A Hybrid CNN-LSTM Model for Industry-Level Stock Price Prediction

Jiancheng Gao*

School of Mathematical Sciences, Inner Mongolia University, Hohhot, 010030, China

*Corresponding author:mathgjc@163.com

Abstract. Due to the rapid price fluctuations in the stock market, traditional stock price prediction methods struggle to achieve satisfactory prediction performance. We use the CNN-LSTM model to predict stock prices at the industry level in this paper. The data used in this paper come from the first-level industries of the Shenwan classification, covering the period from January 4, 2000, to April 1, 2025. It consists of 5612 samples and the last 500 samples are selected as the test set. The dataset includes eight features: opening price, highest price, lowest price, closing price, trading volume, price change, percentage change, and turnover rate. We use three evaluation methods: MAE, RMSE, and R^2 to compare the results with DNN, CNN, and LSTM. It was found that the CNN-LSTM model had the best prediction performance. The methods and results in this paper can provide references and insights for investors to conduct research and make allocations in the A-share market from an industry level.

Keywords: Stock Price Prediction, Convolutional Neural Network, Long-Short-Term Memory Neural Network, Industry-Level

1 Introduction

The trend of stock prices is one of the core issues in financial research. It not only reflects the operating conditions of individual companies but is also influenced by various factors such as domestic and international economic environments, industry development trends, market mechanisms, and investor sentiment [1]. Traditional stock price prediction methods include fundamental analysis and technical analysis. The nonlinear and volatile nature of stock prices makes traditional forecasting methods more difficult.

We collected data from 34 first-level industries under the Shenwan classification from January 4, 2000, to April 11, 2025, using the Wind database. By combining the feature extraction capability of CNN with the ability of LSTM to handle temporal dependencies, we use the CNN-LSTM model to predict stock prices at the industry level.

A. Mallik et al. (eds.), *Proceedings of the 2025 6th International Conference on Management Science and Engineering Management (ICMSEM 2025)*, Atlantis Highlights in Economics, Business and Management 4,
https://doi.org/10.2991/978-94-6463-845-5_69

2 Related Work

In recent years, a large number of researchers have applied deep learning methods to predict stock prices. Kim et al. (2019) combined the Convolutional Neural Network(CNN) with the Long-Short-Term Memory Neural Network(LSTM) and found that the feature fusion CNN-LSTM model performs better than single models through empirical analysis of the S&P 500 index using financial time series data [2]. Nosratabadi S et al. (2020) conducted a review of the applications of deep learning in stock prediction using the PRISMA methodology and found that CNN and LSTM are widely applied in stock price prediction [3]. JMT Wu et al. (2023) proposed a hybrid architecture named SACLSTM, which used CNN to extract features and used them as input to LSTM to achieve the combination of neural networks [4]. They confirm that the predictive effect is better than MLP, CNN, and LSTM by applying the framework to predict 10 stocks from the US and Hong Kong [5]. Q Zhao et al. (2024) also confirm that CNN-LSTM performs better by selecting five features, including opening price, closing price, highest price, lowest price, and trading volume, to predict the stock data of multiple companies [6]. H Liu (2025) selected 12 stocks from different markets and used nearly 10 years of historical data from Yahoo Finance. After testing 800 hyperparameter combinations, the study concluded that the CNN-LSTM model can generally predict stock prices effectively [7].

The current research on stock price prediction using CNN-LSTM mostly focuses on individual stocks or broad indices, with relatively few studies conducted from the industry-level perspective. The potential contribution of this paper is the use of the CNN-LSTM model for industry-level prediction. The main feature of this method is that it first extracts features using CNN and then passes them to LSTM for model training. This method enables more accurate predictions while improving prediction stability and generalization ability by utilizing comprehensive feature extraction and considering both short-term and long-term dependencies. Conducting industry-level analysis enhances visibility into sector index trends, thereby informing more strategic decisions in individual stock allocation.

