
Chinese Stock Market Prediction Using Hybrid Ensemble Model

Jerry Yin

Carnegie Mellon University
Pittsburgh, PA 15213
xianshy@andrew.cmu.edu

Jiayu Wang

Carnegie Mellon University
Pittsburgh, PA 15213
jiayuan@andrew.cmu.edu

Anqi Xie

Carnegie Mellon University
Pittsburgh, PA 15213
anqix@andrew.cmu.edu

Haoyu Yang

Carnegie Mellon University
Pittsburgh, PA 15213
haoyuy@andrew.cmu.edu

Abstract

This report addresses the challenge of stock price prediction in the Chinese stock market, which differs from Western markets due to higher government intervention, dominance of retail investors, and elevated volatility. Building on a baseline Long Short-Term Memory (LSTM) model, we propose an ensemble learning framework tailored for the Shanghai Stock Exchange (SSE) Composite Index. Our ensemble integrates CNN-BiLSTM-Attention, Transformer, and Random Forest models through weighted averaging to enhance predictive accuracy. Using daily data from January 1, 2010, to January 1, 2025, and incorporating technical indicators (e.g., EMA, RSI, MACD), the ensemble model significantly outperforms the baseline LSTM, as evaluated by Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), and R^2 . These findings highlight the value of deep learning ensembles in capturing the complex dynamics of China's financial markets.

1 Introduction

Stock price prediction is a cornerstone of financial forecasting, enabling informed investment strategies and risk management. The Chinese stock market differs significantly from its American counterpart in terms of structure, investor composition, and regulatory environment. The U.S. stock market is primarily driven by institutional investors, whereas China's market is dominated by retail investors, leading to higher volatility and speculative trading. Additionally, government policies and interventions play a substantial role in influencing stock prices in China, creating unique challenges for predictive modeling.

In this study, we develop a deep learning-based framework tailored to the Chinese stock market, using the SSE Composite Index (000001.SS) as our case study. We begin with a baseline model based on Long Short-Term Memory (LSTM) networks, which are well-suited for modeling sequential dependencies in time-series data. Building on this foundation, we introduce an ensemble learning approach that integrates three complementary models: CNN-BiLSTM-Attention, Transformer, and Random Forest. The ensemble predictions are combined using weighted averaging, with weights optimized on validation performance. This approach aims to balance the strengths of different architectures, improving generalization and robustness. The rest of the report outlines the dataset, modeling methodology, empirical results, and future paths for improvement.

2 Literature Review

Stock price prediction has evolved significantly, driven by advancements in computational techniques. Traditional statistical methods, such as Autoregressive Integrated Moving Average (ARIMA), are limited in capturing non-linear patterns inherent in stock markets [2]. Machine learning approaches, including Support Vector Machines (SVM) and Random Forest, improved performance by modeling complex feature interactions but struggled with temporal dependencies [2].

Deep learning has transformed time-series forecasting, particularly in finance. Long Short-Term Memory (LSTM) networks, introduced by Hochreiter and Schmidhuber, excel at capturing sequential patterns, making them suitable for stock price prediction [1]. Recent studies have explored hybrid models, such as combining Convolutional Neural Networks (CNNs) with LSTMs to extract spatial and temporal features [4]. Transformers, leveraging self-attention mechanisms, have shown promise in handling long-range dependencies, offering advantages over RNNs in certain contexts [3].

Ensemble methods, which combine multiple models, have gained traction for improving robustness and generalization. For instance, Zhang et al. proposed a deep learning system for short-term price trend prediction in the Chinese market, achieving high accuracy through feature engineering and model integration [5]. Similarly, Kim demonstrated that deep learning models outperform traditional benchmarks in trend forecasting [6]. These findings underscore the potential of combining diverse models to address the Chinese market's volatility, aligning with our ensemble approach.

3 Methodology

3.1 Data Collection

We sourced daily Open, High, Low, Close (OHLC) prices and trading volume for the SSE Composite Index (000001.SS) from Yahoo Finance, covering January 1, 2010, to January 1, 2025. This 15-year dataset, comprising approximately 3,700 trading days, captures long-term market trends and short-term fluctuations, providing a robust foundation for model training and evaluation.

