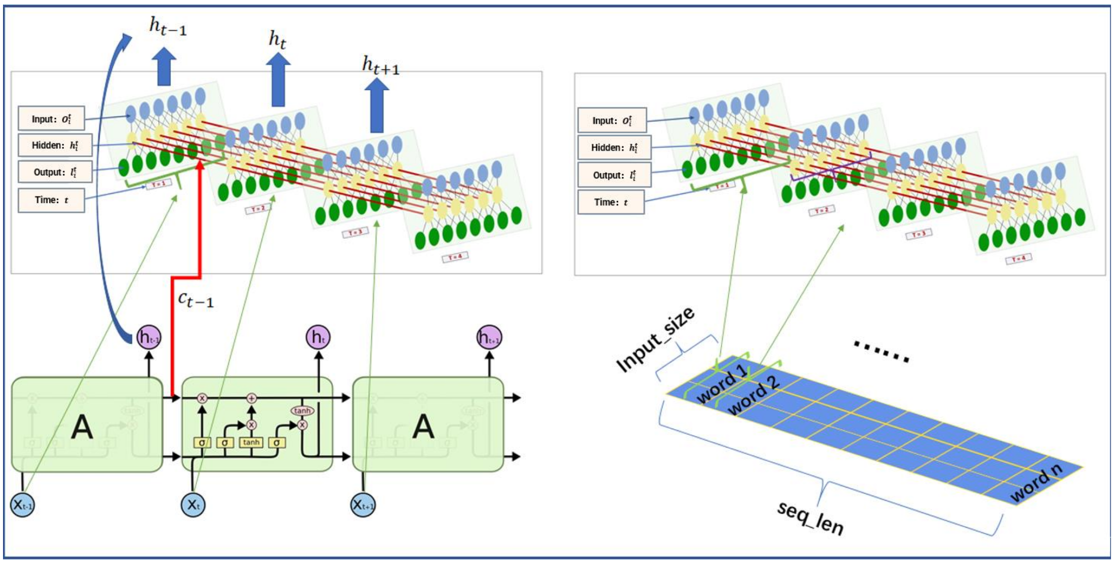

Figure 2: LSTM network parameters and symbols

The LSTM processes the sequence data sequentially through each time step, passing the input vector xt to each cell and calculating the output values of the three gates based on the current input and the previous hidden state ht-1. Through these gates, the LSTM decides how much of the old memory (cell state ct-1) to keep and how to update the current cell state st. Eventually, through the output gates, the new hidden state ht is generated as an output.

3 Method

3.1 Data

3.1.1 Data Collection

Historical stock price data is sourced from Yahoo Finance. This dataset includes daily closing prices and other financial indicators for the DJIA (Dow Jones Industrial Average) index, providing a comprehensive basis for model application and evaluation. The Dow Jones Industrial Average (DJIA) is a stock index of 30 major U.S. companies from various sectors, including financials, technology, consumer goods, and healthcare. It features industry leaders like Microsoft, Disney, and McDonald's, making it a key indicator of the U.S. industrial sector and overall economic health. Like the Stockholm 30 Index (OMXS30), which also consists of a limited number of constituent stocks, the DJIA has fewer component stocks but represents a larger and more diverse market, reflecting a greater overall market capitalization.

3.1.2 Data Preprocessing

Following Fischer (2018) and Fjellström, stock returns were processed by calculating daily medians. Each stock's daily return was classified as 0 if below the median or 1 if above. Sequences of these binary classifications were then created, with the target being the one-day-ahead prediction of whether the return will be above or below the median. Figure *3* illustrates this.

Date	AAPL	AMGN	...	WMT	Median	label_AAPL	label_AMGN	...	label_WMT
1996/1/3	0.00000	-0.01073	...	0.01075	0.00000	0	0	...	1
1996/1/4	-0.01751	-0.04338	...	0.01064	-0.00826	0	0	...	1
1996/1/5	0.08515	0.02268	...	-0.01579	0.00000	1	1	...	0
1996/1/8	0.01095	-0.01663	...	0.00000	0.00278	1	0	...	0
⋮	⋮	⋮	...	⋮	⋮	⋮	⋮	...	⋮
2024/8/29	0.01457	0.00461	...	0.00447	0.00574	1	0	...	0
2024/8/30	-0.00344	0.00852	...	0.01060	0.00769	0	1	...	1