3 THeory

3.1 CNN

The CNN, a type of feedforward neural network proposed by Y. LeCun et al., has been widely used in fields such as image processing and speech recognition [8]. One of the main features of CNN is that it can extract features through convolution operations.

A classic CNN consists of convolutional layers, pooling layers, and fully connected layers. Each convolutional layer contains multiple filters, characterized by parameter sharing and sparse connectivity. These filters can extract features from the original data, forming feature maps. The pooling layer can reduce the dimensions of feature maps, further extract features, and enhance model robustness through pooling operations, such as max pooling or average pooling. Finally, a fully connected neural network is used to generate the output of the model. The structure of the CNN is shown in Figure 1.

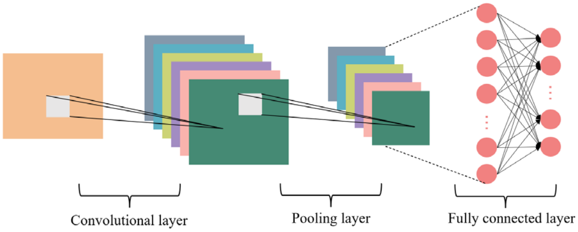

Fig. 1. CNN structure diagram.

The convolutional layer is the core of the CNN. The discrete convolution operation formula is as follows:

$$c(n) = (f \cdot g)(n) = \sum_{\tau=-\infty}^{+\infty} f(\tau) g(n-\tau), \quad (1)$$

where f is the input sample image, g is the weighting function of the filter, and c is the feature map.

By sliding the filter along the image sample, the convolution operation forms different features. In each corresponding subregion, the convolution operation involves matrix dot multiplication and addition, and the resulting scalar value becomes an element in the feature map. An example of a convolution operation is shown in Figure 2. Suppose the input sample has a size of 5×5, the convolution kernel has a size of 3×3, and the stride is 1.

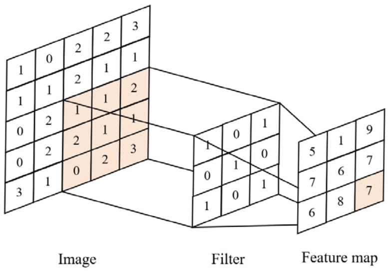

Fig. 2. Convolutional layer calculation example.

The function of the pooling layer is downsampling on the feature maps, eliminating less important features and further reducing the number of parameters. Max pooling and average pooling are the common pooling methods. Max pooling outputs the maximum value from a $n \times n$ region as the sampled value, while average pooling outputs the average value from a $n \times n$ region as the sampled value. Examples of 2×2 max pooling and 2×2 average pooling are shown in Figure 3.

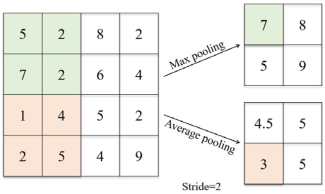

Fig. 3. Max pooling and average pooling examples.

3.2 LSTM

The LSTM is a special RNN proposed by S. Hochreiter et al. in 1997 and it was proposed to solve the gradient explosion or disappearance phenomenon in traditional RNN [9].

LSTM consists of an input gate, a forget gate, an output gate, and a memory cell. The forget gate in LSTM allows the network to retain long-term dependencies in sequential data by selectively critical information over extended time horizons. The structure of the LSTM is shown in Figure 4.

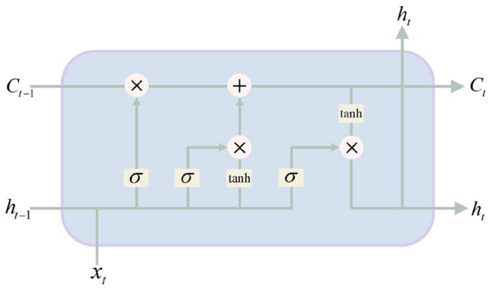

Fig. 4. LSTM structure diagram.

