electronics

Article

A Multi-Feature Stock Index Forecasting Approach Based on LASSO Feature Selection and Non-Stationary Autoformer

Zibin Sheng [1], Qingyang Liu [2,*], Yanrong Hu [1,*] and Hongjiu Liu [1,*]

[1] College of Mathematics and Computer Science, Zhejiang A & F University, Hangzhou 311300, China; zibinsheng@stu.zafu.edu.cn
[2] Institute of Informatics, Georg-August-Universität Göttingen, 37073 Göttingen, Germany
* Correspondence: qingyang.liu@stud.uni-goettingen.de (Q.L.); yanrong_hu@zafu.edu.cn (Y.H.); joe_hunter@zafu.edu.cn (H.L.)

Abstract: The Chinese stock market, one of the largest and most dynamic emerging markets, is characterized by individual investor dominance and strong policy influence, resulting in high volatility and complex dynamics. These distinctive features pose substantial challenges for accurate forecasting. Existing models like RNNs, LSTMs, and Transformers often struggle with non-stationary data and long-term dependencies, limiting their forecasting effectiveness. This study proposes a hybrid forecasting framework integrating the Non-stationary Autoformer (NSAutoformer), LASSO feature selection, and financial sentiment analysis. LASSO selects key features from diverse structured variables, mitigating multicollinearity and enhancing interpretability. Sentiment indices are extracted from investor comments and news articles using an expanded Chinese financial sentiment dictionary, capturing psychological drivers of market behavior. Experimental evaluations on the Shanghai Stock Exchange Composite Index show that LASSO-NSAutoformer outperforms the NSAutoformer, reducing MAE by 8.75%. Additional multi-step forecasting and time-window analyses confirm the method's effectiveness and stability. By integrating multi-source data, feature selection, and sentiment analysis, this framework offers a reliable forecasting approach for investors and researchers in complex financial environments.

Keywords: stock price forecast; Autoformer; financial sentiment analysis; feature selection; time series forecasting

Academic Editors: Ping-Feng Pai and Krzysztof Szczypiorski

Received: 3 April 2025
Revised: 15 May 2025
Accepted: 15 May 2025
Published: 19 May 2025

Citation: Sheng, Z.; Liu, Q.; Hu, Y.; Liu, H. A Multi-Feature Stock Index Forecasting Approach Based on LASSO Feature Selection and Non-Stationary Autoformer. *Electronics* **2025**, *14*, 2059. https://doi.org/10.3390/electronics14102059

1. Introduction

The Chinese stock market has grown rapidly, attracting investors seeking to anticipate trends for financial gain. However, stock price volatility complicates forecasting and amplifies investment risks. Accurate predictions are essential for both individuals and organizations, guiding investment decisions and risk management. For investors, forecasts support decisions on buying, selling, or applying stop-loss strategies. For organizations, reliable forecasts support trading, asset allocation, and portfolio management [1]. Yet, stock price movements are inherently complex and nonlinear [2].

As financial time series data, stock prices are highly volatile and subject to random noise, making prediction particularly challenging [3]. Over the years, researchers have explored various forecasting approaches. Traditional statistical models, such as Moving Average (MA) [4], Auto-Regressive Moving Average (ARMA) [5], and Auto-Regressive Integrated Moving Average (ARIMA) [6], as well as volatility models like Auto-Regressive Conditional Heteroskedasticity (ARCH) [7] and Generalized Auto-Regressive Conditional

Heteroskedasticity (GARCH) [8], have been widely applied. Among them, ARIMA performs well on time series with clear trends and seasonality. However, ARIMA assumes smooth data, limiting its ability to model highly volatile stock prices [9]. These models depend on predefined variable selection and linear assumptions, making it difficult to capture complex, nonlinear patterns in large-scale, high-dimensional financial data. Traditional econometric methods cannot effectively handle unstructured alternative data, leading to a growing shift toward machine learning approaches.

Machine learning methods such as K-Nearest Neighbor (KNN) [10], Support Vector Machine (SVM) [11], and random forests [12] have shown promise in modeling nonlinear relationships and complex market dynamics. Nevertheless, these approaches often encounter challenges such as feature selection and overfitting. To further improve forecasting performance and autonomously extract intricate patterns from financial time series, researchers have turned to deep learning techniques. Models such as Artificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs) [13], Recurrent Neural Networks (RNNs) [14], Long Short-Term Memory (LSTM) networks [15], and Gated Recurrent Units (GRUs) [16] have been widely explored. While RNNs capture temporal dependencies, they suffer from gradient instability. LSTM and GRU alleviate this issue but still face challenges such as overfitting and high training complexity.

Since Vaswani et al. introduced the Transformer in 2017 [17,18], Transformer-based models have been widely used in time series forecasting with notable success. Based on attention mechanisms, the Transformer effectively captures long-term dependencies in sequential data, addressing the limitations of traditional RNN-based models. In 2021, Zhou et al. proposed the Informer model, which optimized the attention mechanism, mitigating long-term information loss [19]. Following these advancements, researchers have further refined Transformer-based architectures and applied them to stock price prediction [20].

The non-stationary and highly volatile nature of financial markets poses considerable challenges for accurate forecasting. Traditional models often fail to capture the complex, nonlinear structures embedded in stock price movements. To address this, we propose the Non-stationary Autoformer (NSAutoformer), which extends the Non-stationary Transformers framework to the Autoformer [21,22], enhancing adaptability to evolving financial data. The model enhances adaptability by decomposing time series into trend and seasonal components and adjusting to distributional shifts. To further improve forecasting, we incorporate the LASSO algorithm and sentiment analysis, forming a multi-source data approach. The LASSO-NSAutoformer is empirically validated on the Chinese stock market index, showing stable, reliable, and superior predictive performance.

The Chinese stock market is one of the largest and most dynamic emerging markets, with a high proportion of individual investors, making it highly sensitive to sentiment. Compared to developed markets, Chinese individual investors are generally less experienced and prone to herd behavior, often reacting to media reports and peer suggestions with panic buying or selling. Moreover, government policies exert strong influence on the market. The active online environment, including financial social media and investor forums, provides a wealth of sentiment-rich data, supporting broad coverage for this study. These characteristics offer a unique opportunity to evaluate our proposed approach in a complex, sentiment-sensitive market.

The proposed framework comprises three main stages. First, the LASSO algorithm selects relevant features from 49 structured variables, including historical prices, trading data, technical indicators, composite indices, and other market information. Second, a Chinese financial sentiment dictionary is used to analyze investor comments and news, producing two sentiment indices incorporated into the forecasting model. Finally, hyperparameters

are tuned using Optuna to improve model performance. Comparative experiments show that the proposed method outperforms traditional models, including RNN, GRU, LSTM, and Transformer-based architectures. Notably, LASSO-based feature selection enhances accuracy by identifying key predictors, reducing overfitting and computational complexity. Additionally, the sentiment indices capture investor psychology, which correlates with market fluctuations. By analyzing key factor interactions, this study provides deeper insight into market dynamics. Combining fundamental and sentiment-driven variables allows the model to more accurately reflect real-world market behavior and improve forecasting performance.

The main work accomplished in the article is as follows:

1. This study incorporates multiple factors, including trading data, technical indicators, geopolitical events, epidemics, public sentiment, and cross-market influences, with a focus on sentiment analysis. A Chinese financial sentiment dictionary is constructed to extract sentiment from investor comments and news articles.
2. We propose NSAutoformer by extending the Non-stationary Transformers framework to the Autoformer architecture. This architecture retains the series decomposition mechanism and improves the ability to model non-stationary patterns in financial time series.
3. We further propose LASSO-NSAutoformer, a hybrid model that integrates LASSO-based feature selection and financial sentiment analysis. Comparative experiments demonstrate that it outperforms existing methods across various Chinese stock market indices.

The remainder of this paper is organized as follows. Section 2 provides a comprehensive literature review on the topic. Section 3 introduces the relevant models and algorithms, and elaborates on the structure of our proposed framework. Section 4 outlines the procedure of the experiment, including data collection, preprocessing, and parameter setting. Section 5 shows the results and analysis of a series of experiments in detail. Section 6 concludes the study and provides an assessment of possible future work.

2. Related Works

Stock price forecasting is a complex task, typically approached through three perspectives: fundamental analysis, technical analysis, and market sentiment. Fundamental analysis focuses on a company's financial health, operational performance, industry standing, and macroeconomic environment. Technical analysis leverages historical stock prices, trading volumes, and technical indicators to predict future price movements. Sentiment analysis captures investor attitudes and emotions, which often influence market trends. For these different perspectives, researchers have explored various deep learning-based methods.

2.1. Fundamental Analysis

Fundamental analysis assesses a company's intrinsic value to forecast stock prices. It relies on financial statements, profitability, price-to-earnings (P/E) ratio, and dividend yield. For market indices, relevant factors include exchange rates, commodity prices, and interest rates. This approach is widely applied in trading and portfolio management. Recent studies have incorporated advanced computational techniques into fundamental analysis. For instance, Nourbakhsh and Habibi proposed a hybrid model combining LSTM, CNN, and fundamental indicators (e.g., P/E ratio, profitability), achieving low prediction error in stock trend forecasting [23]. Similarly, Quadir et al. introduced a multi-layer sequential LSTM (MLS-LSTM) optimized with the Adam algorithm, achieving 98.1% accuracy on test data by analyzing historical trends and patterns [24]. Xu and Liu employed a Bi-LSTM Attention model to examine how major events, such as COVID-19

and the Russia–Ukraine conflict, influenced the volatility of crude oil futures, revealing dynamic market responses [25].

2.2. Technical Analysis

Technical analysis uses historical price data and technical indicators to identify trends and trading opportunities. Traditionally, stock forecasting relies on core price and volume indicators, including open, high, low, close, and volume (OHLCV) [26]. These indicators capture key market dynamics and are widely used in academic research [27]. Li et al. proposed MSFCE, a Transformer-based model designed for stock trend prediction. It integrates multi-scale feature encoding and graph attention to capture interactions among technical indicators [28]. Lin et al. examined technical and online sentiment measures for forecasting stock volatility, confirming that technical indicators remain the more reliable predictor [29].

Feature selection improves machine learning performance by identifying key variables and reducing multicollinearity. Htun et al. surveyed feature selection and extraction techniques used in stock market prediction [30]. They identified correlation analysis, random forests, principal component analysis (PCA), and autoencoders as the most effective and commonly used techniques. Feature selection is critical in stock prediction, as it removes irrelevant variables, reduces computation, mitigates overfitting, and enhances model accuracy.

2.3. Market Sentiment Analysis

Sentiment analysis is vital in stock forecasting as it captures investor emotions and identifies irrational behavior. Emotions like panic, greed, and optimism can significantly affect market movements. Quantifying these emotions helps interpret stock price fluctuations. Many studies have confirmed the effectiveness of sentiment analysis in financial forecasting. For example, De Oliveira Carosia et al. optimized ANN structures for analyzing sentiment in Brazilian Portuguese financial news and developed sentiment-based investment strategies [31]. Ji et al. developed an attention-based LSTM (ALSTM) model, incorporating price data, technical indicators, and social media sentiment. Results showed that combining sentiment features with technical indicators improved accuracy, with a 5-day input window performing best [32]. Liu et al. introduced SA-TrellisNet, which integrates news sentiment analysis and a sentiment attention mechanism into a TrellisNet-based model, demonstrating strong performance in index forecasting [33].

3. Methodology

This section introduces NSAutoformer, which extends the Non-stationary Transformers framework to the Autoformer architecture. In addition, we present a forecasting framework that integrates sentiment analysis and LASSO-based feature selection.

3.1. LASSO

LASSO (Least Absolute Shrinkage and Selection Operator) [34] is a regression method designed to enhance model interpretability, particularly in high-dimensional settings. It introduces L_1 regularization to shrink coefficients and perform feature selection simultaneously.

First, define $X_{n\times k}$ to be the matrix of explanatory variables that characterize the model inputs. $Y_{n\times 1}$ is the response variable that characterizes the model outputs. Its mathematical form can be expressed as follows:

$$Y = X\beta + \epsilon \tag{1}$$

where β is a vector of coefficients to be estimated, each corresponding to an input feature. ϵ is an error term, accounting for random noise unexplained by the model.

Conventional linear regression fits the data by minimizing the residual sum of squares (RSS). However, in high-dimensional settings, it often results in overfitting and poor interpretability due to the inclusion of irrelevant features. To mitigate this, LASSO adds an L_1-norm penalty to the loss function, encouraging sparsity in the model coefficients.

$$\min_{\beta} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{k} x_{ij}\beta_j \right)^2 \text{ subject to } \sum_{j=1}^{k} |\beta_j| < t \tag{2}$$

where t is a pre-set threshold to control the complexity of the model. This constraint motivates the model to fit the data while limiting the size of the coefficients, thus avoiding overfitting.