Figure 3: Input sequences and targets

3.2 Network Architecture

The LSTM model consists of one input layer, one hidden Bidirectional LSTM layer, and one output layer. A sigmoid activation function is applied in the output layer, which represents the model's confidence in the predicted outcome. Predictions closer to 1 indicate higher confidence that the stock return will be above the median, while predictions closer to 0 suggest confidence that the return will be below the median. With the help of Nogueira's (2014) Bayesian optimization algorithm to determine the optimal value of the hpyerparameters in the model, We use Adam optimizer with a learning rate of 0.08. The hyperparameters were fine-tuned for optimal performance as follows:

·Number of neurons in the hidden LSTM layer = 25

·Dropout = 0.4

·Recurrent dropout = 0.1

·Batch size = 6800

3.2.1 Training and Testing Strategy

The dataset for each stock is divided into training, validation, and testing sets, with the test set also serving as the validation set during training. Stock data sequences are fed into the LSTM model, with each sequence having a length defined by look_back, which in our case is 30 days.

During testing, a weighted ensemble strategy is applied. The predictions from N models are used for predict each data points with respective accuracy ω. And each has prediction $p_{i,j}$ as 1 or 0, from all models are combined into a weighted sum, normalized by the total sum of the weights. This is calculated as followed:

$$\hat{p}_j = \frac{\Sigma_{i=1}^{N} \omega_i \cdot p_{i,j}}{\Sigma_{i=1}^{N} \omega_i}$$

By converting the weighted average into a binary prediction based on the threshold of 0.465, the final binary prediction for the jth data point is saved. This setup helps in avoiding overfitting while ensuring robust predictions by continuously retraining on newer data blocks.

The weighted average is then converted into a binary prediction based on a threshold of 0.465. If $\hat{p}_j$ exceeds 0.465, the final prediction for the jth data point is 1; otherwise, it is 0. This ensemble strategy reduces the risk of overfitting and ensures more robust predictions by continually retraining on updated data blocks.

3.3 Ensemble and Threshold

In our study, we leveraged the work of Fjellström, which implemented an ensemble of Long Short-Term Memory (LSTM) models to reduce variance and improve generalization. Fjellström's approach utilized several independent LSTM networks, each with the same architecture but initialized with different weight initializers. These LSTM networks were trained in parallel, and their predictions were combined to form an ensemble. The initialization methods employed were drawn from standard techniques available in Keras, as detailed in Fjellström's study.

Following the same methodology, we applied an ensemble of LSTM models in our work. Each LSTM model has the same architecture, these initializers, including methods such as RandomNormal, GlorotUniform, VarianceScaling, and others, allow us to capture different patterns in the data, ultimately enhancing model robustness. However, unlike Fjellström's original setup, we excluded the Identity initializer, as it is not suitable for use in Bidirectional LSTM layers due to the model's specific architecture.

3.4 Trading Strategy

We implemented a dynamic trading strategy based on an ensemble of LSTM model predictions. Each day, the model predicts which stocks are likely to outperform, and those with positive predictions are selected for the portfolio. The selected stocks are held for a fixed period of 10 days and their cumulative returns are calculated over the holding period. The portfolio is dynamically adjusted based on new predictions, and stocks are continuously bought or sold according to the model's forecasts. This approach aims to optimize returns by capturing short-term stock movements, while mitigating the risk of overfitting through the use of an ensemble and varying initializations.

4 Results

4.1 Accuracy

The prediction accuracy of the ensemble Bidirectional LSTM model is shown to be near 50%, which is consistent with the results of stock price predictions by Di Persio and Honchar (2016) and Nelson, Pereira, and de Oliveira (2017). Due to high complexity and volatility of the stock market, it is logical for our accuracy to stay around 50%.

4.2 Model Accuracy

Figure *4* shows the performance of LSTM models with three different initialization methods as examples for our model. The GlorotNormal and TruncatedNormal initializations both achieve peak accuracies around **0.513**, with GlorotNormal reaching this after six epochs before slightly declining, indicating potential overfitting. The Orthogonal initialization starts with a lower accuracy of **0.501** but improves steadily to around **0.510** after four epochs and remains stable. While both GlorotNormal and TruncatedNormal show some fluctuation in performance, Orthogonal is more consistent but with slightly lower accuracy overall. These results suggest that GlorotNormal and TruncatedNormal may offer better peak performance but require careful tuning to avoid overfitting.