3.2 Data Preprocessing

The raw data underwent rigorous preprocessing:

- **Missing Values:** Removed duplicate entries and dropped records with missing values.
- **Normalization:** All input features were scaled to $[0, 1]$ ranges using Min-Max normalization. This scaling, applied only on the training data and then reused for validation/test, helps the neural networks train faster and prevents any single feature from dominating due to scale.
- **Target Definition:** Our prediction target was defined as the next-day return of the index (percentage change from today's close to tomorrow's close). This framing (predicting return rather than price directly) stabilizes the time series and allows us to later convert predictions into prices. We shifted the daily return series by one day to align it as the target for each input window.
- **Data Splitting:** Divided into training (all except the last 350 samples), validation (last 100 before test), and testing (final 250 samples) sets.

Our forecasting pipeline (illustrated in Figure 1) consists of several stages, from data acquisition and preprocessing to sequence preparation, model development (including a baseline and advanced deep learning models), training, and evaluation.

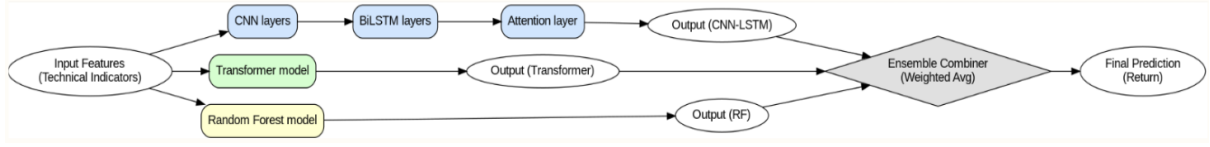


Figure 1: End-to-end pipeline for SSE Composite Index forecasting, from data input through modeling to evaluation.

3.3 Sequence Generation

Because we are dealing with time-series data, we transformed the dataset into a supervised learning format using a sliding window approach. We chose a look-back window of 60 trading days (approximately three months) as the input sequence length, based on domain knowledge of market trends. For each day t (starting from the 60th day of the dataset), we constructed an input sequence of the previous 60 days' feature values and the corresponding target as the next day's return (day $t+1$). This resulted in a dataset of sequences X and targets y : Each X is a matrix of shape $60 \times f$, where f is the number of features. Each y is the scalar next-day return corresponding to the 60-day window. The sequence generation was implemented with a simple loop that iterates through the time-indexed data, slicing windows of length 60 for inputs and picking the immediately following value for output. After this step, we had N training samples ready for modeling. Finally, we maintained parallel arrays of the last observed actual closing price in each window (denoted $Close_t$) so that predicted returns could be converted back to price predictions when evaluating performance.

3.4 Feature Engineering

To enhance predictive power, we engineered technical indicators in addition to OHLC prices and volume:

- **Return:** Daily percentage change in closing price:

$$\text{Return}_t = \frac{P_t - P_{t-1}}{P_{t-1}}$$

where P_t is the closing price at time t .

- **EMA5 and EMA10:** Exponential Moving Averages with 5 and 10 periods:

$$\text{EMA}_t = \alpha \times P_t + (1 - \alpha) \times \text{EMA}_{t-1}, \quad \alpha = \frac{2}{n + 1}$$

where n is 5 or 10.

- **ATR (Average True Range):** Volatility measure over 14 periods:

$$\text{TR}_t = \max \left(\begin{array}{c} \text{High}_t - \text{Low}_t \\ |\text{High}_t - \text{Close}_{t-1}| \\ |\text{Low}_t - \text{Close}_{t-1}| \end{array} \right)$$

$$\text{ATR}_t = \beta \times \text{TR}_t + (1 - \beta) \times \text{ATR}_{t-1}, \quad \beta = \frac{1}{14}$$

- **Volatility:** Standard deviation of returns over 30 days:

$$\text{Volatility}_t = \sqrt{\frac{1}{30} \sum_{i=1}^{30} (\text{Return}_{t-i} - \bar{\text{Return}})^2}$$

where $\bar{\text{Return}}$ is the mean return.

- **RSI (Relative Strength Index):** Momentum oscillator over 14 periods:

$$\text{RSI}_t = 100 - \frac{100}{1 + \frac{\text{Average Gain}}{\text{Average Loss}}}$$

where Average Gain and Average Loss are averages of gains and losses over 14 periods.