As shown in Figure 4, at the time step t, the LSTM has three inputs: the input value of the network at the current time step, the output value of LSTM at the previous time step, and the cell state at the previous time step and LSTM has two outputs: the output value of LSTM at the current time step and the cell state at the current time step. LSTM can control the switches of three gates to filter features effectively: retaining important information and filtering out irrelevant ones.

The first gate in Figure 4 is the forget gate, which determines how much feature is retained at the previous time step. Its calculation formula is as follows:

$$f_t = \sigma(w_f \cdot [h_{t-1}, x_t] + b_f), \tag{2}$$

where the range of f is (0,1), w_f and b_f are trainable parameters, and σ is *sigmoid* function.

The second gate in Figure 4 is the input gate, which is used to control the update and generate candidate values for the cell state. Its calculation formula is as follows:

$$i_t = \sigma(w_i \cdot [h_{t-1}, x_t] + b_i), \tag{3}$$

$$\tilde{C} = \tanh(w_c \cdot [h_{t-1}, x_t] + b_c), \tag{4}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t, \tag{5}$$

where the range of C_t and i_t are (0,1) and w_i, b_i, w_c, and b_c are trainable parameters.

The third gate in Figure 4 is the output gate, which determines the final output at the current time step. Its calculation formula is as follows:

$$O_t = \sigma(w_o \cdot [h_{t-1}, x_t] + b_o), \tag{6}$$

$$h_t = O_t \cdot \tanh(C_t), \tag{7}$$

where the range of O_t is (0,1), w_o and b_o are trainable parameters.

3.3 CNN-LSTM

To fully capture the changes in stock prices and improve the accuracy of predictions, we combine the advantages of CNN, a model that is often used in image processing and can extract features efficiently, with LSTM which can capture long-term and short-term dependencies. The CNN-LSTM model constructed in this paper includes the input layer, CNN layer (convolutional and pooling layer), LSTM layer, fully connected layer, and output layer. The specific structure of CNN-LSTM is shown in Figure 5.

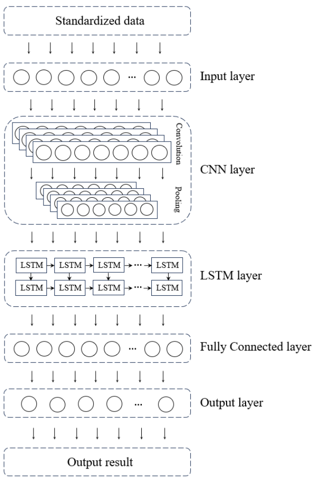

Fig. 5. CNN-LSTM structure diagram.

First, to eliminate the influence of different dimensions, we performed $z-score$ normalization on the original data that needs to be trained. The specific formula is as follows:

$$y_i = \frac{x_i - \overline{x}}{s}, \tag{8}$$

where x_i is the original data, $\overline{x}$ is the mean value of the original data, y_i is the standardized value, and s is the standard deviation of the original data.

Then we put the standardized data, which will be fed into the CNN layer, into the input layer and initialize the parameters across all layers. The CNN layer involves two steps: the convolutional layer can use filters to extract features from the input data and generate feature maps and the pooling layer can through pooling operation further refine features and reduce the dimensionality to minimize parameters. The LSTM layer can capture long-term dependencies and process the data coming from the CNN layer. The output coming from the CNN layer was then passed to the fully connected layer, which finally produced the standardized prediction through the output layer.

After obtaining the output values, we compared them with the actual values. If the error is less than the predetermined threshold, save the trained network. Otherwise, continue to train the CNN-LSTM.

We input the standardized test data into the trained model. At this time, the model's output value was still in standardized form and we obtained the final predicted values through inverse normalization to complete the prediction process.

4 EXperiments

We used MATLAB to train the CNN-LSTM model. The MATLAB toolboxes used in the experiment are listed in Table 1. To intuitively demonstrate the prediction performance of the CNN-LSTM model, we selected DNN, CNN, and LSTM for comparison.