The problem can be reduced to the following vector:

$$\min_{\beta} \frac{1}{n} \|\mathrm{Y} - \mathrm{X}\beta\|_2^2 \text{ subject to } \sum_{j=1}^{k} |\beta_j| < t \tag{3}$$

To simplify computation and unify the objective, LASSO introduces a penalty term λ, transforming the constrained problem into an unconstrained one.

$$\hat{\beta}(\lambda) = \arg\min_{\beta} \left(\frac{1}{n} \parallel \mathrm{Y} - \mathrm{X}\beta \parallel_2^2 + \lambda \sum_{j=1}^{k} |\beta_j| \right) \tag{4}$$

In this formula, $\frac{1}{n} \parallel \mathrm{Y} - \mathrm{X}\beta \parallel_2^2$ denotes the standardized residual sum of squares, indicating model fit. $\lambda \sum_{j=1}^{k} |\beta_j|$ is the L_1 regularization term and λ is the hyperparameter that controls the strength of the regularization. By adjusting λ, LASSO can find an optimal balance between model complexity and fitting effectiveness. A larger λ increases regularization, shrinking more coefficients to zero and enabling feature selection; a smaller λ prioritizes data fit.

Therefore, LASSO is a commonly used and effective method for feature selection. In the context of economic and financial data, it helps identify key indicators and variables for constructing forecasting models.

3.2. Non-Stationary Transformers

Transformers have been widely used in time series forecasting due to their ability to capture long-range temporal dependencies [35–39]. However, real-world time series are often non-stationary, exhibiting time-varying means and variances, which can degrade the performance of standard attention mechanisms. Although traditional smoothing techniques such as differencing and moving averages can alleviate this issue, they often compromise the model's ability to retain structural information. To address these limitations, we introduce the Non-stationary Transformers framework [22], a lightweight and generalizable architecture for modeling non-stationary time series. This framework forms the theoretical basis for our method, which extends its core mechanisms to the Autoformer architecture.

3.2.1. Series Stationarization

As shown in Figure 1, a normalization module and a de-normalization module perform the processing of the input and output data, respectively. By transforming the sequence data with these two modules, the model maintains better predictability. It is assumed that the input data $\mathrm{x} = [x_1, x_2, \ldots, x_L]^\top \in \mathbb{R}^{L \times N}$ are processed as $\mathrm{x}' = [x_1', x_2', \ldots, x_L']^\top \in \mathbb{R}^{L \times N}$,

where L and N denote the input data length and number of variables. The formula can be expressed as follows:

$$\mu_{\mathrm{x}} = \frac{1}{L}\sum_{i=1}^{L} x_i, \sigma_{\mathrm{x}}^2 = \frac{1}{L}\sum_{i=1}^{L}(x_i - \mu_{\mathrm{x}})^2, x_i' = \frac{1}{\sigma_{\mathrm{x}}} \odot (x_i - \mu_{\mathrm{x}}) \tag{5}$$

where $\mu_{\mathrm{x}}, \sigma_{\mathrm{x}} \in \mathbb{R}^{N\times 1}$, $\frac{1}{\sigma_{\mathrm{x}}}$ denotes the element-wise division, and $\odot$ denotes the element-wise product.

It is assumed that the length of the prediction is S and the output data $\mathrm{y}' = [y_1', y_2', \ldots, y_S']^\top \in \mathbb{R}^{S\times N}$. Processing through the de-normalization module results in $\hat{\mathrm{y}} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_S]^\top$. The formula can be expressed as follows:

$$\mathrm{y}' = \mathrm{model}(\mathrm{x}'), \hat{y}_i = \sigma_{\mathrm{x}} \odot (y_i' + \mu_{\mathrm{x}}) \tag{6}$$

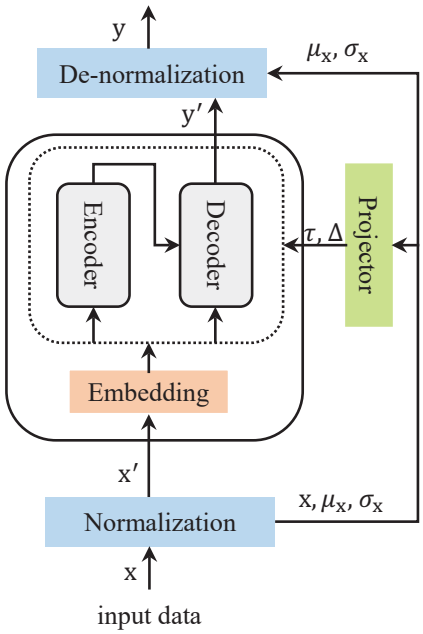


Figure 1. The architecture of the Series Stationarization module.

3.2.2. De-Stationary Attention

This section introduces an improved attention mechanism to better capture temporal dependencies in non-stationary time series data. The architecture of the modified attention mechanism is illustrated in Figure 2.

The model receives the normalized input $\mathrm{x}' = (\mathrm{x} - \mathbf{1}\mu_{\mathrm{x}}^\top)/\sigma_{\mathrm{x}}$ and subsequently computes the new Q', K', V'. The query is represented as $Q' = [f(x_1'), \ldots, f(x_L')]^\top = (Q - \mathbf{1}\mu_Q^\top)/\sigma_{\mathrm{x}}$, where $f(x_i)$ denotes feedforward layer, and $\mathbf{1} \in \mathbb{R}^{L\times 1}$ is an all-one vector. The transformations of $K'$and V' are similar to Q'. Then, a multilayer perceptron is used to learn the de-stationary factors τ, Δ from the statistics μ_{x}, and σ_{x} of the unstationarized original time series x. Finally, the new formula for attention is as follows:

$$\log \tau = \mathrm{MLP}(\sigma_{\mathrm{x}}, \mathrm{x}), \Delta = \mathrm{MLP}(\mu_{\mathrm{x}}, \mathrm{x}) \tag{7}$$

$$\mathrm{Attention}(Q', K', V', \tau, \Delta) = \mathrm{Softmax}\left(\frac{\tau Q' K'^\top + \mathbf{1}\Delta^\top}{\sqrt{d_k}}\right) V' \tag{8}$$

where $\tau = \sigma_{\mathrm{x}}^2 \in \mathbb{R}^+$and $\Delta = K\mu_Q \in \mathbb{R}^{L\times 1}$ represent the positive scale scalar and shifting vector. Now, the attention mechanism combines the information of the original series with

the attention information of the stationarized series. It can improve the predictability of the non-stationary series while maintaining the temporal dependencies of the original series.

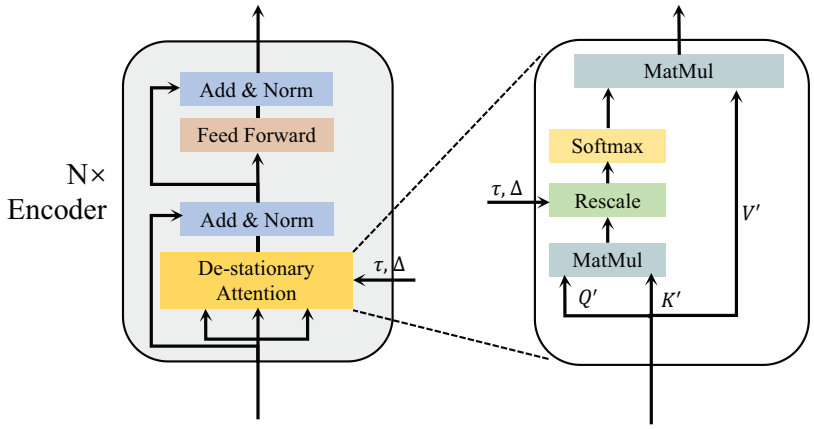


Figure 2. The architecture of the De-stationary Attention.

3.3. Autoformer

The Autoformer enhances the Transformer architecture by introducing a novel self-attention mechanism, improving both computational efficiency and forecasting accuracy [21]. It employs an Auto-Correlation mechanism to identify dependencies across series and effectively aggregate information, capturing both local and global patterns. It integrates time series decomposition with a progressive decomposition module to overcome limitations of traditional preprocessing-based methods. The model consists of three core components: Decomposition Block, Auto-Correlation Mechanism, and Encoder–Decoder Architecture. Its structure is illustrated in Figure 3.

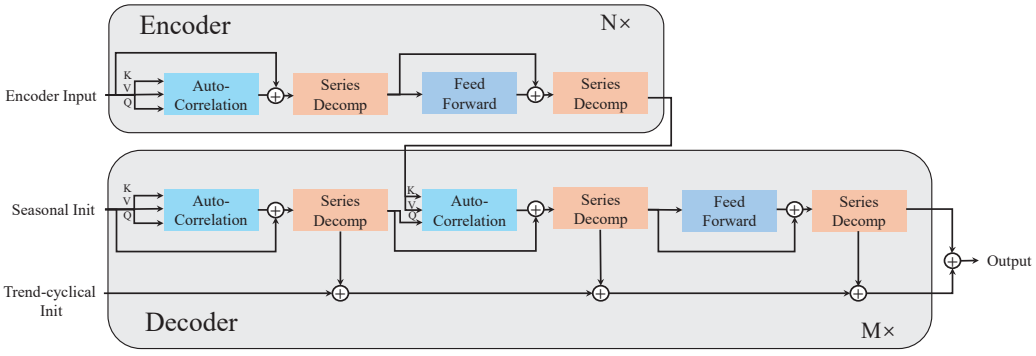


Figure 3. The architecture of the Autoformer.

3.3.1. Decomposition Block

The Autoformer model's series decomposition block splits time series into trend and seasonal components. The block maintains time series length with padding and extracts seasonal terms by subtracting trend terms. To effectively extract the trend information, a moving average method is used. For series $\mathcal{X} \in \mathbb{R}^{L \times d}$ of length L, the processing results in the following series:

$$\mathcal{X}_{\mathrm{t}} = \mathrm{AvgPool}(\mathrm{Padding}(\mathcal{X})) \tag{9}$$

$$\mathcal{X}_{\mathrm{s}} = \mathcal{X} - \mathcal{X}_{\mathrm{t}} \tag{10}$$

where $\mathcal{X}_{\mathrm{s}}, \mathcal{X}_{\mathrm{t}} \in \mathbb{R}^{L \times d}$ denote the seasonal part and the extracted trend part, respectively. The equation is summarized by $\mathcal{X}_{\mathrm{s}}, \mathcal{X}_{\mathrm{t}} = \mathrm{SeriesDecomp}(\mathcal{X})$.

The inputs to the encoder are series $\mathcal{X}_{\text{en}} \in \mathbb{R}^{I \times d}$ of length I time steps in the past, and the input to the decoder consists of a seasonal part $\mathcal{X}_{\text{des}} \in \mathbb{R}^{(\frac{I}{2}+O) \times d}$ and a trend part $\mathcal{X}_{\text{det}} \in \mathbb{R}^{(\frac{I}{2}+O) \times d}$, O denotes the padding length serving as a placeholder for future predictions.

$$\mathcal{X}_{\text{ens}}, \mathcal{X}_{\text{ent}} = \text{SeriesDecomp}\left(\mathcal{X}_{\text{en}\frac{I}{2}:I}\right) \tag{11}$$

$$\mathcal{X}_{\text{des}} = \text{Concat}(\mathcal{X}_{\text{ens}}, \mathcal{X}_0) \tag{12}$$

$$\mathcal{X}_{\text{det}} = \text{Concat}(\mathcal{X}_{\text{ent}}, \mathcal{X}_{\text{Mean}}) \tag{13}$$

where $\mathcal{X}_{\text{ens}}$ and $\mathcal{X}_{\text{ent}} \in \mathbb{R}^{\frac{I}{2} \times d}$ represent the seasonal and trend-cyclical components of $\mathcal{X}_{\text{en}}$ respectively. The matrices $\mathcal{X}_0$ and $\mathcal{X}_{\text{Mean}} \in \mathbb{R}^{O \times d}$ are placeholder sequences filled with zeros and the mean value of $\mathcal{X}_{\text{en}}$ respectively.

3.3.2. Encoder and Decoder

The encoder module is concerned with the modeling of the seasonal part, and the output contains the past seasonal information, which will be used as mutual information to help the decoder adjust the forecasting results. Assuming we have N encoder layers, the l-th encoder layer $\mathcal{X}_{\text{en}}^l = \text{Encoder}(\mathcal{X}_{\text{en}}^{l-1})$ has the following internal structure:

$$\mathcal{S}_{\text{en}}^{l,1}, \_ = \text{SeriesDecomp}\left(\text{Auto} - \text{Correlation}(\mathcal{X}_{\text{en}}^{l-1}) + \mathcal{X}_{\text{en}}^{l-1}\right) \tag{14}$$

$$\mathcal{S}_{\text{en}}^{l,2}, \_ = \text{SeriesDecomp}\left(\text{FeedForward}(\mathcal{S}_{\text{en}}^{l,1}) + \mathcal{S}_{\text{en}}^{l,1}\right) \tag{15}$$

where "$\_$" is the portion of the trend that was excluded. $\mathcal{X}_{\text{en}}^l = \mathcal{S}_{\text{en}}^{l,2}, l \in \{1, \cdots, N\}$ denotes the output of the l-th encoder layer, and $\mathcal{X}_{\text{en}}^0$ is the embedded $\mathcal{X}_{\text{en}}$. $\mathcal{S}_{\text{en}}^{l,i}, i \in \{1, 2\}$ denotes the seasonal component after the i-th series of the decomposition block in the l-th layer. Finally, $\text{Auto} - \text{Correlation}(\cdot)$ here replaces self-attention.