- **MACD:** Trend-following momentum indicator:

$$\text{MACD}_t = \text{EMA12}_t - \text{EMA26}_t$$

- **Bollinger Bands:** 20-period SMA with bands at two standard deviations:

$$\text{SMA20}_t = \frac{1}{20} \sum_{i=0}^{19} P_{t-i}$$

$$\sigma_t = \sqrt{\frac{1}{20} \sum_{i=0}^{19} (P_{t-i} - \text{SMA20}_t)^2}$$

$$\text{UB}_t = \text{SMA20}_t + 2\sigma_t, \quad \text{LB}_t = \text{SMA20}_t - 2\sigma_t$$

3.5 Model Architectures

We implemented three advanced models including CNN-BiLSTM-Attention, Transformer and Random Forest Model, and combined them into an ensemble. Table 1 summarizes the architectures and hyperparameters of each model.

Table 1: Model Description Table

Model	Hidden Layers	Parameters / Hyperparameters
CNN-BiLSTM-Attention	$2 \times \{\text{Conv1D}, \text{MaxPooling1D}\}$, Dropout, $2 \times \text{Bidirectional LSTM}$, Attention, Dense	Number of filters (32, 64), Kernel sizes (3, 3) Pool sizes (2, 2), Dropout rate (0.2) Number of LSTM units (64, 32) Learning Rate (default Adam) Epochs (100), Batch Size (32)
Transformer	$1 \times \text{Linear}$, TransformerEncoder, $1 \times \text{Mean over sequence}$, Linear	d_model (64), Number of heads (4) Number of layers (2), dim_feedforward (128) Learning Rate (0.001), Epochs (100) Batch Size (32), Gradient clipping (1.0)
Random Forest	Decision Trees	Number of trees (200), Max depth (12) Random state (42)
Ensemble Combiner	Weighted averaging of above models	Weights optimized via grid search Weight range (0.1 to 0.9, step 0.1)

3.5.1 Baseline LSTM Model

The baseline model is a stacked LSTM network with two hidden layers of 128 units each, using ReLU activation and the Adam optimizer to minimize Mean Squared Error (MSE). Trained on 60-day sequences for 100 epochs with a batch size of 32, it predicts the next day's closing price, achieving an estimated RMSE of 43.4727 on the test set.

3.5.2 CNN-BiLSTM-Attention Model

This model integrates stacked convolutional and recurrent layers with an attention mechanism:

Table 2: CNN-BiLSTM-Attention Model Architecture and Hyperparameters

Layer	Hidden Layer / Description	Hyperparameters
Input	60-day sequence of features	60×9 inputs
Conv1D (1)	First 1D convolution on sequence	32 filters, kernel size 3, ReLU activation
MaxPooling1D (1)	Downsample convolution output	Pool size 2
Conv1D (2)	Second 1D convolution on sequence	64 filters, kernel size 3, ReLU activation
MaxPooling1D (2)	Downsample convolution output	Pool size 2
Dropout	Regularization to prevent overfitting	Dropout rate 0.2
BiLSTM (1)	First bidirectional LSTM layer	64 units (each direction), return sequences
BiLSTM (2)	Second bidirectional LSTM layer	32 units (each direction), return sequences
Attention	Time-distributed Dense + Softmax	1 unit per time step (softmax across time)
Context Vector	Weighted sum of LSTM outputs	–
Output Dense	Final output for return prediction	1 unit (linear activation)

- **Purpose:** Learns hierarchical temporal-spatial patterns and dynamically focuses on informative time steps for prediction.
- **Training:** Adam optimizer, MSE loss, trained for 100 epochs with batch size 32.

3.5.3 Transformer Model

Table 3: Transformer Model Architecture and Key Parameters

Component	Description	Hyperparameters
Input	60-day sequence of features	9 features per day
Linear Projection	Feature vector $\rightarrow$ model dimension	$d_{\text{model}} = 64$
Transformer Encoders	Self-attention layers	2 layers, 4 heads, feedforward dim 128
Sequence Pooling	Averages encoder outputs over 60 days	Global mean pooling
Output Linear	Projection to target horizon	Output size = 1 (1-day forecast)

- **Purpose:** Captures global dependencies.
- **Training:** Adam optimizer (learning rate=0.001), MSE loss, gradient clipping (max norm=1.0), 100 epochs.