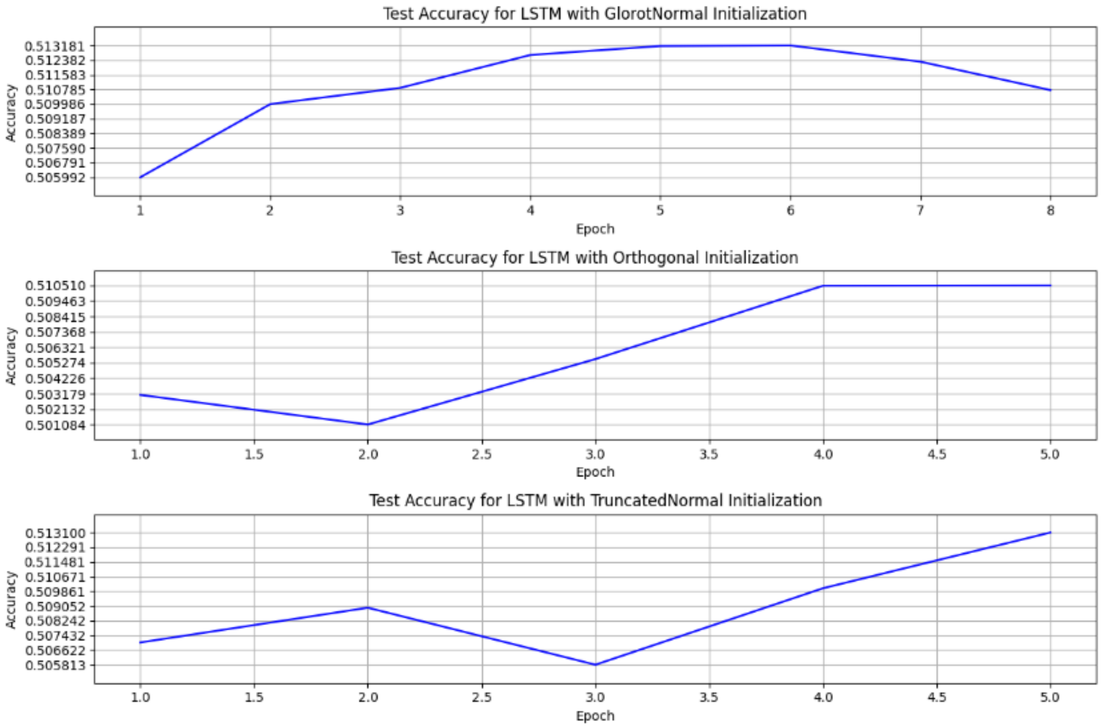

Figure 4: Test Accuracy of three models

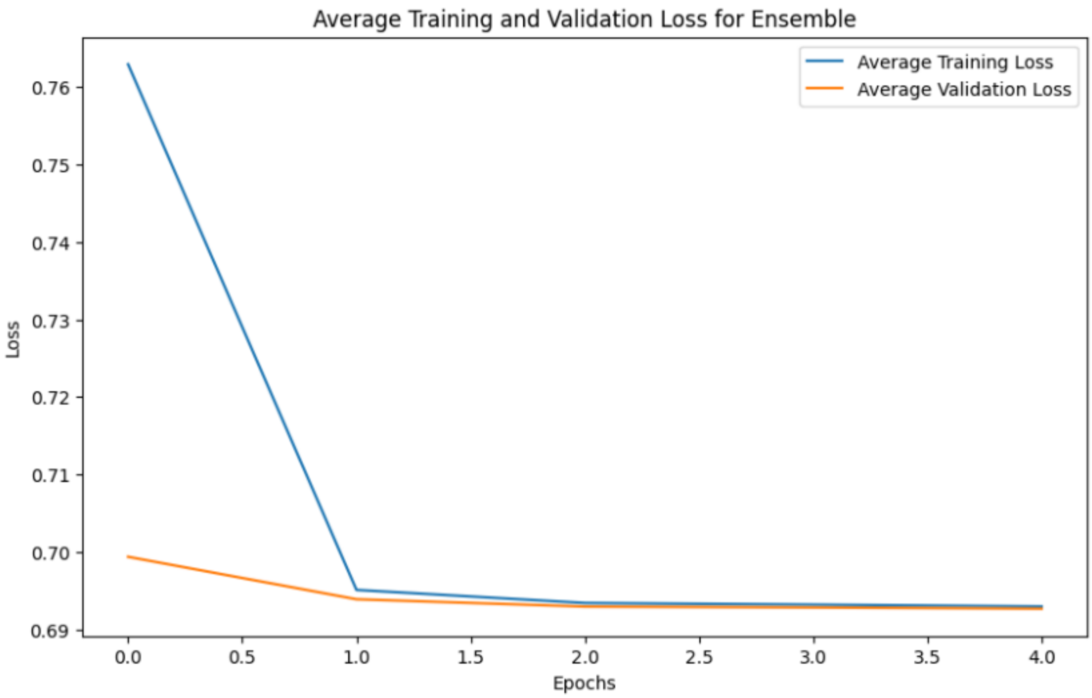

Figure 5: Average and Validation Loss for Ensemble

4.3 Model Training and Validation Loss

Figure 5 presents the average training and validation loss over several epochs for the model. Initially, the training loss starts at approximately 0.76, while the validation loss begins around 0.70. During the first epoch, the training loss drops sharply to approximately 0.69, indicating rapid convergence. After the initial epoch, both training and validation losses stabilize, exhibiting minimal change and remaining almost constant over the subsequent epochs. This behavior suggests that the model converges efficiently in the early stages of training, and the close alignment of training and validation losses indicates a well-balanced model that is not overfitting. The lack of significant divergence between the two curves further supports the model's ability to generalize effectively to unseen data. Due to the use of an early stopping strategy, each model pauses at different epochs based on its validation loss during training. For simplicity, the figure displays the minimum number of epochs across all models to better visualize overall model performance.