The decoder consists of two parts: a cumulative operation for trend components and a stacked autocorrelation mechanism for seasonal components. Assuming that there are M decoding layers, using the latent variables $\mathcal{X}_{\text{en}}^N$ from the encoder, the equations for the l-th decoder layer can be summarized as $\mathcal{X}_{\text{de}}^l = \text{Decoder}\left(\mathcal{X}_{\text{de}}^{l-1}, \mathcal{X}_{\text{en}}^N\right)$. The internal details are as follows:

$$\mathcal{S}_{\text{de}}^{l,1}, \mathcal{T}_{\text{de}}^{l,1} = \text{SeriesDecomp}\left(\text{Auto} - \text{Correlation}(\mathcal{X}_{\text{de}}^{l-1}) + \mathcal{X}_{\text{de}}^{l-1}\right) \tag{16}$$

$$\mathcal{S}_{\text{de}}^{l,2}, \mathcal{T}_{\text{de}}^{l,2} = \text{SeriesDecomp}\left(\text{Auto} - \text{Correlation}(\mathcal{S}_{\text{de}}^{l,1}, \mathcal{X}_{\text{en}}^N) + \mathcal{S}_{\text{de}}^{l,1}\right) \tag{17}$$

$$\mathcal{S}_{\text{de}}^{l,3}, \mathcal{T}_{\text{de}}^{l,3} = \text{SeriesDecomp}\left(\text{FeedForward}(\mathcal{S}_{\text{de}}^{l,2}) + \mathcal{S}_{\text{de}}^{l,2}\right) \tag{18}$$

$$\mathcal{T}_{\text{de}}^l = \mathcal{T}_{\text{de}}^{l-1} + \mathcal{W}_{l,1} * \mathcal{T}_{\text{de}}^{l,1} + \mathcal{W}_{l,2} * \mathcal{T}_{\text{de}}^{l,2} + \mathcal{W}_{l,3} * \mathcal{T}_{\text{de}}^{l,3} \tag{19}$$

where $\mathcal{X}_{\text{de}}^l = \mathcal{S}_{\text{de}}^{l,3}, l \in \{1, \cdots, M\}$ denotes the output of the l-th decoder layer. $\mathcal{X}_{\text{de}}^0$ embedding from $\mathcal{X}_{\text{des}}$ is used for deep transform and $\mathcal{T}_{\text{de}}^0 = \mathcal{X}_{\text{det}}$ is used for accumulation. $\mathcal{S}_{\text{de}}^{l,i}, \mathcal{T}_{\text{de}}^{l,i}, i \in \{1, 2, 3\}$ denote the seasonal component and the trend component following the i-th series of decomposition block in l-th layer, respectively. $\mathcal{W}_{l,i}, i \in \{1, 2, 3\}$ denotes the i-th extracted trend $\mathcal{T}_{\text{de}}^{l,i}$ of the projection layer. Finally, the predicted value, as $\mathcal{W}_{\mathcal{S}} * \mathcal{X}_{\text{de}}^M + \mathcal{T}_{\text{de}}^M$, is obtained from the sum of the two decomposition components, where $\mathcal{W}_{\mathcal{S}}$ projects the seasonal component $\mathcal{X}_{\text{de}}^M$ of the deep transformation to the target dimension.

3.3.3. Autocorrelation Mechanism

The autocorrelation module, as demonstrated in Figure 4, is designed to identify period-based dependencies. It achieves this by calculating the autocorrelation coefficients of the series. Then, it rolls the series to perform time-delay aggregation of similar sub-series. The correlation between the original series and its lagged series is calculated. This helps identify periodically similar sub-series.

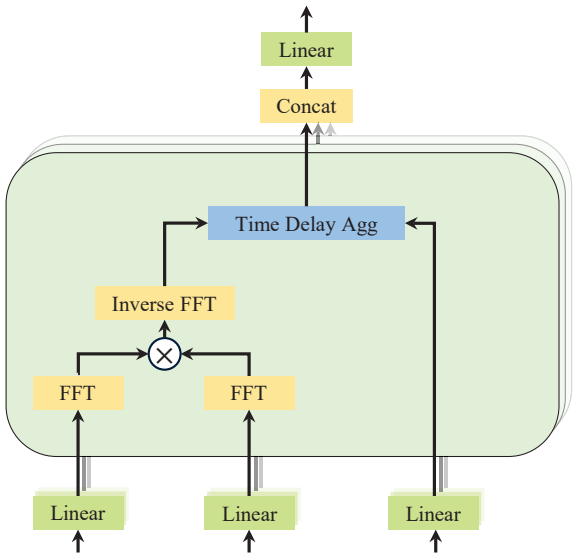


Figure 4. Diagram of autocorrelation mechanism.

$$\mathcal{R}_{\mathcal{X}\mathcal{X}}(\tau) = \lim_{L\to\infty} \frac{1}{L} \sum_{t=1}^{L} \mathcal{X}_t \mathcal{X}_{t-\tau} \tag{20}$$

For a real discrete-time process $\{\mathcal{X}_t\}$, the autocorrelation $\mathcal{R}_{\mathcal{X}\mathcal{X}}(\tau)$ can be obtained by the equation. It quantifies the time-delay similarity between $\{\mathcal{X}_t\}$ and its lagged counterpart $\{\mathcal{X}_{t-\tau}\}$. By selecting the top-k delays with the highest autocorrelation scores, dominant periodic patterns in the sequence can be identified. To utilize these patterns, a time-delay aggregation mechanism is used to aligns sub-series at the same phase positions across the estimated periods. Unlike conventional dot-product attention, this method explicitly captures periodic dependencies and performs weighted aggregation using softmax-normalized autocorrelation scores.

For the single-head case and time series $\mathcal{X}$ of length L, after the projection layer (projector), the query $\mathcal{Q}$, the key $\mathcal{K}$, and the value $\mathcal{V}$ can be obtained. Thus, it can seamlessly replace the self-attention mechanism.

$$\tau_1, \cdots, \tau_k = \underset{\tau\in\{1,\cdots,L\}}{\arg \text{Topk}} \left(\mathcal{R}_{\mathcal{Q},\mathcal{K}}(\tau)\right) \tag{21}$$

$$\widehat{\mathcal{R}}_{\mathcal{Q},\mathcal{K}}(\tau_1), \cdots, \widehat{\mathcal{R}}_{\mathcal{Q},\mathcal{K}}(\tau_k) = \text{SoftMax}(\mathcal{R}_{\mathcal{Q},\mathcal{K}}(\tau_1), \cdots, \mathcal{R}_{\mathcal{Q},\mathcal{K}}(\tau_k)) \tag{22}$$

$$\text{Auto} - \text{Correlation}(\mathcal{Q}, \mathcal{K}, \mathcal{V}) = \sum_{i=1}^{k} \text{Roll}(\mathcal{V}, \tau_i) \widehat{\mathcal{R}}_{\mathcal{Q},\mathcal{K}}(\tau_i) \tag{23}$$

where $k = \lfloor c \times \log L \rfloor$ in the $\arg \text{Topk}(\cdot)$ operator, and c is the hyperparameter. The autocorrelation between the series $\mathcal{Q}$ and $\mathcal{K}$ is denoted as $\mathcal{R}_{\mathcal{Q},\mathcal{K}}$. $\text{Roll}(\mathcal{X}, \tau)$ is an operation on $\mathcal{X}$ with time delay τ, and the element moved out of the first position will be reintroduced in the last position. In the encoder–decoder autocorrelation, $\mathcal{K}$ and $\mathcal{V}$ come from the encoder $\mathcal{X}_{\text{en}}^{N}$ and are resized to length O. $\mathcal{Q}$ is from the previous block of the decoder.

In order to improve the computational efficiency, the model is based on the Winer–Khinchin theory and the autocorrelation coefficients are computed using the Fast Fourier Transform (FFT):

$$\mathcal{S}_{\mathcal{X}\mathcal{X}}(f) = \mathcal{F}(\mathcal{X}_t)\mathcal{F}^*(\mathcal{X}_t) = \int_{-\infty}^{\infty} \mathcal{X}_t e^{-i2\pi tf} \mathrm{d}t \overline{\int_{-\infty}^{\infty} \mathcal{X}_t e^{-i2\pi tf} \mathrm{d}t} \tag{24}$$

$$\mathcal{R}_{\mathcal{X}\mathcal{X}}(\tau) = \mathcal{F}^{-1}(\mathcal{S}_{\mathcal{X}\mathcal{X}}(f)) = \int_{-\infty}^{\infty} \mathcal{S}_{\mathcal{X}\mathcal{X}}(f) e^{i2\pi f\tau} \mathrm{d}f \tag{25}$$

where $\tau \in \{1, \cdots, L\}$, $\mathcal{F}$ denotes the FFT, and * denotes the conjugate operation. This is achieved by converting the signal to frequency $\mathcal{S}_{\mathcal{X}\mathcal{X}}(f)$ and then calculating $\mathcal{R}_{\mathcal{X}\mathcal{X}}(\tau)$ through the frequency domain.

3.4. Non-Stationary Autoformer

Non-stationary Autoformer is a novel architecture designed for non-stationary time series forecasting. It extends the Autoformer backbone by incorporating key components from the Non-stationary Transformers framework.

Compared to Autoformer, NSAutoformer introduces a normalization-based preprocessing module. This module stabilizes the input distribution by extracting the mean and standard deviation of the input sequence. These statistics are subsequently fed into multilayer perceptrons to produce two learnable de-stationary factors: a global scaling factor (τ) and a temporal shifting vector (Δ). These de-stationary factors are then injected into the autocorrelation mechanism to modulate temporal dependency modeling.

Instead of directly relying on raw autocorrelation values, De-stationary autocorrelation modulates the correlation scores using these adjustment factors. This design allows the model to better capture evolving patterns and distributional shifts over time. The time-delay aggregation process follows that of Autoformer. NSAutoformer integrates non-stationary-aware attention with the efficient autocorrelation structure. At the same time, it significantly improves robustness to time-varying distributions.

By integrating these modules, NSAutoformer enhances the adaptability of the baseline to non-stationary environments. It retains the ability to model trend and seasonal components through decomposition. As a result, the model can better capture complex, time-varying dynamics in real-world time series. It ensures stable and accurate forecasting of non-stationary data across domains such as finance, energy, and macroeconomics.

3.5. Sentiment Analysis

In finance, sentiment analysis is primarily conducted using sentiment dictionaries, machine learning, and deep learning [40,41]. Sentiment dictionaries offer a straightforward and efficient method for analyzing financial texts rich in domain-specific terms. As our experiments used unlabeled social media texts, sentiment dictionaries were a suitable choice. They do not require labeled data, thus reducing both time and cost. Moreover, investor comments tended to be consistent and repetitive, allowing a fixed vocabulary list to classify sentiments effectively with low computational cost.

To improve sentiment analysis, we developed a financial sentiment dictionary by extending an existing lexicon to better capture financial terms and sentiment nuances. The processes of sentiment categorization and feature construction are illustrated in Figure 5.

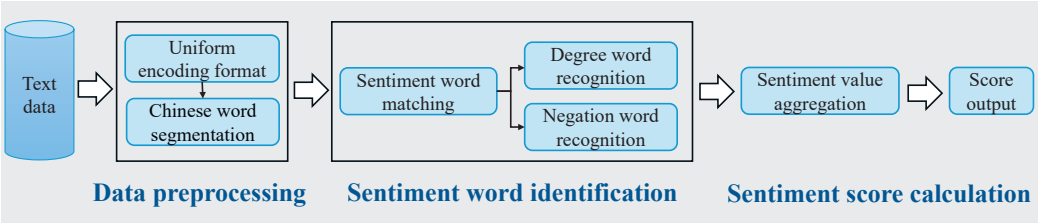


Figure 5. A dictionary-based method for financial sentiment analysis in Chinese.

3.5.1. Construction of Financial Sentiment Dictionary

We integrated two Chinese financial sentiment dictionaries developed by Jiang Fuwei and Yao Jiaquan to construct an enhanced lexicon [42–44]. Their original dictionaries were built using data mining and deep learning techniques to analyze both formal sources (e.g., annual reports) and informal sources (e.g., social media data). The formal sentiment dictionary categorized annual reports as positive or negative based on cumulative stock returns within three trading days of release. Investor posts from A-share companies on Xueqiu and Eastmoney were analyzed to extract market sentiment from informal sources.

3.5.2. Sentiment Index Calculation

In this study, investor posts from social platforms were transformed into sentiment scores using the constructed dictionary. The computation process begins with segmentation and lexical labeling of Chinese sentences, followed by sentiment word extraction. Sentiment words such as *increase*, *bullish*, and *optimistic* are assigned a score of +1, whereas negative words like *decline*, *bearish*, and *pessimistic* are assigned a score of –1. If a sentiment word is preceded by a degree adverb or a negation, its score is multiplied by the corresponding weight. Degree adverbs are weighted as follows: *extreme* = 4, *very* = 3, *more* = 2, and *ish* = 1. A negation word, such as *not* or *never*, is assigned a weight of -1. This process continues sequentially until the entire text is processed. Finally, the scores for all words in a sentence are summed to generate the final sentiment score.