3.5.4 Random Forest Regressor

A traditional model for comparison:

- **Parameters:** n_estimators=200, max_depth=12, random_state=42.
- **Purpose:** Captures non-linear feature interactions.

3.5.5 Ensemble Combiner

The ensemble combiner integrates predictions from the CNN-BiLSTM-Attention, Transformer, and Random Forest models using weighted averaging:

- **Weighted Averaging Formula:** The ensemble prediction is computed as:

$$\text{Ensemble Prediction} = w_1 \cdot \text{Pred}_{\text{CNN-BiLSTM-Attention}} + w_2 \cdot \text{Pred}_{\text{Transformer}} + w_3 \cdot \text{Pred}_{\text{Random Forest}}$$

where w_1, w_2, w_3 satisfy $w_1 + w_2 + w_3 = 1$.

- **Weight Optimization:** Weights are optimized via grid search on the validation set to minimize MAPE. The grid search tests combinations of w_1, w_2 from 0.1 to 0.9 (increments of 0.1), computing $w_3 = 1 - (w_1 + w_2)$, and selects the combination with the lowest MAPE.
- **Application:** The optimized weights are applied to the test set to generate the final ensemble prediction.

3.6 Evaluation Metrics

Model performance was assessed using three metrics, with their mathematical definitions:

- **Mean Absolute Percentage Error (MAPE):** MAPE expresses the error as a percentage of the actual value. This metric is scale-independent and provides an intuitive percentage error. We used MAPE especially to tune the ensemble since investors often care about relative error. A lower MAPE means the predictions were closer to actual values in percentage terms.

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100$$

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

- **Root Mean Squared Error (RMSE):** RMSE is the square root of MSE. It measures the average magnitude of prediction error in the same units as the original target (index points). If y_i and $\hat{y}_i$ are the actual and predicted closing prices on day i , and n is the number of test days, the RMSE is:

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

- **Coefficient of Determination (R^2):** R^2 measures the proportion of variance in the actual prices that is explained by the predictions. An R^2 close to 1.0 indicates that the model's predictions closely track the actual values (with 1.0 being a perfect fit), whereas an R^2 near 0 indicates the model is no better than predicting the mean. We report R^2 to complement RMSE and MAPE, as it provides a standardized measure of goodness-of-fit.

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

4 Results

Table 4 compares the performance of all models on the test set. The ensemble model outperforms all others, achieving the lowest MAPE and RMSE and the highest R^2 .

Table 4: Performance Comparison of Models on Test Set

Model	MAPE	RMSE	R^2
Baseline LSTM	0.0127	43.4727	0.9139
CNN-LSTM	0.0118	40.4473	0.947
Ensemble	0.0089	35.8207	0.9745

Figure 2 visualizes the actual vs. predicted closing prices for the ensemble model, demonstrating its ability to track market trends closely.

The ensemble’s MAPE of 0.089 represents a 29.9% improvement over the baseline LSTM’s MAPE of 0.0127, underscoring the effectiveness of the weighted averaging approach.