4.4 Portfolio Return

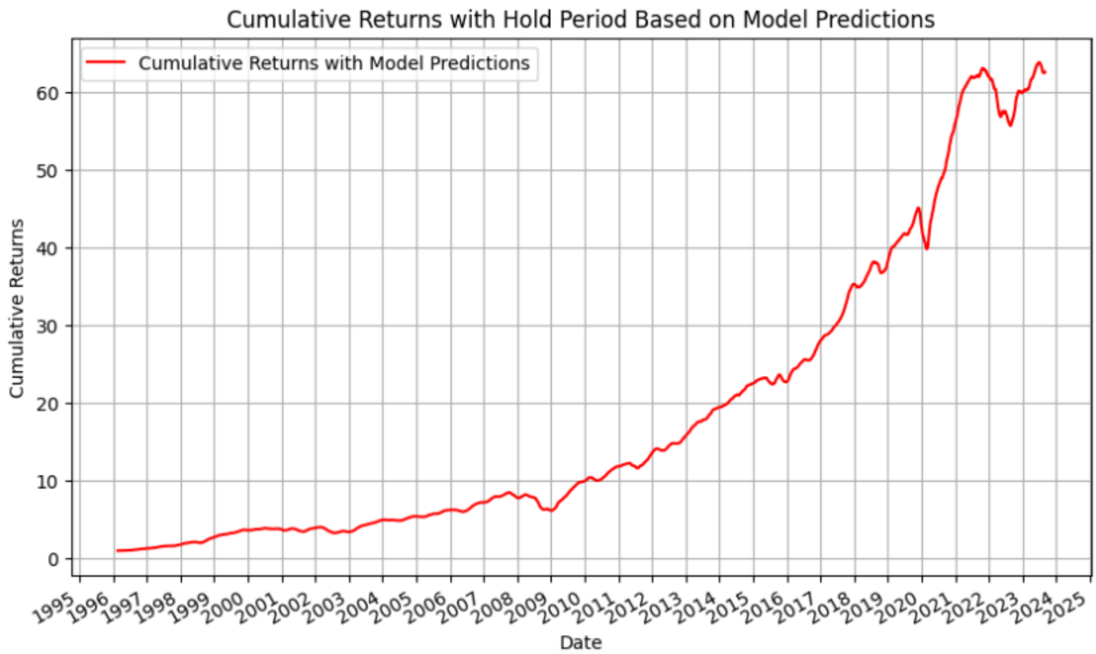

Figure 6: Cumulative Returns on Model Predictions

Figure 6 illustrates the cumulative returns generated based on the model's predictions over time. Starting in 1995, the returns show steady growth with minor fluctuations until around 2009, where the upward trend becomes more pronounced. Between 2012 and 2022, the cumulative returns accelerate significantly, reaching a peak in 2023 before experiencing a slight decline. Overall, the model's predictions lead to fine long-term cumulative returns, demonstrating consistent growth, particularly during the last decade. The fluctuations in the more recent years suggest some volatility in the model's predictions, but the general trend remains upward.

5 Summary and Conclusions

5.1 Analysis

The modified ensemble BiLSTM model have pretty close performance levels comparing to the previous studies, which is near 50%. While the model's profitability shows to be positive, we did not build up comparison for the model's profitability with alternative investment strategy. Hence, we cannot deduce our investment strategy is competitive, and the enhancements made may not be sufficient for substantial improvements. Therefore, we suggest for a deeper investigation to generate more effective and profitable investment strategy and a comparison with altenative strategies to evalute the competivity of our model.

5.1.1 Limitations

This study faces several limitations. Firstly, due to the constraints in computational resources and time, extensive hyperparameter tuning was not conducted enough, which may impact the model's performance. Secondly, using daily stock data instead of high-frequency real-time data limits the volume of data available for training, potentially affecting model accuracy. Additionally, the feature selection process, which relies on daily median returns, may overlook other significant variables that could enhance the model's predictive capability.

5.2 Future Research

Future research should focus on several key areas to address the limitations identified in this study. Enhancing feature selection methods to include additional financial indicators or technical indicators could improve model performance. Additionally, incorporating higher-frequency data and exploring alternative data sources may provide a richer dataset for training. Further hyperparameter optimization and model tuning are necessary to enhance predictive accuracy and generalizability. Finally, investigating different LSTM architectures or combining LSTM with other machine learning techniques might yield better financial time series forecasting results.

