Proceedings of CONF-MLA 2025 Symposium: Applied Artificial Intelligence Research
DOI: 10.54254/2755-2721/2025.BJ28891

Research on Stock Price Prediction Based on LSTM Model

Han Lv

Brunel College, North China University of Technology, Beijing, China
15201186092@163.com

Abstract. With the increasing complexity of financial markets and the rise of algorithmic trading, deep learning-based stock prediction methods have attracted widespread attention. This study focuses on the daily closing prices of Apple Inc. (AAPL) from 1980 to 2008. Time-series samples were constructed using Min–Max normalization and a sliding window approach. The predictive performance of single-layer and double-layer LSTM models (with or without Dropout) was systematically compared against support vector regression (SVR). Experiments were quantitatively evaluated using MSE, RMSE, and MAE metrics. Combined with loss curves and visualization of prediction results, the study found that the single-layer LSTM without Dropout achieved the best performance in terms of RMSE (2.020) and MAE (1.604), while SVR exhibited significantly higher errors. The results indicate that a moderate application of Dropout can effectively enhance model generalization, though it may slightly slow convergence. This study provides intuitive comparisons and optimization suggestions for applying LSTM in financial time-series prediction.

Keywords: LSTM, stock prediction, sliding window, Dropout

1. Introduction

The stock market has long attracted the attention of investors and economists. Fluctuations in stock prices not only directly affect individual investors' financial status but also have significant implications for the stability and development of the broader economic system. Therefore, accurately predicting stock prices has been regarded as one of the major challenges in finance. However, the complexity and uncertainty of the stock market impose many limitations on traditional time-series prediction methods. Conventional neural network models, such as RNNs, often rely on assumptions of linearity and normality, which prevent them from fully capturing the nonlinear, non-normal, and temporal dependency characteristics inherent in stock markets. As a result, accurately forecasting complex patterns and trends in stock prices is difficult. The LSTM network, first proposed by Hochreiter and Schmidhuber, effectively mitigates the vanishing gradient problem through a gating mechanism and has been widely applied in natural language processing and time-series analysis [1]. Fischer and Krauss introduced LSTM to stock price prediction and demonstrated its superior performance in short-term and medium-term market forecasts [2].

A review of stock index prediction models shows that early studies mainly relied on linear time-series models. The ARIMA (AutoRegressive Integrated Moving Average) model fits stationary

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2025.BJ28891

sequences using differencing and autoregressive terms, capturing short-term linear dependencies but exhibiting limited capability in handling nonlinearities and structural breaks [3]. To characterize volatility clustering commonly observed in financial data, the GARCH (Generalized Autoregressive Conditional Heteroskedasticity) family of models incorporates autoregressive and moving average terms in the variance equation, enabling the modeling of time-varying volatility [4]. With the growing need for nonlinear modeling, support vector regression (SVR) employs kernel techniques to map nonlinear time series into high-dimensional feature spaces for regression fitting, significantly improving predictive accuracy [5]. Random forests (RF) enhance model stability and noise resistance by aggregating multiple decision trees through bagging and random feature selection [6]. Extreme learning machines (ELM) randomly initialize hidden layer parameters and train output weights via least squares, greatly accelerating training speed, though regularization strategies are needed to improve generalization [7]. To better exploit sequential data, recurrent neural networks (RNNs) were introduced for prediction tasks, encoding sequences step by step through recursive hidden states [8]. However, standard RNNs are prone to vanishing or exploding gradients in long sequences, limiting their ability to model long-term dependencies. LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) networks alleviate this issue by introducing input, forget, and output gates, effectively capturing longer-range dependencies, and have become mainstream recurrent architectures [9]. For local pattern extraction, one-dimensional convolutional neural networks (1D-CNNs) automatically learn short-term fluctuation features via sliding convolutional kernels and reduce dimensionality through pooling layers, enhancing feature extraction [10]. Self-attention mechanisms and Transformer models capture global dependencies, efficiently associating features at arbitrary positions within sequences, providing new approaches for modeling stock price movements [11–12]. Hybrid models combining convolutional networks and LSTM have also been explored [13–15].

This study focuses on the daily closing prices of Apple Inc. (AAPL) from 1980 to 2008. Time-series samples were constructed using Min–Max normalization and a sliding window approach. Single-layer and double-layer LSTM models (with or without Dropout) were designed and compared with SVR in controlled experiments. Quantitative evaluation was conducted using MSE, RMSE, and MAE metrics, combined with visualization of training/validation loss curves and prediction results. The study systematically compares the performance of different models, providing optimization suggestions for applying LSTM in financial time-series prediction.