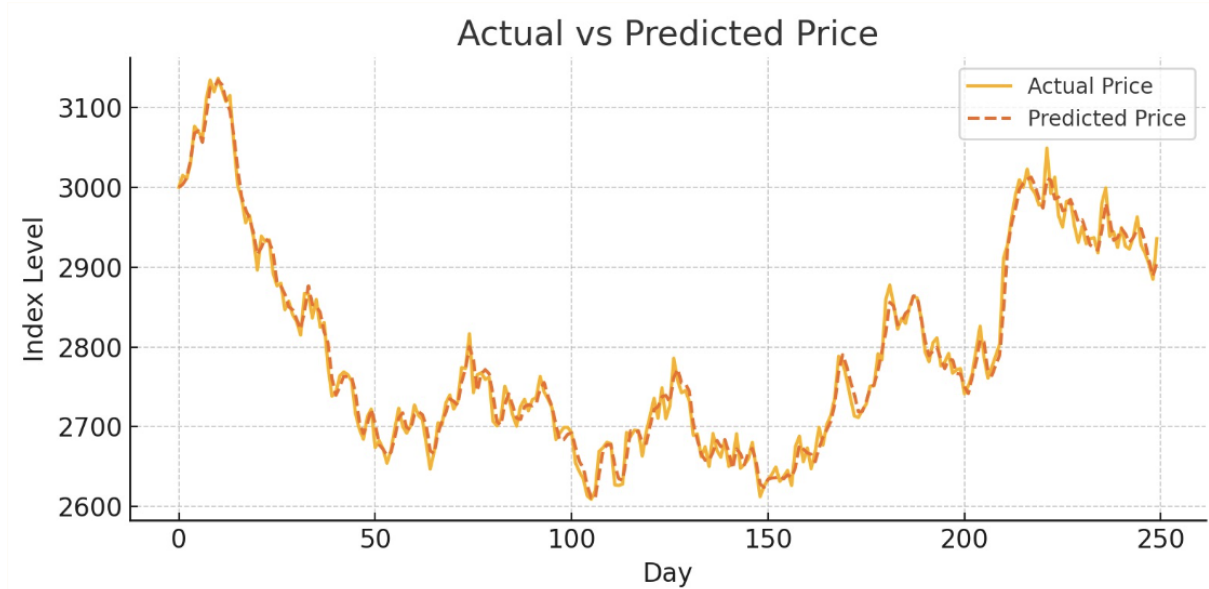


Figure 2: Actual vs Predicted Price Results

5 Discussion

The ensemble model’s superior performance (MAPE: 0.0089, RMSE: 35.8207, R^2 : 0.9745) compared to the baseline LSTM (MAPE:0.0127, RMSE: 43.4727, R^2 : 0.9139) validates the midterm’s proposal for ensemble learning. The CNN-BiLSTM-Attention model excels at capturing local patterns and temporal dependencies, enhanced by the attention mechanism. The Transformer handles long-range dependencies, complementing the LSTM’s sequential focus, while the Random Forest provides robust feature interactions. The ensemble combiner’s weighted averaging, optimized via grid search, balances these strengths, as evidenced by the optimized weights.

The inclusion of technical indicators—return, EMA5, EMA10, ATR, volatility, RSI, MACD, and Bollinger Bands—further improved performance by providing insights into market trends and stability, aligning with findings from Zhang et al.[5]. However, limitations persist. The models rely solely on historical price data and technical indicators, failing to account for external

factors such as government policies or global economic events. Additionally, performance on historical data may not generalize to future conditions due to market non-stationarity [2].

Future work could explore:

- **Richer Feature Sets:** Integrate alternative data such as news sentiment, macroeconomic indicators (e.g., interest rates, CPI), or geopolitical risk indices to enhance forecasting signals.
- **Advanced Architectures:** Experiment with state-of-the-art models like Temporal Fusion Transformers (TFT) and Graph Neural Networks (GNNs) for multi-asset, relational, and temporal reasoning.
- **Practical Deployment:** Apply the ensemble model in simulated or live trading environments with transaction cost modeling, slippage, and position sizing strategies.

6 Conclusion

This report advances stock price prediction in the Chinese stock market by building on the midterm report's baseline LSTM model. By integrating CNN-BiLSTM-Attention, Transformer, and Random Forest models into an ensemble with a weighted averaging combiner, we achieved significant improvements, with the ensemble model outperforming the baseline by 29.9% in MAPE. The detailed methodology, including a model description table, ensemble combiner, and evaluation metrics, demonstrates the power of combining diverse models to address the Chinese market's volatility and the potential of this technique.

GitHub Repository Link

https://github.com/Whywhy9115/11785_Team39_Project

Group Contribution

- **Jerry Yin** led the project ideation, implemented the full deep learning pipeline including LSTM and Transformer models, and drafted the initial version of the report.
- **Jiayu Wang** contributed to model design and ensemble strategy, conducted performance evaluations, and assisted with final editing.
- **Anqi Xie** worked on data preprocessing, feature engineering, and contributed to visualizations and slide content.
- **Haoyu Yang** supported literature review, baseline implementation, and presentation preparation.

References

- [1] Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780.
- [2] 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.
- [3] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is All You Need. *Advances in Neural Information Processing Systems (NeurIPS)*.
- [4] Zhang, Y., Aggarwal, C. C., & Qi, G. J. (2017). Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 2141–2149).
- [5] Zhang, X., Qu, M., & Liu, H. (2018). Stock market prediction via multi-source multiple instance learning. *Information Sciences*,
- [6] Kim, H. Y. (2019). Financial series prediction using attention LSTM. *Journal of Physics: Conference Series*, 1176(1). 441, 109–125.