Example 1 (Positive Comment). *"The market is very bullish today and expected to rise further".*

- *The word bullish is a positive sentiment word* $(+1)$*, preceded by the degree adverb very:* $(weight = 3) \rightarrow score = 3 \times 1 = +3$.
- *The word rise is a positive sentiment word* $(+1)$*, with no modifier* $\rightarrow score = +1$.
- *Final sentiment score:* $+3 + 1 = +4$.

Example 2 (Negative Comment). *"The outlook is not optimistic and prices may slightly decline".*

- *The word optimistic is a positive sentiment word* $(+1)$*, but is preceded by the negation not:* $(weight = -1) \rightarrow score = -1 \times 1 = -1$.
- *The word decline is a negative sentiment word* (-1)*, preceded by the degree adverb slightly:* $(weight = 1) \rightarrow score = 1 \times (-1) = -1$.
- *Final sentiment score:* $-1 + (-1) = -2$.

The daily sentiment index, serving as a proxy for investor sentiment, is computed as the average sentiment score across all posts on a given day, where i represents the i-th post, and n denotes the total number of posts.

$$Sentiment_{investor} = \frac{1}{n} \sum_{i=1}^{n} score_i \tag{26}$$

The same method was applied to compute sentiment scores for financial news. Long articles were split into sentences, which were analyzed sequentially and aggregated into an

overall sentiment score. Due to their length, news articles may exhibit inflated sentiment scores. To mitigate this effect, each article was classified as positive, negative, or neutral based on its aggregated score. The calculation formula is as follows:

$$Sentiment_{news} = \frac{N_{pos} - N_{neg}}{N_{pos} + N_{neg}} \quad (27)$$

where N_{pos} and N_{neg} denote the number of positive and negative news articles for the day, respectively.

3.6. Forecasting Framework

To capture complex market behaviors, we designed LASSO-NSAutoformer, a hybrid model integrating feature selection, sentiment analysis, and deep learning. It combines filtered structured features and sentiment signals from unstructured text to form a unified forecasting input. This integration enables a comprehensive representation of market dynamics. As illustrated in Figure 6, the proposed method consisted of three main stages: data collection, data processing, and empirical forecasting.

Step 1: Data collection. Factors were collected from various domains and categorized into structured and unstructured sources. Structured data included historical price series, trading data, technical indicators, composite indices, and other financial market variables. Unstructured data consisted of investor comments and financial news.

Step 2: Data processing. Structured variables were preprocessed, and LASSO was applied to select key features, reducing redundancy. For unstructured text, sentiment scores were computed using a domain-specific financial sentiment dictionary, yielding investor and news sentiment indices. These indices were combined with LASSO-selected features to form the final model input.

Step 3: Empirical forecasting. The combined input was fed into NSAutoformer. Several comparative experiments were conducted across different feature sets, selection methods, and forecasting models to evaluate the robustness and accuracy of the proposed approach. Additionally, sensitivity tests on time window lengths and forecast horizons were performed to assess stability.

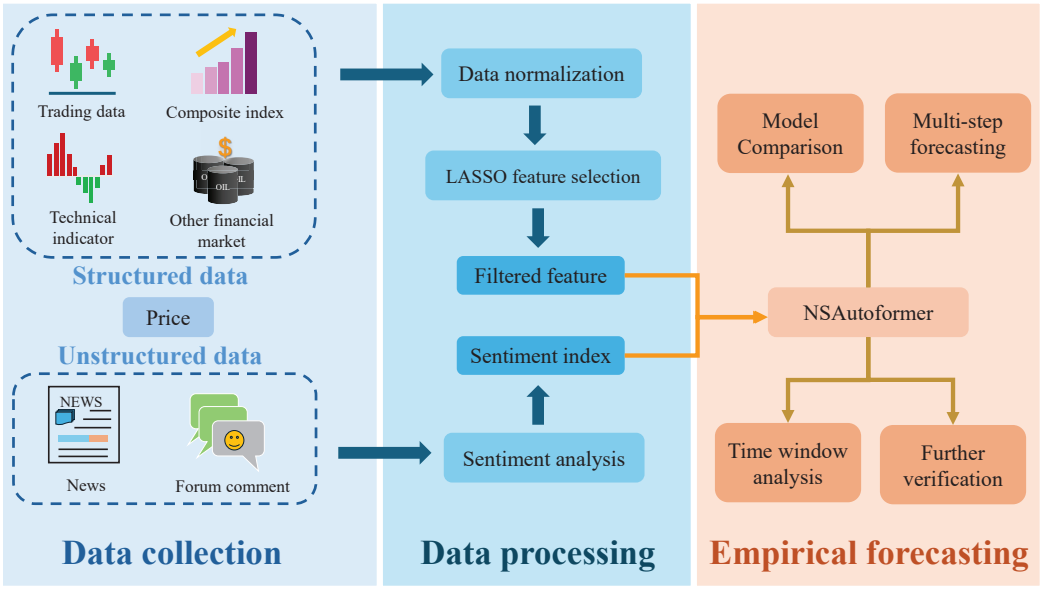


Figure 6. Hybrid forecasting framework.

3.7. Evaluation Metrics

To evaluate the model's forecasting accuracy, we used four standard regression metrics: MAE, RMSE, MAPE, and R^2:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{28}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{29}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|\hat{y}_i - y_i|}{y_i} \times 100\% \tag{30}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{31}$$

where $\hat{y}_i$ is the predicted value, y_i is the true value, $\bar{y}$ is the average of y_i, and n is the sample size. MAE measures the average absolute error, while RMSE reflects error magnitude in the same units as the target. MAPE expresses the average percentage error, and R^2 quantifies the proportion of variance explained by the model.

4. Experiments

4.1. Data Preparation

In experiments, we found that individual stocks are more volatile due to firm-specific or industry factors, while market indices are better suited for research [45]. To ensure representativeness, we selected the Shanghai Securities Exchange Composite Index (SSECI), which includes all A-share stocks on the Shanghai Stock Exchange (SSE) and covers a wide range of industries, making it the most comprehensive stock index in China's financial market. Table 1 shows a sample of SSECI price and trading data, obtained using the efinance package in Python 3.13.2 [46]. The dataset spans from 2 January 2019 to 8 February 2024. Figure 7 illustrates an exploratory analysis of SSECI price trends.

Table 1. Data samples of the SSECI.

Date	Open Price	Highest Price	Lowest Price	Close Price	Volume	Amount	Change	Change Rate	Turnover Rate
2019/1/2	2497.88	2500.28	2456.42	2465.29	109,932,014	97,592,573,952	−28.61	−1.15	0.24
2019/1/3	2461.78	2488.48	2455.93	2464.36	124,397,496	106,922,790,912	−0.93	−0.04	0.27
2019/1/4	2446.02	2515.32	2440.91	2514.87	168,877,668	139,298,676,736	50.51	2.05	0.37
2019/1/7	2528.7	2536.98	2515.51	2533.09	177,305,011	145,513,242,624	18.22	0.72	0.39
2019/1/8	2530.3	2531.34	2520.16	2526.46	158,099,182	123,379,040,256	−6.63	−0.26	0.35

We used TA-Lib (Technical Analysis Library) to calculate 28 commonly used technical indicators based on price and trading data. Details are listed in Table 2. Trend indicators identify market direction—uptrend, downtrend, or consolidation [47]. Momentum indicators measure price movement speed, detecting overbought or oversold conditions. Volume indicators assess trend strength and market interest through trading volume changes.

In addition, we selected 12 variables categorized into two groups: (1) Composite indices: Daily Geopolitical Risk Index (GPRD), Daily Infectious Disease Equity Market Volatility Index (IDEMVI), search volume on Google Trends (GT), and Baidu Index (BI). (2) Other financial markets: COMEX gold futures closing price (GC), West Texas Intermediate (WTI), Dow Jones Industrial Average (DJIA), NASDAQ Composite Index (NASDAQ), S&P 500 Index (S&P), USD to CNY Exchange Rate (USDCNY), U.S. Dollar Index (DXY), and HANG SENG INDEX (HSI). Table 3 summarizes the details of these two categories.

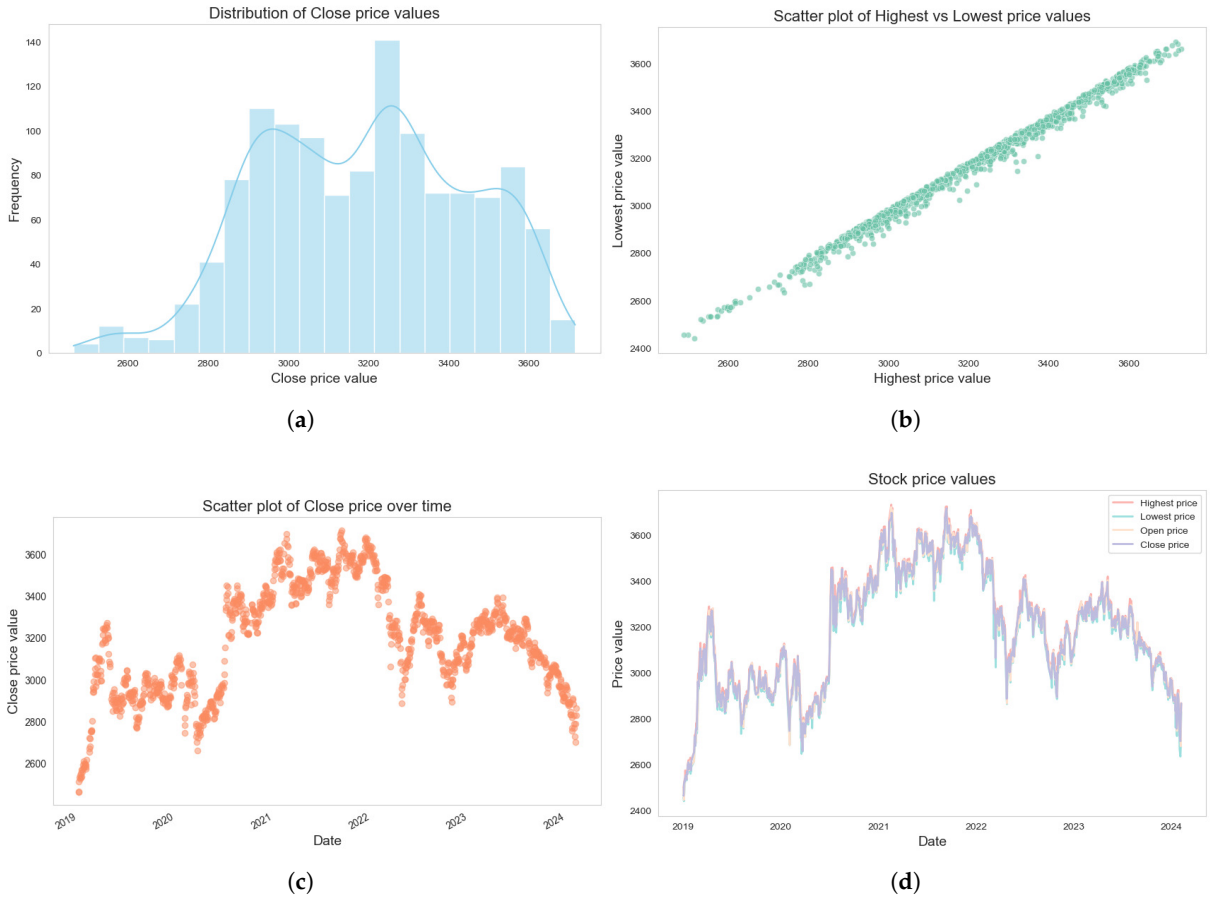


Figure 7. Data analysis of the SSECI. (**a**) Distribution of close price. (**b**) Scatter plot of highest vs. lowest price. (**c**) Scatter plot of close price over time. (**d**) Stock price values.

Table 2. Calculated technical indicators.