Table 1. MATLAB toolboxes were used in the experiments.

MATLAB Toolbox
Deep Learning Toolbox
Image Processing Toolbox
Statistics and Machine Learning Toolbox

4.1 Data

In this paper, the data were obtained from the Wind database. There are 31 industries in the Shenwan first-level classification, such as Electronics, Basic Chemicals, and Steel. Each sample from these industries contains eight features, including opening price, highest price, lowest price, closing price, trading volume, price change, price change percentage, and turnover rate. For the 24 industries, the data were collected from January 4, 2020, to April 11, 2025, with a total of 5612 samples, of which the last 500 samples were selected as the test set. Due to missing data, for the other seven industries, the data were collected from January 2, 2014, to April 11, 2025, with a total of 2741 samples, and the last 224 samples were selected as the test set.

4.2 Parameter Settings

In the CNN-LSTM model, manual hyperparameter tuning is necessary. Based on prior experience, we adjusted these hyperparameters by observing the loss curve. The specific parameter settings of the CNN-LSTM model are listed in Table 2.

Table 2. CNN-LSTM parameters.

Layer		Parameter	Value
CNN	Convolution	Number of filters	32
		Size of filter	1
		Activation function	Tanh
		Padding	Same
	Pooling	Size of pooling	1
		Padding	Same
LSTM		Units	64
		Activation function	Tanh
Optimizer			Adam
Learning rate			0.001

4.3 Results Evaluation

In this paper, we use the Mean Absolute Error, Root Mean Square Error, and R-square to evaluate the prediction performance.

The Mean Absolute Error formula is as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | \hat{y}_i - y_i |, \tag{9}$$

where n is the number of samples, $\hat{y}_i$ is the predicted value, and y_i is the actual value.

The Root Mean Square Error formula is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}, \tag{10}$$

The R-square formula is as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{n} (\overline{y}_i - y_i)^2}, \tag{11}$$

MAE and RMSE reflect the deviation between predicted values and actual values. The smaller the value, the better the model effect. R^2 reflects the fit between predicted values and actual values. The smaller the value, the better the model effect.

4.4 Results

We used DNN, CNN, LSTM, and CNN-LSTM to predict the data of 31 industries. The prediction results are shown in Figure 6-13. Due to space limitations, only the prediction results of the Integrated Industry and Building Materials Industry are presented here. The predicted values obtained from the four neural network models are quantitatively evaluated using evaluation metrics, and the results are shown in Table 3.

Fig. 6. Comparison of Predicted and Actual Values for DNN in the Integrated Industry.

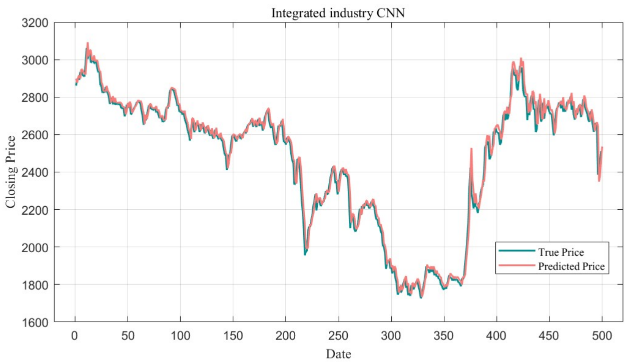

Fig. 7. Comparison of Predicted and Actual Values for CNN in the Integrated Industry.

Fig. 8. Comparison of Predicted and Actual Values for LSTM in the Integrated Industry.

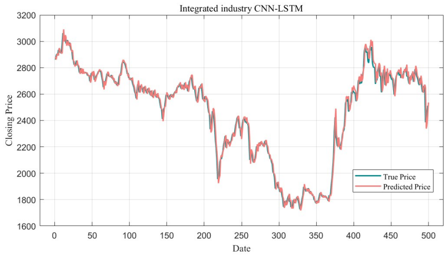

Fig. 9. Comparison of Predicted and Actual Values for CNN-LSTM in the Integrated Industry.