2. Related theory and model design

Long Short-Term Memory (LSTM) networks are a special type of recurrent neural network (RNN) that effectively address the vanishing or exploding gradient problems encountered in training traditional RNNs on long sequences through the introduction of gating mechanisms. The core structure of an LSTM unit includes three gates—forget gate, input gate, and output gate—as well as a cell state, which together regulate the flow and retention of information.

The computations within an LSTM unit at time step t are as follows:

The forget gate determines which information to discard from the cell state:

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \tag{1}$$

The input gate decides which new information will be stored in the cell state:

$$i_t = \sigma(W_i \bullet [h_{t-1}, x_t] + b_i) \tag{2}$$

DOI: 10.54254/2755-2721/2025.BJ28891

$$\widetilde{C_t} = tanh\left(W_c \bullet \left[h_{t-1}, x_t\right] + b_c\right) \tag{3}$$

The cell state is then updated:

$$C_t = f_t \bullet C_{t-1} + i_t \bullet \widetilde{C_t} \tag{4}$$

The output gate determines the hidden state to output:

$$o_t = \sigma(W_o \bullet [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \bullet tanh(C_t) \tag{6}$$

Here, σ denotes the sigmoid function; W and b represent the weight matrices and bias terms, respectively; and [h_{t-1}, x_t] denotes the concatenation of the previous hidden state and the current input vector.

Based on LSTM theory, this study uses sequences of normalized closing prices from the past 60 days as input. Single-layer or double-layer LSTM networks are employed to extract deep temporal dependencies. In the Dropout-augmented model variants, a Dropout layer with a 20% drop rate is added at the network output to mitigate overfitting. The final network layer is a fully connected regression layer that maps the extracted hidden representation to the predicted closing price of the next day. During training, the model parameters are updated using the Adam optimizer, and L2 regularization is applied to constrain model weights, improving generalization. Mini-batch training is adopted along with an early stopping mechanism: training is terminated when the validation loss does not decrease over several consecutive epochs to prevent overfitting. Additionally, time-series cross-validation is used, dividing the data into multiple sequential segments to ensure that each model evaluation occurs on genuine future time steps, providing a robust performance assessment. By systematically adjusting hyperparameters such as the number of LSTM layers, the size of hidden units, and the Dropout rate, and conducting experiments across multiple validation splits, this study ultimately identifies a model configuration that achieves an optimal balance between predictive accuracy and training efficiency.

3. Dataset description

This study uses daily closing price data of Apple Inc. (AAPL) from January 1, 1980, to October 31, 2008, comprising a total of 6,021 valid records. As a global technology leader by market capitalization, Apple Inc.'s stock price is influenced not only by macroeconomic factors but also by event-driven factors such as product releases and seasonal earnings fluctuations. This dataset is publicly accessible, high-quality, and spans a long continuous period, making it suitable for reflecting both long-term trends and market volatility of AAPL stock. Due to its high liquidity and strong market attention, AAPL stock has become a classical benchmark dataset in the field of financial time-series prediction.

Data preprocessing was performed as follows. First, missing values were checked using the Pandas library, and no missing entries were found in the original CSV file. Outliers were identified and handled based on the three-sigma (3σ) rule, with extreme values beyond the mean ± three standard deviations replaced by nearby non-outlier values. Next, the closing prices were scaled to the [0,1] range using MinMaxScaler, according to the following formula:

DOI: 10.54254/2755-2721/2025.BJ28891

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{7}$$

where $x_{\min}$ and $x_{\max}$ represent the minimum and maximum closing prices in the training set, respectively.

Subsequently, a time-series sliding window was constructed with a time step of 60 days. Each sample consists of a sequence of normalized closing prices from the previous 60 days, used to predict the closing price of day t+1. Finally, the dataset was split in chronological order, with the first 80% of samples used as the training set and the remaining 20% as the test set, resulting in 4,816 training samples and 1,205 testing samples.