Types	Technical Indicators	Abbreviation
Trend indicators	Moving Average (5)	MA (5)
	Moving Average (10)	MA (10)
	Moving Average (20)	MA (20)
	Moving Average (30)	MA (30)
	Moving Average (60)	MA (60)
	Exponential Moving Average (5)	EMA (5)
	Exponential Moving Average (10)	EMA (10)
	Exponential Moving Average (20)	EMA (20)
	Exponential Moving Average (30)	EMA (30)
	Exponential Moving Average (60)	EMA (60)
	Moving Average Convergence and Divergence (6, 13, 5)	MACD (6, 13, 5), MACDsignal (6, 13, 5), MACDhist (6, 13, 5)
	Moving Average Convergence and Divergence (12, 26, 9)	MACD (12, 26, 9), MACDsignal (12, 26, 9), MACDhist (12, 26, 9)
	Moving Average Convergence and Divergence (30, 60, 30)	MACD (30, 60, 30), MACDsignal (30, 60, 30), MACDhist (30, 60, 30)
Momentum indicators	Relative Strength Index (14)	RSI (14)
	Williams %R (14)	WILLR (14)
	Momentum Index (14)	MOM (14)
	Chande Momentum Oscillator (14)	CMO (14)
	Ultimate Oscillator (7, 14, 28)	ULTOSC (14)
	Commodity Channel Index (14)	CCI (14)
	Rate of Change (10)	ROC (10)
Volume indicators	On Balance Volume	OBV
	Chaikin A/D Oscillator (3,10)	ADOSC (3,10)

In the composite index, we considered the impact of geopolitics and COVID-19 on the stock market. The GPRD [48] reflects a measure of geopolitical events and associated risks, based on newspaper statistics. Prior studies have shown that wars and crises typically reduce investment, depress stock prices, and lower employment, indicating a strong link

between geopolitical risks and economic performance. The IDEMVI [49] captures the relationship between infectious disease outbreaks and equity market volatility, particularly during the COVID-19 period. GT [50] and BI [51] represent search activity on Google and Baidu, the largest search engines globally and in China, respectively. These indices reflect keyword search volumes and serve as proxies for market attention and interest preferences.

Table 3. Selected variables within two categories.

Category	Variables
Composite index	GPRD, IDEMVI, GT, BI
Other financial markets	GC, WTI, DJIA, NASDAQ, S&P, USDCNY, DXY, HSI

In recent years, globalization and financial integration have strengthened linkages among regional markets. Correlations and risk spillovers across markets have attracted significant scholarly attention, particularly during crises when such linkages intensify. Asset price movements often propagate across markets, forming a complex network of interdependencies. All variables related to other financial markets were obtained using the efinance package.

In summary, 49 candidate variables were collected for structured data. Regarding unstructured text data, investor comments were collected from the Eastmoney Guba forum [52] as shown in Table 4. Financial news was sourced from China News [53] (A-share section) and macro/news finance columns on China.com.cn [54]. Additional policy-related content, including current affairs, key events, press conferences, and policy interpretations, was collected from the official website of the China Securities Regulatory Commission (CSRC) [55]. All text data were obtained using the Bazhuayu web crawler [56], covering the period from 2 January 2019 to 8 February 2024 [57].

Table 4. Sample text data from investor forum.

Reading Volume	Content	Author	Date
675	Bottom tomorrow, last drop, speculators sad.	Shareholder 65853W8Y76	2020-01-21 18:28
526	Digging a hole today for tomorrow's upward leap.	Tao Junjiang	2020-01-21 18:30
878	Just wondering, why did global stocks plummet today? Any news?	YingerJJ	2020-01-21 18:33

4.2. Data Preprocessing

The dataset was divided into training (70%), validation (10%), and test sets (20%). Prior to experimentation, data normalization was applied to ensure numerical stability and feature comparability. Each feature was standardized by computing its mean and variance independently. The formula is as follows:

$$X_{score} = \frac{X_{org} - X_{mean}}{X_{std}} \tag{32}$$

where X_{org} denotes the original data, X_{mean} denotes the mean of the data, X_{std} denotes the standard deviation of the input data, and X_{score} denotes the standardized result.

Excessive features can hinder model efficiency and forecasting accuracy, so LASSO was applied for feature selection by compressing some coefficients to zero. The LASSO model was optimized using grid search and 10-fold cross-validation, resulting in features with thresholds greater than zero, as shown in Table 5. Spearman correlations among the selected features are shown in Figure 8.

Table 5. Results of the LASSO feature selection.

Features	Coefficients
Close price	0.913147
Highest price	0.036236
NASDAQ	0.032230
DXY	0.005875
Amount	0.001701
CCI(14)	0.001464

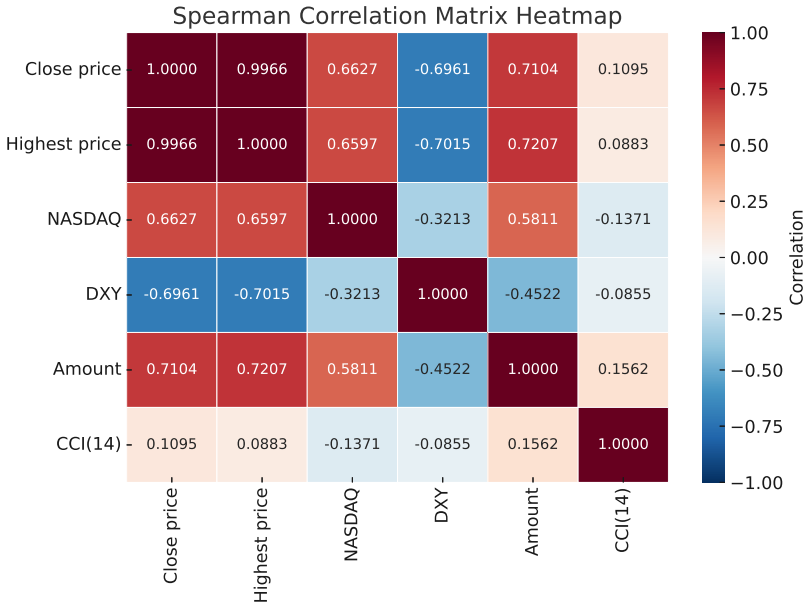


Figure 8. Spearman correlation matrix heatmap.

For text data, we applied the constructed Chinese financial sentiment dictionary after preprocessing to extract investor and news sentiment indices. To minimize the impact of extreme values, obvious outliers were removed. In total, 4,529,650 valid comments were retained. Figure 9 illustrates the complete data processing pipeline, from raw data acquisition to the construction of model input features.

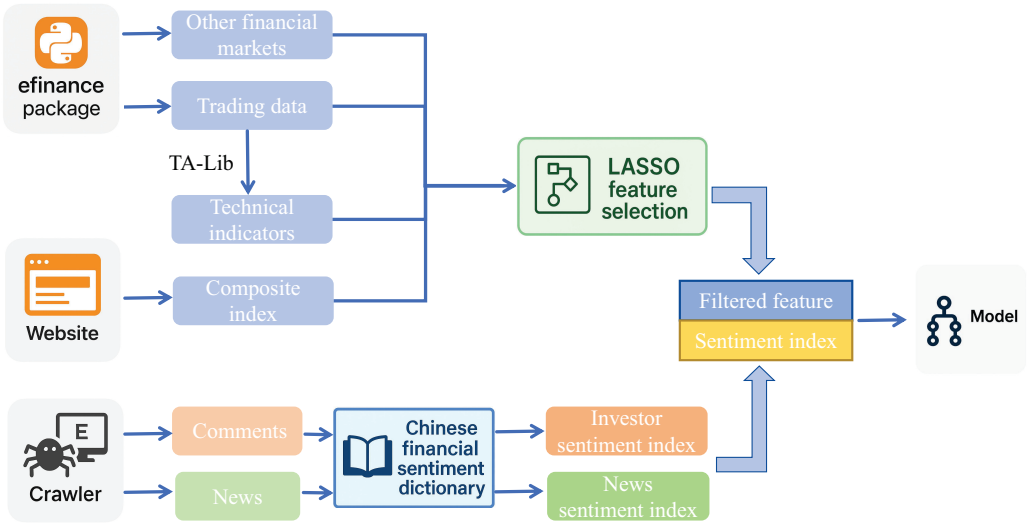


Figure 9. The full pipeline of the proposed forecasting framework.

4.3. Hyperparameters Setting

Hyperparameter tuning was performed using Optuna, which applies Bayesian optimization with a Tree-structured Parzen Estimator (TPE) to efficiently search the parameter space. The framework minimizes validation loss across multiple trials and selects the best configuration accordingly. Table 6 defines the search space and Table 7 reports the optimal hyperparameters for each model. Mean absolute error (MAE) was used as the loss function. Training was conducted for up to 50 epochs with early stopping to prevent overfitting and ensure convergence.

Table 6. Hyperparameter search space.

	Model	
Hyperparameters	**Transformer-Based Model**	**RNN-Based Model**
Learning rate	Loguniform $(1 \times 10^{-5}, 1 \times 10^{-1})$	Loguniform $(1 \times 10^{-5}, 1 \times 10^{-1})$
Dropout rate	Float (0.0, 0.5, step = 0.01)	Float (0.0, 0.5, step = 0.01)
Batch size	{8, 16, 32, 64, 128, 256}	{8, 16, 32, 64, 128, 256}
Number of RNN layers	-	{1, 2, 3, 4, 5}
Number of hidden-layer neurons	-	{8, 16, 32, 64, 128, 256}
Number of encoder layers	{1, 2, 3, 4}	-
Number of decoder layers	{1, 2, 3, 4}	-
Number of attention heads	{4, 8, 16}	-
Dimension of model	{256, 512, 1024}	-
Number of ProbSparse self-attention factors	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10}	-

Table 7. Optimal parameters for each model.

Model	**Parameter Settings**	**Value**
NSAutoformer	Learning rate	0.002
	Dropout rate	0.05
	Batch size	16
	Number of encoder layers	1
	Number of decoder layers	2
	Number of attention heads	16
	Dimension of model	256
Non-stationary Transformer	Learning rate	0.0001
	Dropout rate	0.07
	Batch size	32
	Number of encoder layers	1
	Number of decoder layers	4
	Number of attention heads	8
	Dimension of model	512
Autoformer	Learning rate	0.0022
	Dropout rate	0.06
	Batch size	8
	Number of encoder layers	2
	Number of decoder layers	1
	Number of attention heads	8
	Dimension of model	256
Informer	Learning rate	0.0002
	Dropout rate	0.07
	Batch size	8
	Number of encoder layers	2
	Number of decoder layers	2
	Number of attention heads	8
	Dimension of model	256
	Number of ProbSparse self-attention factors	9

Table 7. *Cont.*

Model	Parameter Settings	Value
Transformer	Learning rate	0.0006
	Dropout rate	0.1
	Batch size	16
	Number of encoder layers	4
	Number of decoder layers	2
	Number of attention heads	16
	Dimension of model	512
BiLSTM	Learning rate	0.0162
	Dropout rate	0.08
	Batch size	8
	Number of RNN layers	1
	Number of hidden-layer neurons	128
LSTM	Learning rate	0.003
	Dropout rate	0.05
	Batch size	8
	Number of RNN layers	2
	Number of hidden-layer neurons	128
BiGRU	Learning rate	0.0006
	Dropout rate	0.01
	Batch size	8
	Number of RNN layers	2
	Number of hidden-layer neurons	256
GRU	Learning rate	0.0068
	Dropout rate	0.11
	Batch size	16
	Number of RNN layers	1
	Number of hidden-layer neurons	256
RNN	Learning rate	0.0013
	Dropout rate	0.11
	Batch size	8
	Number of RNN layers	4
	Number of hidden-layer neurons	256

5. Results and Analysis

We adopted the moving window approach, a common method in time series forecasting. This approach uses a sliding window over the data, allowing the model to generate predictions at each time step. In our experiment, features from the trading day T were used to predict the price on the following day, $T + 1$.

5.1. Feature Set Evaluation

Given the inclusion of multiple data sources, we conducted an experiment to evaluate the predictive performance of different feature sets. Table 8 presents the results of five candidate variable sets using the proposed NSAutoformer. The results show that adding technical indicators, composite indices, and financial market data to the base set reduces forecasting errors and improves accuracy. This indicates that incorporating a wider range of relevant features can enhance price forecasting capability. Notably, the lowest MAE was observed in Set_2, which includes price, trading, and technical indicators, confirming the positive impact of technical factors. However, when all variables are included (Set_5), performance declines, indicating that excessive features may introduce noise and increase overfitting risk. To mitigate this, we apply the LASSO algorithm for feature selection.

Table 8. The prediction performance of different feature sets.

Feature Set	Factor	MAE	RMSE	MAPE	R^2
Set_1	Price and trading	20.921978	27.791372	0.006735	0.967110
Set_2	Price and trading, technical indicators	20.718376	27.653543	0.006669	0.967436
Set_3	Price and trading, composite indices	20.755861	26.990023	0.006667	0.968980
Set_4	Price and trading, other financial markets	20.770327	27.186878	0.006663	0.968526
Set_5	All factors	21.098995	27.456692	0.006787	0.967898

Subsequently, we constructed new feature sets by combining the LASSO-filtered variables with sentiment indices. As shown in Table 9, Set_6, which contains only the filtered features, outperforms all raw feature sets (Sets 1–5), indicating that LASSO improves model accuracy by removing irrelevant or noisy variables. Compared to the best raw set (Set_2), Set_6 reduces MAE from 20.718376 to 20.350271 and increases R^2 from 0.967436 to 0.970052.

Furthermore, incorporating sentiment indices (Sets 7–9) improved model performance. The best result occurred when both investor and news sentiment were included, yielding the lowest MAE (19.253584) and highest R^2 (0.972357). These findings confirm the complementary benefits of feature selection and sentiment integration. LASSO reduces redundancy, while sentiment indices capture psychological factors often missing from traditional financial features.