5.3 Conclusion

This study aimed to evaluate and improve the ensemble LSTM model for predicting stock price movements by applying it to the DJIA index components and comparing its performance with other investment strategies. Despite the enhancements made to the model, the results remained near 50%, similar to those of precious studies on stock price movement. Also, the modified ensemble LSTM model exhibited competitive profitability, but did not show significant advantages over other strategies.

The study indicates the complexities of transferring and adapting machine learning models to different financial contexts. The LSTM model has the potential to achieve better performance in prediction accuracy and profitability, but we need to do further research and refinement. Key areas for future work include optimizing feature selection, exploring higher-frequency data, and fine-tuning model parameters.

Overall, this study contributes to understanding LSTM applications in financial forecasting and emphasizes the importance of careful model adaptation. Continued exploration in this field could lead to more effective predictive models, enhancing decision-making and investment strategies in financial markets.

References

Barra, S., Carta, S. M., Corriga, A., Podda, A. S., & Recupero, D. R. (2020). Deep learning and time series-to-image encoding for financial forecasting. IEEE/CAA Journal of Automatica Sinica, 7(3), 683–692.

Di Persio, L., & Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: Comparisons and applications. International Journal of Circuits, Systems and Signal Processing, 10, 403–413.

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

Fjellström, C. (2022). Long short-term memory neural network for financial time series. Retrieved from https://arxiv.org/pdf/2201.08218

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Nelson, D. M., Pereira, A. C., & de Oliveira, R. A. (2017). Stock market's price movement prediction with LSTM neural networks. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 1419–1426).

Nogueira, F. (2014–). Bayesian optimization: Open source constrained global optimization tool for Python.

Unknown author. (2021, December 22). Detailed explanation of LSTM parameters in PyTorch. CSDN. https://blog.csdn.net/baidu_38963740/article/details/117197619

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of our report, we used ChatGPT to improve language and diction of words. After using ChatGPT, we reviewed, edited and refined the content as needed and take full responsibility for the content of our report.