4. Experimental setup and results analysis

4.1. Experimental configuration

The experiments were conducted on a workstation equipped with an NVIDIA GTX 1080 Ti GPU (11 GB VRAM) and an Intel Core i7-9700K CPU. The software environment included TensorFlow 2.4.1, scikit-learn 0.24.2, and Python 3.8. For the network models, the baseline consisted of a single-layer LSTM (50 hidden units) + Dropout (0.2) + fully connected regression layer. Ablation models included a double-layer LSTM (50 units per layer, no Dropout) and a single-layer LSTM (50 units, no Dropout). For comparison with traditional machine learning, support vector regression (SVR) with an RBF kernel was employed. The Adam optimizer was used for all models (initial learning rate 0.001, β_1=0.9, β_2=0.999) [16], and the loss function was mean squared error (MSE):

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i - y_i\right)^2 \tag{8}$$

Training was run for 20 epochs. If the training curve showed convergence after epoch 15, early stopping was applied. The batch size was set to 32, balancing GPU memory capacity and convergence efficiency. A grid search was conducted over hidden units {50,100} and Dropout rates {0.1,0.2,0.3}, using 5-fold time-series cross-validation. The configuration with the lowest RMSE on the test set was selected. The main evaluation metrics were root mean squared error (RMSE) and mean absolute error (MAE):

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{y}_i - y_i\right)^2}, \quad \mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|\hat{y}_i - y_i\right|. \tag{9}$$

The prediction results for each model are summarized in Table 1:

Table 1. Prediction results of different models

Model	RMSE	MAE
Single-layer LSTM + Dropout	2.085	1.672
Single-layer LSTM (no Dropout)	2.020	1.604
Double-layer LSTM (no Dropout)	2.053	1.639
SVR (RBF kernel)	13.198	10.456

As shown in Table 1, introducing Dropout increased the RMSE from 2.020 to 2.085 and the MAE from 1.604 to 1.672 compared to the single-layer LSTM without Dropout. This indicates that Dropout can reduce overfitting risk on this dataset but requires a trade-off between generalization

DOI: 10.54254/2755-2721/2025.BJ28891

and fitting ability. The double-layer LSTM did not significantly improve performance compared to the single-layer structure (RMSE only slightly decreased to 2.053), suggesting that increasing depth yields limited marginal benefit given this data volume and sequence characteristics. SVR performed poorly in modeling nonlinear temporal dependencies, with RMSE far higher than that of LSTM models, highlighting the clear advantage of deep learning methods in long-sequence prediction tasks.

To further verify the statistical significance of model differences, paired t-tests were conducted on the prediction residuals of single-layer LSTM models (with and without Dropout) and double-layer LSTM. The p-values were all greater than 0.05, indicating no statistically significant differences between configurations. In summary, the single-layer LSTM (no Dropout) achieved the best performance with the lowest RMSE and MAE, suggesting that a simple structure is sufficiently effective for this experiment. Future research could incorporate additional macroeconomic indicators, event-driven features, or ensemble learning to further improve predictive accuracy.

4.2. Analysis of experimental results

To visually assess the training process and predictive performance of the models, this study plotted the training/validation loss curves (Figure 4.1), the comparison of actual and predicted values on the test set (Figure 4.2), and an overall comparison of results across models (Figure 4.3). The analysis below refers to these figures.

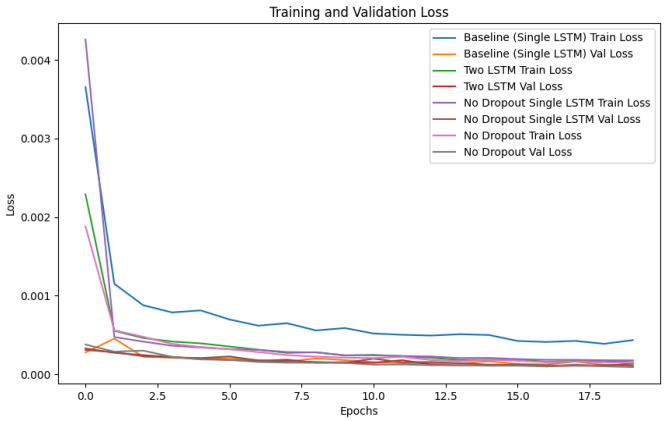

Figure 1. Training and validation loss curves

From Figure 1, the convergence speed and stability of the models can be observed. All models exhibit a rapid decline in the first five epochs. The baseline single-layer LSTM with Dropout (blue line) decreases more slowly and stabilizes after 15 epochs. Both the double-layer LSTM (green line) and single-layer LSTM without Dropout (purple line) converge more quickly, indicating that while Dropout helps suppress overfitting, it slightly slows convergence. Signs of overfitting are minimal: the validation loss curves (orange, red, brown, and gray lines) closely follow the training loss without noticeable upward deviation, suggesting stable performance on the validation set.