The consistent improvement suggests that psychological factors significantly influence market dynamics. Integrating sentiment from multiple sources improves index prediction, with broader group sentiment yielding better results. Investor sentiment reflects short-term fluctuations driven by individual behavior, while news sentiment captures broader market tone and policy direction. Together, these two sentiment sources provide complementary information. These findings further support integrating sentiment analysis into the forecasting framework, allowing the model to adapt to dynamic market conditions.

Table 9. The prediction performance of combined filtered features and sentiment factors.

Feature Set	Factor	MAE	RMSE	MAPE	R^2
Set_6	Filtered features	20.350271	26.519423	0.006544	0.970052
Set_7	Filtered features, investor sentiment	19.830795	25.967213	0.006378	0.971286
Set_8	Filtered features, news sentiment	19.568808	25.896173	0.006279	0.971443
Set_9	Filtered features, investor and news sentiment	19.253584	25.478617	0.006189	0.972357

We also compared several feature selection algorithms, including XGBoost, Random Forest, CatBoost, and AdaBoost, which are widely used in machine learning. To ensure fairness, model structures and hyperparameter tuning followed settings from relevant studies. As shown in Table 10, all algorithms improved forecasting accuracy, with LASSO achieving the lowest error and best overall fit.

Among the feature selection methods compared, LASSO-filtered features achieved the best forecasting accuracy. Unlike ensemble-based methods like Random Forest or XGBoost, which tend to produce dense feature sets requiring extra tuning, LASSO yields sparse and interpretable models with fewer variables. This aligns with the goals of economic and financial forecasting, where transparency and stability are essential. Moreover, its simplicity, computational efficiency, and robustness to multicollinearity make LASSO well suited for high-dimensional financial time series.

Table 10. Comparative experiment applying different feature selection algorithms.

Factor	MAE	RMSE	MAPE	R^2
All factors	21.098995	27.456692	0.006787	0.967898
CatBoost filtered features	20.717237	27.376080	0.006669	0.968086
AdaBoost filtered features	20.685173	27.260538	0.006663	0.968355
XGBoost filtered features	20.635792	27.435696	0.006656	0.967947
Random forest filtered features	20.552847	26.862595	0.006611	0.969272
LASSO filtered features	20.350271	26.519423	0.006544	0.970052

5.2. Forecasting Results

This section presents comparative experiments evaluating different forecasting models on the SSECI dataset. The evaluation focuses on one-step forecasting, with results reported using MAE, RMSE, MAPE, and R^2, as shown in Table 11. The results reveal a clear pattern: advanced model architectures generally yield better predictive performance. In particular, Transformer-based models consistently outperform RNN-based models, with NSAutoformer delivering the most accurate results overall.

The performance of the basic RNN model is the weakest, underscoring its limitations in capturing long-range dependencies and handling high-dimensional feature inputs. Although GRU (MAE: 34.960663, RMSE: 47.148529, MAPE: 0.011468, R^2: 0.905338) and LSTM (MAE: 35.166011, RMSE: 47.015923, MAPE: 0.011536, R^2: 0.905870) improve upon RNN, they remain inferior to their bidirectional versions and Transformer-based models.

The vanilla Transformer (MAE: 35.786094, RMSE: 43.768764, MAPE: 0.011411, R^2: 0.918423), while more effective than RNN-based models, was surpassed by its enhanced variants. Informer (MAE: 33.746243, RMSE: 43.102985, MAPE: 0.010937, R^2: 0.920886) improved accuracy through the ProbSparse self-attention mechanism, which reduces computational cost by focusing on dominant queries. Autoformer further improved forecasting by decomposing time series into trend and seasonal components.

Among Transformer-based models, the proposed NSAutoformer is built on the Autoformer architecture. It incorporates non-stationary modeling to improve adaptability to dynamic data distributions. It adapts the autocorrelation mechanism to handle distributional shifts, improving ability to represent complex, evolving temporal patterns. Without relying on feature selection or sentiment inputs, NSAutoformer already achieved the best results across all metrics (MAE: 21.098995, RMSE: 27.456692, MAPE: 0.006787, R^2: 0.967898). These results demonstrate NSAutoformer's architectural advantage in capturing both structural and statistical complexity, highlighting the importance of models tailored to non-stationary time series forecasting.

Table 11. Forecasting error of single models.

Model	MAE	RMSE	MAPE	R^2
RNN	56.669086	64.467438	0.018306	0.823022
GRU	34.960663	47.148529	0.011468	0.905338
BiGRU	29.442695	37.329960	0.009607	0.940659
LSTM	35.166011	47.015923	0.011536	0.905870
BiLSTM	33.801483	41.940468	0.010884	0.925096
Transformer	35.786094	43.768764	0.011411	0.918423
Informer	33.746243	43.102985	0.010937	0.920886
Autoformer	22.093636	28.635998	0.007138	0.965081
Non-Stationary Transformer	21.619484	28.404745	0.006940	0.965643
NSAutoformer	21.098995	27.456692	0.006787	0.967898

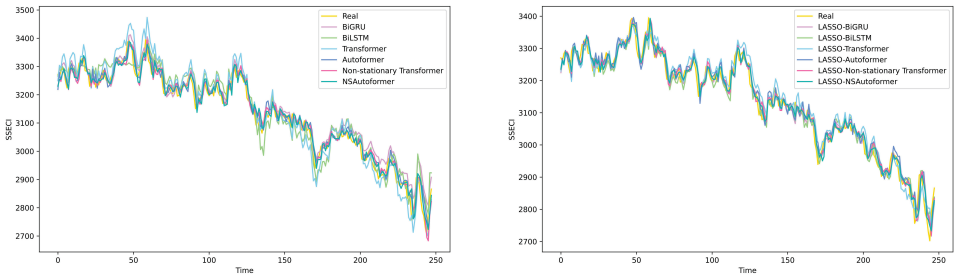
Building on the single-model results, we evaluated a hybrid forecasting framework that integrates LASSO-based feature selection and sentiment analysis. As shown in Table 12

and Figure 10, incorporating filtered features and sentiment indices significantly reduced MAE, RMSE, and MAPE values, while consistently improving R^2. These results highlight the benefit of combining data-driven feature selection with sentiment inputs, which together enrich the input space.

Notably, the hybrid model reduced MAE by 58.28%, 38.82%, 28.31%, 35.42%, 35.71%, 24.65%, 21.44%, 2.57%, 6.75%, and 8.75% relative to individual baseline models. These results clearly demonstrate the advantage of the proposed framework over individual models. Figure 11 further illustrates forecasting performance across all metrics, highlighting the superiority of the hybrid approach.

However, within the hybrid models, Transformer and Informer underperformed compared to RNN. This may be due to their architectural complexity: although effective in high-dimensional settings, they require large datasets and rich feature representations to perform optimally. In contrast, LASSO reduces dimensionality, potentially limiting these models' capacity to capture nonlinear interactions. As a result, they become less effective than RNN-based models when inputs are simplified. This finding aligns with machine learning theory, which suggests simpler models often generalize better with limited input spaces.

Among all hybrid configurations, LASSO-NSAutoformer achieved the best performance (MAE: 19.253584, RMSE: 25.478617, MAPE: 0.006189, R^2: 0.972357). These results provide strong empirical support for the proposed framework, highlighting both its architectural advantage and its effectiveness in integrating heterogeneous data sources to enhance predictive accuracy.

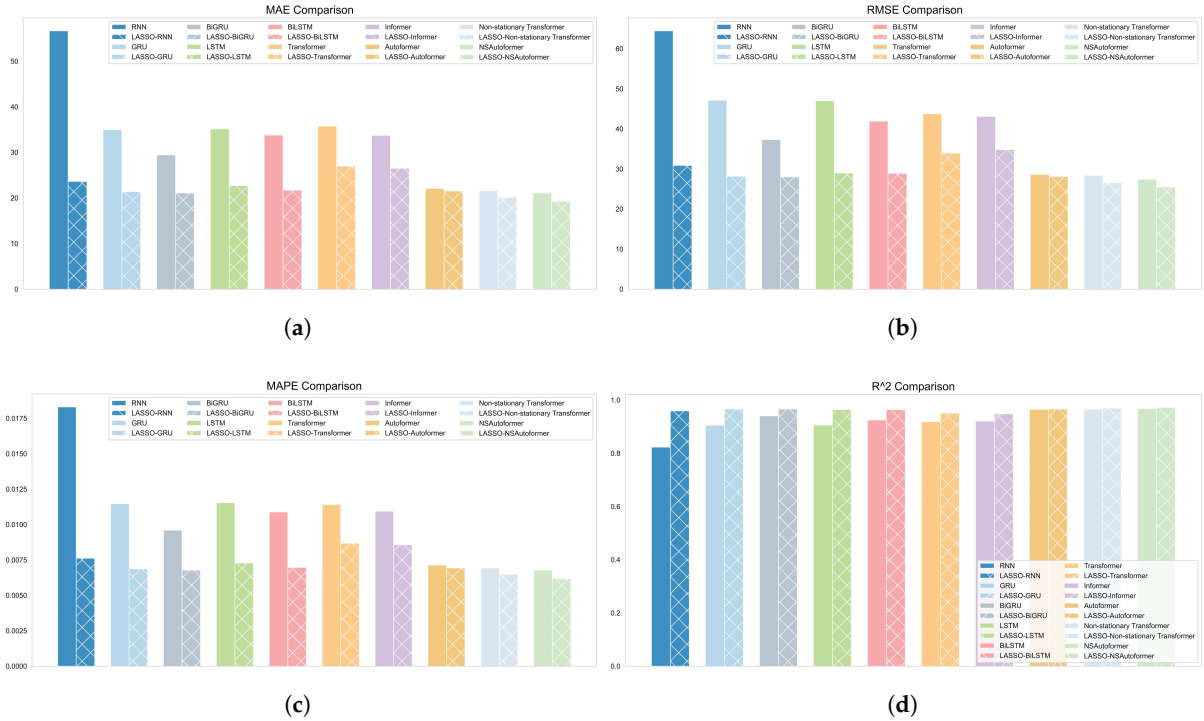


(**a**) (**b**)

Figure 10. Comparison of fitting curves of different forecasting models. (**a**) Single model. (**b**) Hybrid model.

Table 12. Forecasting error of hybrid forecasting models.

Model	MAE	RMSE	MAPE	R^2
LASSO-RNN	23.644413	30.889042	0.007625	0.959370
LASSO-GRU	21.386953	28.184166	0.006874	0.966174
LASSO-BiGRU	21.106899	28.013762	0.006787	0.966582
LASSO-LSTM	22.713284	29.025877	0.007290	0.964124
LASSO-BiLSTM	21.726543	28.899588	0.006979	0.964435
LASSO-Transformer	26.960608	33.957767	0.008680	0.950896
LASSO-Informer	26.516241	34.835022	0.008567	0.948326
LASSO-Autoformer	21.525358	28.114588	0.006934	0.966341
LASSO-Non-stationary Transformer	20.158091	26.561501	0.006490	0.969957
LASSO-NSAutoformer	19.253584	25.478617	0.006189	0.972357



(**a**) (**b**)

(**c**) (**d**)

Figure 11. The forecasting model's predictive performance is evaluated across four metrics. (**a**) MAE. (**b**) RMSE. (**c**) MAPE. (**d**) R^2.

5.3. Multi-Step Forecasting Evaluation

To assess the model's temporal scalability, we conducted multi-step forecasting using a moving window across horizons of 3, 5, 7, 10, and 12 steps. This setup enables comprehensive evaluation across short-, medium-, and long-term forecasting tasks. Table 13 presents the error metrics for each forecast horizon, followed by an analysis of performance variation across temporal scales. Figure 12 visually compares the prediction results of different models across various forecasting horizons.

Table 13. Evaluated performance of multi-step forecasting.