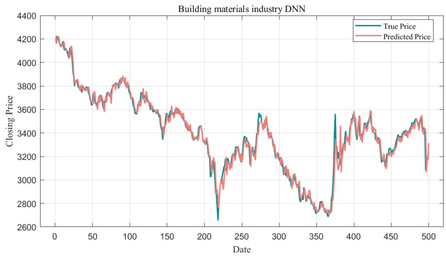

Fig. 10. Comparison of Predicted and Actual Values for DNN in the Building Materials Industry.

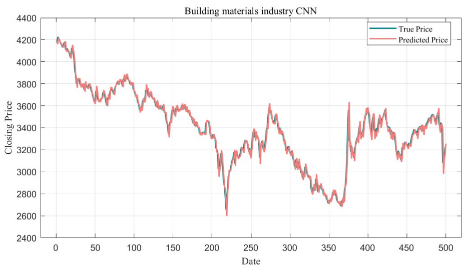

Fig. 11. Comparison of Predicted and Actual Values for CNN in the Building Materials Industry.

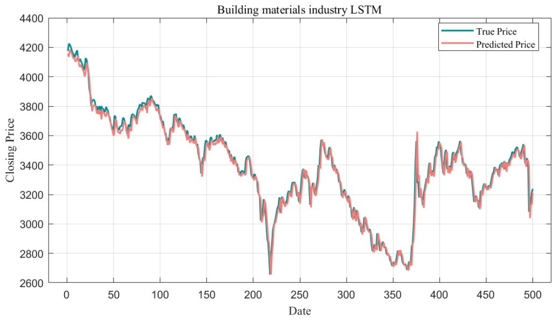

Fig. 12. Comparison of Predicted and Actual Values for LSTM in the Building Materials Industry.

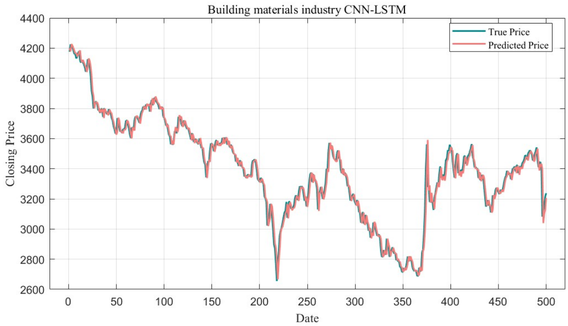

Fig. 13. Comparison of Predicted and Actual Values for CNN-LSTM in the Building Materials Industry.

Table 3. Evaluation results for Integrated Industry and Building Materials Industry.

Industry	Model	MAE	MRSE	R^2
Integration	DNN	36.1910	48.3126	0.9811
	CNN	31.6430	44.5563	0.9839
	LSTM	31.2042	43.6203	0.9846
	CNN-LSTM	30.8058	42.3619	0.9855
Building materials	DNN	70.9791	95.1600	0.9798
	CNN	66.7226	93.8488	0.9804
	LSTM	58.4873	80.6632	0.9855
	CNN-LSTM	56.2116	74.8224	0.9875

Figure 6-9 shows the predicted values and actual values of the Integrated industry using DNN, CNN, LSTM, and CNN-LSTM respectively. We can see that the prediction performance of DNN in the first 200 days is poor, and the prediction performance of CNN and LSTM in 400-450 days is also not good. The prediction performance of CNN-LSTM in 400-450 days is better than CNN and LSTM. This shows that the training effect of CNN-LSTM is poor for the situation of rapid change and non-smooth data, but CNN-LSTM has the best prediction performance overall.