DOI: 10.54254/2755-2721/2025.BJ28891

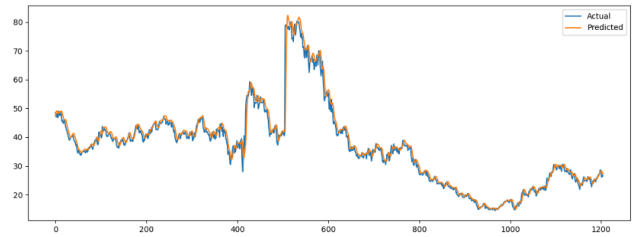

Figure 2. Prediction results of single models

Figure 2 shows that the baseline single-layer LSTM (orange line) closely tracks major price fluctuations. The predicted values align well with actual prices (blue line) at peaks and troughs, with only minor deviations. During large fluctuations, such as sharp spikes in the middle of the sequence, the model exhibits slight lag, but the overall error remains within an acceptable range.

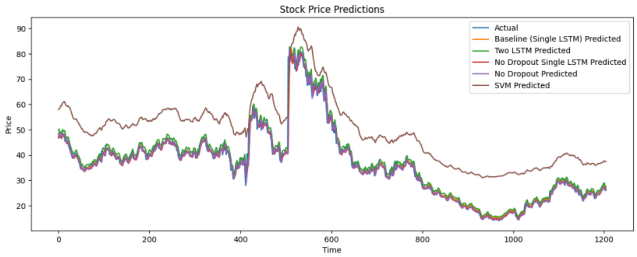

Figure 3. Overall comparison across models

The curves of all deep models (orange, green, red, purple lines) overlap substantially, confirming the close performance metrics shown in Table 1. SVR (brown line) consistently overestimates prices and fails to capture the nonlinear trends, demonstrating a clear performance gap with LSTM models. The double-layer LSTM (green line) slightly outperforms other LSTM variants during abrupt price changes, though the difference is marginal. The single-layer LSTM without Dropout (red line) performs slightly better in stable regions. Overall, all LSTM variants fit the general trend well, indicating that a single-layer structure is sufficient for this task. Appropriately applied Dropout helps improve model generalization but slightly affects learning speed. SVR cannot effectively model long-sequence dependencies, resulting in substantial prediction deviations.

5. Conclusion

This study focused on daily closing prices of Apple Inc. (AAPL) from 1980 to 2008 and systematically constructed and compared multiple stock prediction models, including single-layer and double-layer LSTMs (with or without Dropout) and SVR. By applying Min–Max normalization and a sliding window approach, time-series samples were efficiently generated. Combined with RMSE and MAE metrics, as well as visual analysis of training/validation loss and prediction results, the study demonstrated the superior prediction accuracy and training efficiency of the single-layer LSTM without Dropout. The study further revealed that Dropout affects both model generalization and convergence speed, and that increasing network depth does not significantly improve performance under the current dataset and sequence characteristics. It also highlighted the limitations of traditional SVR in capturing long-sequence dependencies. Overall, this research proposes a simple yet effective LSTM modeling framework, providing practical guidance for

DOI: 10.54254/2755-2721/2025.BJ28891

financial time-series prediction and laying a solid foundation for future work incorporating additional features or ensemble learning methods.

References

[1] Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation, 1997, 9(8): 1735-1780.
[2] 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.
[3] Box G E P, Jenkins G M. Time series analysis: forecasting and control. Holden-Day, 1970.
[4] Bollerslev T. Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics, 1986, 31(3): 307-327.
[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] Huang G B, Zhu Q Y, Siew C K. Extreme learning machine: theory and applications. Neurocomputing, 2006, 70(1-3): 489-501.
[8] Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors. Nature, 1986, 323(6088): 533-536.
[9] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. EMNLP 2014.
[10] Sezer O B, Ozbayoglu A M. Algorithmic financial trading with deep convolutional neural networks: time series to image conversion approach. Applied Soft Computing, 2020, 93: 106275.
[11] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. NeurIPS 2017.
[12] Qiu X, Song D. Predicting the direction of stock market index movement using an optimized artificial neural network model. Expert Systems with Applications, 2016, 79: 206-213
[13] Zhang L, Aggarwal C C, Qi G J. Stock price prediction via stacked autoencoder and LSTM neural network. Expert Systems with Applications, 2018, 110: 20–30.
[14] Wang X, Li Y. Wavelet transform and LSTM-based neural network for stock prediction. Applied Soft Computing, 2021, 110: 107642.
[15] Nawaz K, Ghufran A. A hybrid CNN-LSTM model for time series forecasting. Neural Computing and Applications, 2019, 31(1): 123–135.
[16] Goodfellow I, Bengio Y, Courville A. Deep Learning. MIT Press, 2016.