Model	Step	MAE	RMSE	MAPE	R^2
LASSO-RNN	3	43.188992	57.145924	0.014126	0.859717
	5	51.967407	69.742378	0.017034	0.787520
	7	65.749321	87.254677	0.021641	0.660065
	10	77.670517	102.719220	0.025394	0.515159
	12	90.754562	119.042760	0.029656	0.338843
LASSO-GRU	3	35.750938	46.092323	0.011524	0.908738
	5	43.450806	56.193111	0.013997	0.862060
	7	51.929230	65.786865	0.016843	0.806760
	10	68.435005	86.322083	0.022039	0.657595
	12	73.448906	93.432213	0.023853	0.592721
LASSO-BiGRU	3	35.078178	45.372971	0.011307	0.911564
	5	44.040451	58.065067	0.014317	0.852717
	7	54.005966	71.492027	0.017536	0.771791
	10	65.161751	87.689674	0.021190	0.646660
	12	74.452072	101.253490	0.024197	0.521680

Table 13. *Cont.*

Model	Step	MAE	RMSE	MAPE	R^2
LASSO-LSTM	3	36.054440	47.499245	0.011665	0.903082
	5	47.946892	62.373707	0.015528	0.830048
	7	58.613914	75.970055	0.019007	0.742307
	10	71.950310	92.409950	0.023301	0.607596
	12	77.852409	98.014580	0.025040	0.551791
LASSO-BiLSTM	3	35.619205	47.316196	0.011527	0.903827
	5	47.037033	61.520142	0.015271	0.834667
	7	54.839153	69.498940	0.017577	0.784338
	10	68.615288	89.684113	0.022284	0.630404
	12	74.977692	93.511497	0.024149	0.592030
LASSO-Transformer	3	39.123184	49.077202	0.012547	0.896535
	5	53.265762	66.649567	0.017254	0.805948
	7	62.481350	76.897873	0.020189	0.735974
	10	69.944923	85.276726	0.022350	0.665838
	12	80.602280	103.711590	0.026383	0.498174
LASSO-Informer	3	46.155979	61.358738	0.015006	0.838272
	5	51.952137	67.886536	0.016868	0.798678
	7	65.082924	83.330078	0.021190	0.689957
	10	87.948105	116.928140	0.028829	0.371748
	12	105.200800	136.698410	0.034424	0.128182
LASSO-Autoformer	3	30.168144	39.117413	0.009734	0.934269
	5	46.565857	59.484188	0.015049	0.845429
	7	52.720787	67.037750	0.017032	0.799342
	10	68.571671	84.232521	0.022186	0.673972
	12	71.131798	87.741501	0.023080	0.640823
LASSO-Non-stationary Transformer	3	29.778294	39.400871	0.009622	0.933312
	5	38.530983	51.212330	0.012496	0.885429
	7	42.040813	56.086330	0.013668	0.859547
	10	49.331158	64.164070	0.016033	0.810818
	12	59.020134	73.250259	0.019220	0.749668
LASSO-NSAutoformer	3	28.095316	37.111507	0.009050	0.940837
	5	37.026203	48.447479	0.011978	0.897466
	7	40.293056	52.063671	0.013054	0.878972
	10	48.852676	62.379807	0.015847	0.821193
	12	56.029739	70.127258	0.018164	0.770558

The multi-step forecasting results provide strong evidence for the effectiveness of LASSO-NSAutoformer, particularly in terms of accuracy and stability across different horizons. Specifically, it achieved the lowest MAE and MAPE in short-term (three-step: MAE 28.095316, MAPE 0.009050) and medium-term forecasts (five-step: MAE 37.026203, MAPE 0.011978), significantly outperforming other models. These results confirm the model's strong ability to capture market dynamics, which is essential for financial forecasting.

As the forecasting horizon increased to 7, 10, and 12 steps, all models showed reduced accuracy. However, LASSO-NSAutoformer consistently maintained the lowest errors and showed slower performance degradation than other models. This stability across longer horizons highlights the model's adaptability to evolving market dynamics and supports its application in long-term forecasting. In practice, reliable multi-step forecasts help investors and portfolio managers anticipate future trends across various time horizons, enabling more informed decisions and effective risk management.

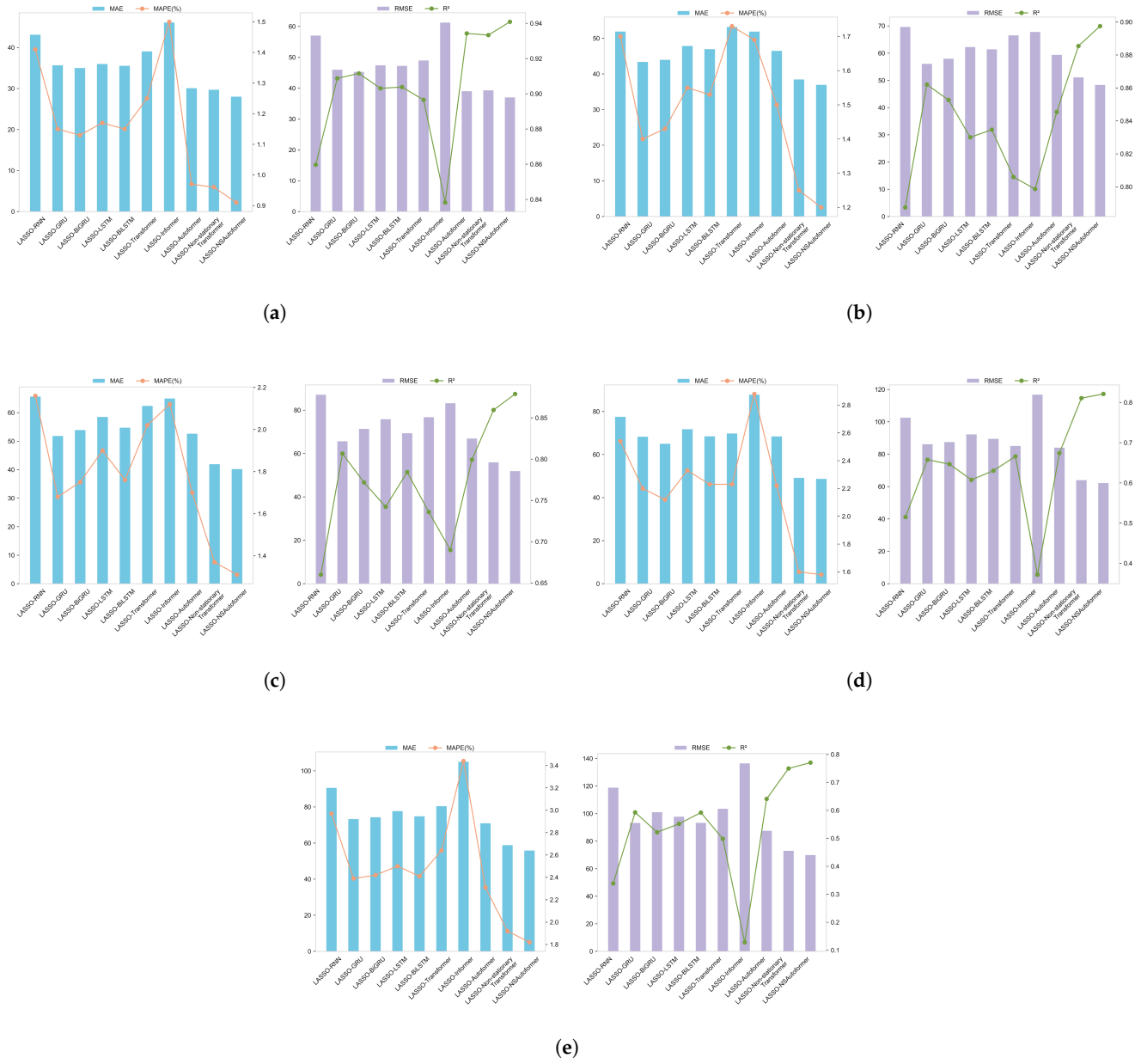


Figure 12. Multi-step forecasting performance of different models. (**a**) Three-step. (**b**) Five-step. (**c**) Seven-step. (**d**) Ten-step. (**e**) Twelve-step.

5.4. Time Window Analysis

The moving window approach is widely used in stock price forecasting, where window length is a critical parameter. It structures historical data into a format suitable for neural networks, directly influencing model performance. Generally, shorter windows capture short-term volatility but may miss long-term trends, while longer windows reduce noise but are less responsive to rapid changes. Thus, balancing short-term and long-term effects is key. Prior studies commonly use window lengths ranging from 5 to 30 days.

Table 14 shows model performance across window lengths from 5 to 30 days, evaluated using MAE, RMSE, MAPE, and R^2. The results indicate that longer windows generally lead to higher error values and reduced model performance. Since the experiments focus on single-step forecasting, a window length of 5 to 15 days is optimal, with most models performing best at 10 days.

As shown in Figure 13, LASSO-NSAutoformer consistently outperformed all other models across window lengths, with minimal performance degradation as the window increased. Its stable accuracy over longer historical spans suggests an effective balance between short-term fluctuations and long-term dependencies. This adaptability enhances the model's practicality, enabling it to accommodate varying historical lengths common in real-world financial forecasting. Overall, the results confirm the framework's strength in predictive stability and generalizability.

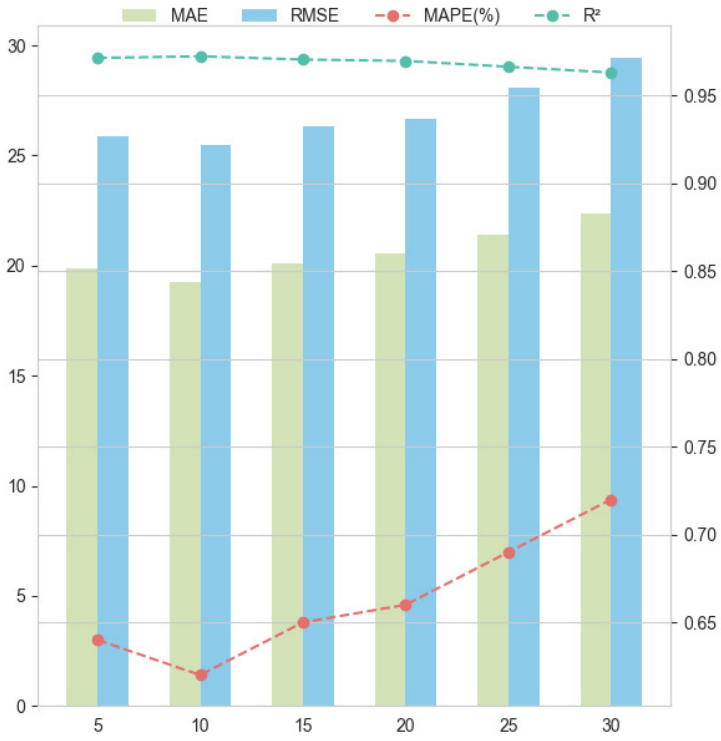


Figure 13. The LASSO-NSAutoformer forecast results for different time window lengths.

Table 14. Time window length experiment.

Model	Step	MAE	RMSE	MAPE	R^2
LASSO-RNN	5	24.483894	32.681717	0.007959	0.954517
	10	23.644413	30.889042	0.007625	0.959370
	15	25.247023	32.941116	0.008150	0.953792
	20	26.473789	36.125835	0.008677	0.944426
	25	28.962196	38.566822	0.009449	0.936662
	30	27.240345	36.317226	0.008895	0.943835
LASSO-GRU	5	21.974298	28.604694	0.007082	0.965157
	10	21.386953	28.184166	0.006874	0.966174
	15	21.539961	28.225548	0.006916	0.966075
	20	22.072958	28.998404	0.007128	0.964192
	25	22.881193	29.885086	0.007390	0.961968
	30	23.962088	31.257002	0.007762	0.958396
LASSO-BiGRU	5	22.704512	29.710104	0.007326	0.962412
	10	21.106899	28.013762	0.006787	0.966582
	15	22.242140	29.276299	0.007182	0.963502
	20	22.880867	30.291847	0.007405	0.960926
	25	22.999109	30.269529	0.007441	0.960983
	30	24.162012	31.737589	0.007833	0.957107

Table 14. *Cont.*

Model	Step	MAE	RMSE	MAPE	R^2
LASSO-LSTM	5	22.732540	29.397474	0.007335	0.963199
	10	22.713284	29.025877	0.007290	0.964124
	15	22.646397	30.325708	0.007335	0.960838
	20	23.424490	30.868370	0.007579	0.959424
	25	23.966026	31.944355	0.007791	0.956546
	30	24.932585	31.959023	0.008033	0.956506
LASSO-BiLSTM	5	21.645945	28.010662	0.006963	0.966589
	10	21.726543	28.899588	0.006979	0.964435
	15	22.603664	29.273764	0.007280	0.963508
	20	23.003563	29.650867	0.007400	0.962562
	25	24.576044	32.913986	0.007948	0.953868
	30	25.011755	32.083099	0.008101	0.956168
LASSO-Transformer	5	27.377705	34.768635	0.008787	0.948523
	10	26.960608	33.957767	0.008680	0.950896
	15	27.610346	35.769169	0.008875	0.945518
	20	29.937407	38.892006	0.009617	0.935589
	25	30.961660	39.201160	0.009906	0.934561
	30	29.898859	38.539780	0.009624	0.936751
LASSO-Informer	5	27.254883	36.831005	0.008825	0.942235
	10	26.516241	34.835022	0.008567	0.948326
	15	26.786543	36.176796	0.008645	0.944269
	20	30.294575	38.417736	0.009670	0.937151
	25	28.968504	36.717640	0.009299	0.942590
	30	29.620420	39.524673	0.009554	0.933477
LASSO-Autoformer	5	20.986221	27.377190	0.006775	0.968084
	10	21.525358	28.114588	0.006934	0.966341
	15	21.869249	28.549051	0.007055	0.965293
	20	24.905035	32.215263	0.008040	0.955806
	25	26.908518	35.175846	0.008725	0.947310
	30	32.751427	41.552368	0.010587	0.926476
LASSO-Non-stationary Transformer	5	20.728664	27.043882	0.006667	0.968856
	10	20.158091	26.561501	0.006490	0.969957
	15	20.251436	26.735344	0.006532	0.969562
	20	21.927008	28.842699	0.007086	0.964575
	25	22.153427	28.385496	0.007153	0.965689
	30	22.482288	29.163088	0.007264	0.963784
LASSO-NSAutoformer	5	19.898645	25.896940	0.006386	0.971442
	10	19.253584	25.478617	0.006189	0.972357
	15	20.100599	26.311350	0.006487	0.970520
	20	20.527988	26.662642	0.006609	0.969728
	25	21.415125	28.076717	0.006906	0.966432
	30	22.376661	29.437485	0.007245	0.963099

5.5. Further Verification

To further validate the generalizability of our proposed forecasting approach, we conducted additional experiments on three Chinese stock indices: the Shenzhen Stock Exchange Composite Index (SZSECI), the Growth Enterprise Index (GEI), and the CSI 300 Index. Following the same methodology, structured variables and investor sentiment data were collected and processed. Sentiment indices were derived from 323,764 (SZSECI), 624,311 (GEI), and 39,488 (CSI 300) valid investor comments from related forums. These sentiment indices were combined with LASSO-selected variables to form the final input for model forecasting. The corresponding results are presented in Table 15.