Figure 10-13 shows the predicted values and actual values of the Building Materials industry using DNN, CNN, LSTM, and CNN-LSTM respectively. We can see that DNN has poor prediction results at the two peaks of 220 days and 270 days, CNN performs poorly between 390 and 430 days due to the rapid price fluctuations, and LSTM also has poor prediction results at 0-100 days due to the rapid price fluctuations. CNN-LSTM has good prediction performance throughout the entire period.

Table 3 lists the evaluation results of the Integrated industry and Building Materials industry. The MAE and MRSE values of the two industries consistently show that the prediction performance of the four models from lowest to highest is as follows: DNN, CNN, LSTM, and CNN-LSTM. This shows that the CNN-LSTM has the smallest deviation in prediction results. The R^2 values of the two industries also show that the CNN-LSTM achieves the highest degree of fit in prediction results.

The results show that the CNN-LSTM performs well in both deviation and goodness of fit and can effectively predict the performance of first-level industries under the Shenwan classification. It is worth noting that even minor prediction errors can have significant impacts in practical scenarios. For example, underestimating an upward trend may lead to missed buying opportunities. This approach and the results presented in the paper provide valuable references for investors to conduct industry-level research in the A-share market, such as industry index construction, sector rotation analysis, and fund investment direction assessment.

5 Conclusions

This paper uses the CNN-LSTM model to predict stock prices at the industry level. During model training, CNN is first used for feature extraction, and then LSTM is used for training. After passing through the fully connected network, the output value is obtained. Finally, the trained CNN-LSTM model is used to predict the closing prices of each first-level industry under the Shenwan classification. The predicted and actual images intuitively reflect the comparison of the prediction results, while MAE and MRSE show that CNN-LSTM has the smallest deviation, and R^2 shows that CNN-LSTM has the highest degree of fit. The above results confirm the performance of the CNN-LSTM model in predicting stock prices at the industry level. The results of this paper can provide valuable references for investors in sector rotation, and the method offers insights for industry-level research in the A-share market. The paper has certain limitations. Due to data scarcity, the training samples across seven industries are nearly halved. Moreover, this paper solely focuses on the impact of fundamental indicators on stock prices, without taking other potential influencing factors into account. Our future research will mainly focus on selecting technical indicators and sentiment indicators to further improve prediction performance. Additionally, the attention mechanism can be introduced to enhance the model's transparency.

References

1. Zhang, Lin Lin, and H. Kim. "The influence of financial service characteristics on use intention through customer satisfaction with mobile fintech." Journal of System and Management Sciences 10.2 (2020): 82-94.Davis, A. R., Bush, C., Harvey, J. C. and Foley, M. F., "Fresnel lenses in rear projection displays," SID Int. Symp. Digest Tech. Papers 32(1), 934-937 (2001).
2. Kim, Taewook, and Ha Young Kim. "Forecasting stock prices with a feature fusion LSTM-CNN model using different representations of the same data." PloS one 14.2 (2019): e0212320.
3. Nosratabadi, Saeed, et al. "Data science in economics: comprehensive review of advanced machine learning and deep learning methods.
4. ." Mathematics 8.10 (2020): 1799.
5. Wu, Jimmy Ming-Tai, et al. "A graph-based CNN-LSTM stock price prediction algorithm with leading indicators." Multimedia Systems 29.3 (2023): 1751-1770.
6. Zhao, Qinghe, Yue Hao, and Xuechen Li. "Stock price prediction based on hybrid CNN-LSTM model." Applied and Computational Engineering 104 (2024): 110-115.
7. Liu, Haotian. "A Hybrid CNN-LSTM Approach for Effective Stock Price Prediction in Optimizing Investment Strategies." International Workshop on Navigating the Digital Business Frontier for Sustainable Financial Innovation (ICDEBA 2024). Atlantis Press, 2025.
8. Campbell, John Y., and Robert J. Shiller. "Yield spreads and interest rate movements: A bird's eye view." The Review of Economic Studies 58.3 (1991): 495-514.
9. Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.