Table 15. Forecasting errors of models in other stock markets.

Stock Market	Model	MAE	RMSE	MAPE	R^2
SZSECI	RNN	380.468048	473.919678	0.037356	0.763531
	GRU	458.067474	581.565186	0.047094	0.643908
	BiGRU	388.156830	460.188690	0.039164	0.777035
	LSTM	457.298218	580.325745	0.047061	0.645425
	BiLSTM	425.774262	523.829529	0.043109	0.711102
	Transformer	340.938507	400.257935	0.032956	0.831327
	Informer	297.517700	381.009491	0.030306	0.847160
	Autoformer	98.346413	124.384178	0.009633	0.983711
	Non-stationary Transformer	94.279228	118.854759	0.009213	0.985127
	NSAutoformer	93.363426	121.459396	0.009114	0.984468
	LASSO-RNN	150.048843	203.535416	0.015314	0.956384
	LASSO-GRU	99.558372	130.430878	0.009733	0.982089
	LASSO-BiGRU	98.994049	131.434845	0.009675	0.981812
	LASSO-LSTM	195.845352	267.926697	0.020251	0.924422
	LASSO-BiLSTM	189.992401	267.022736	0.019659	0.924931
	LASSO-Transformer	162.001160	215.351395	0.016290	0.951173
	LASSO-Informer	149.046051	203.464493	0.014706	0.956414
	LASSO-Autoformer	94.025230	123.956970	0.009172	0.983823
	LASSO-Non-stationary Transformer	90.831985	118.011932	0.008879	0.985337
	LASSO-NSAutoformer	88.166016	114.085823	0.008620	0.986297
GEI	RNN	153.780304	181.706451	0.042134	0.545588
	GRU	126.518242	149.124405	0.064749	0.587659
	BiGRU	79.934395	95.838745	0.040635	0.829689
	LSTM	145.656662	169.963272	0.074441	0.464364
	BiLSTM	121.91526	135.809891	0.060184	0.658003
	Transformer	114.940208	134.001312	0.057583	0.667051
	Informer	87.232407	109.011948	0.043640	0.779653
	Autoformer	26.084963	32.327923	0.012623	0.980622
	Non-stationary Transformer	23.230169	29.055370	0.011296	0.984346
	NSAutoformer	22.420321	28.004850	0.010885	0.985458
	LASSO-RNN	28.380545	36.457798	0.013889	0.975354
	LASSO-GRU	25.119701	32.339462	0.012257	0.980608
	LASSO-BiGRU	24.554764	31.833340	0.011898	0.981210
	LASSO-LSTM	25.681181	33.395302	0.012814	0.979321
	LASSO-BiLSTM	25.377560	33.052277	0.012445	0.979744
	LASSO-Transformer	40.961773	57.122398	0.020993	0.939498
	LASSO-Informer	35.367172	46.888432	0.017687	0.959235
	LASSO-Autoformer	25.025919	32.302807	0.012133	0.980652
	LASSO-Non-stationary Transformer	21.596684	27.399195	0.010528	0.986080
	LASSO-NSAutoformer	20.471539	26.318472	0.009904	0.987157
CSI300	RNN	156.386368	217.138519	0.044782	0.351093
	GRU	84.334709	120.705154	0.024013	0.799479
	BiGRU	82.955772	106.818321	0.023428	0.842964
	LSTM	144.733047	174.516510	0.040390	0.580838
	BiLSTM	143.489700	180.842987	0.040348	0.549897
	Transformer	112.041649	149.263153	0.031861	0.693370
	Informer	73.754555	93.122894	0.020740	0.880650
	Autoformer	29.529997	37.642647	0.007949	0.980498
	Non-stationary Transformer	28.713575	36.630978	0.007710	0.981533
	NSAutoformer	27.881372	35.970390	0.007488	0.982193
	LASSO-RNN	50.738983	68.513290	0.014292	0.935396
	LASSO-GRU	32.533649	40.668610	0.008875	0.977237
	LASSO-BiGRU	30.832920	38.655544	0.008330	0.979435
	LASSO-LSTM	30.801199	38.781799	0.008338	0.979300
	LASSO-BiLSTM	30.342756	38.989689	0.008170	0.979078
	LASSO-Transformer	44.886173	56.785843	0.012352	0.955620
	LASSO-Informer	39.245991	50.700714	0.010773	0.964622
	LASSO-Autoformer	29.717245	37.785686	0.007958	0.980350
	LASSO-Non-stationary Transformer	27.744621	35.282570	0.007439	0.982867
	LASSO-NSAutoformer	26.974031	34.512257	0.007228	0.983607

The proposed model showed stable and accurate performance across all three structurally distinct indices, rather than being effective in only one market condition. This consistency highlights the model's ability to capture diverse market dynamics: SZSECI reflects overall market trends, GEI emphasizes volatile, growth-oriented stocks, and CSI 300 tracks stable, blue-chip firms. These indices differ in volatility, investor composition, and information sensitivity, offering a robust test for model adaptability. The model's strong performance across all indices confirms its adaptability to varied financial environments.

Once again, integrating LASSO-selected features with sentiment analysis significantly improved forecasting performance across all models and indices. For instance, in the SZSECI, the MAE decreased from 98.346413 in the single Autoformer model to 94.025230 in the LASSO-enhanced Autoformer, eventually reaching an optimal value of 88.166016 in the LASSO-NSAutoformer. Similar improvements were observed for both GEI and CSI 300.

Overall, these findings demonstrate that the proposed framework is not only effective in a single setting, but also generalizes well to multiple market environments. Its consistent performance across structurally different markets suggests potential for broader application in financial forecasting tasks.

6. Conclusions

Stock market forecasting is a critical yet challenging task, primarily due to the volatility and non-stationarity of financial time series. This study proposes LASSO-NSAutoformer, a hybrid forecasting model for Chinese stock index that integrates multi-source data, LASSO feature selection, and deep learning. The framework consists of three components: (1) collecting 49 structured financial variables, with LASSO used to remove redundancy and retain key predictors; (2) constructing a Chinese financial sentiment dictionary to quantify market sentiment from investor comments and financial news; (3) integrating structured and sentiment-based features into a deep learning model that extends Autoformer to better capture non-stationary patterns.

This study conducts comparative experiments to evaluate the proposed framework. First, we examined the contributions of trading data, technical indicators, composite indices, and other market-related variables. Results showed that these variables contribute unevenly to prediction accuracy, highlighting the need to retain informative features and eliminate redundancy. Next, we evaluated the impact of sentiment by incorporating investor and news sentiment indices. These indices were derived from a domain-specific Chinese financial sentiment dictionary. Incorporating sentiment significantly improved forecasting accuracy, indicating that behavioral signals offer valuable information beyond traditional financial metrics. To optimize input features, we compared five selection algorithms (CatBoost, AdaBoost, XGBoost, Random Forest, and LASSO) and found LASSO consistently yielded the best results, supporting its use in the proposed framework.

To evaluate model effectiveness, we applied NSAutoformer to four major Chinese stock indices: SSECI, SZSECI, GEI, and CSI 300. In benchmark comparisons with nine deep learning models, NSAutoformer consistently achieved the best predictive performance, confirming its ability to capture complex temporal dependencies in financial time series. Building on this foundation, we developed LASSO-NSAutoformer, a hybrid model that integrates LASSO-based feature selection, sentiment analysis, and NSAutoformer. Compared to both single and hybrid baseline models, LASSO-NSAutoformer achieved superior results across all evaluation metrics. It also showed lower error growth and more stable performance across different forecast horizons and input window lengths. These findings demonstrate the model's adaptability to dynamic and evolving market patterns, which is essential for practical stock index forecasting.

In summary, this study proposes and systematically validates a deep learning-based forecasting framework, demonstrating its effectiveness across multiple dimensions. First, the use of multi-source data provides a diverse set of predictors, enabling the model to capture complex market dynamics. Second, the integration of sentiment analysis allows the model to reflect investor responses and market tone. Third, LASSO-based feature selection enhances interpretability by identifying key predictors and reducing redundancy. Finally, the introduction of NSAutoformer yields superior predictive performance compared to existing models, confirming the reliability of deep learning in stock index forecasting.

By extracting sentiment from news, investor comments, and social media, the method helps assess market expectations and investor psychology. This approach is useful for individual investors, asset managers, and financial institutions. It shows the value of combining deep learning with sentiment analysis in financial forecasting and points to future directions for both research and practice.

Despite improved forecasting accuracy, several limitations remain. First, although the model integrates fundamentals, technical indicators, and sentiment, it excludes macroeconomic variables such as GDP growth, interest rates, and inflation expectations. Including these in future work may help capture macroeconomic cycles and enhance model interpretability. Second, the model is only validated on the Chinese stock market. Future research should extend it to developed markets and other asset classes (e.g., futures, bonds, foreign exchange). Third, the integration of feature selection, sentiment analysis, and deep neural networks may incur high computational costs, especially in large-scale applications. Future work should evaluate training and inference efficiency, and explore lightweight alternatives or model pruning. Finally, as stock indices are heavily influenced by a few high-weighted constituents, constructing features based on these components may enhance index-level forecasting accuracy.

Author Contributions: Conceptualization, Z.S.; methodology, Z.S.; software, Z.S.; validation, Z.S. and Q.L.; formal analysis, Z.S.; investigation, Z.S.; resources, Z.S.; data curation, Z.S.; writing—original draft, Z.S.; writing—review and editing, Z.S., Q.L., Y.H. and H.L.; visualization, Z.S.; supervision, Y.H. and H.L.; project administration, Z.S.; funding acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Humanity and Social Science Foundation of the Ministry of Education of China (No. 18YJA630037, 21YJA630054), and the Zhejiang Province Soft Science Research Program Project (No. 2024C350470).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data used in this study were obtained from publicly available sources, as detailed in the manuscript. The financial market data (including stock prices and trading data) are available from the efinance package (https://github.com/Micro-sheep/efinance, accessed on 14 May 2025) and Eastmoney (https://www.eastmoney.com/, accessed on 14 May 2025). The composite indices data were taken from https://www.matteoiacoviello.com/gpr.htm (accessed on 14 May 2025), https://www.policyuncertainty.com/infectious_EMV.html (accessed on 14 May 2025), https://trends.google.com/trends/ (accessed on 14 May 2025), and https://index.baidu.com/v2/index.html (accessed on 14 May 2025). The text data were taken from https://guba.eastmoney.com/ (accessed on 14 May 2025), https://www.chinanews.com/ (accessed on 14 May 2025), http://www.china.com.cn/ (accessed on 14 May 2025), and http://www.csrc.gov.cn/csrc/xwfb/index.shtml (accessed on 14 May 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen, Y.; Wu, J.; Wu, Z. China's commercial bank stock price prediction using a novel K-means-LSTM hybrid approach. *Expert Syst. Appl.* **2022**, *202*, 117370. [CrossRef]
2. Ayyappa, Y.; Kumar, A.P.S. Stock market prediction with political data Analysis (SP-PDA) model for handling big data. *Multimed. Tools Appl.* **2024**, *83*, 80583–80611. [CrossRef]
3. Zhang, Q.; Qin, C.; Zhang, Y.; Bao, F.; Zhang, C.; Liu, P. Transformer-based attention network for stock movement prediction. *Expert Syst. Appl.* **2022**, *202*, 117239. [CrossRef]
4. Billah, M.M.; Sultana, A.; Bhuiyan, F.; Kaosar, M.G. Stock price prediction: Comparison of different moving average techniques using deep learning model. *Neural Comput. Appl.* **2024**, *36*, 5861–5871. [CrossRef]
5. An, Y.; Wang, D.; Chen, L.; Zhang, X. TCP-ARMA: A Tensor-Variate Time Series Forecasting Method. *IEEE Trans. Autom. Sci. Eng.* **2024**, *21*, 2251–2263. [CrossRef]
6. Behera, J.; Kumar, P. An approach to portfolio optimization with time series forecasting algorithms and machine learning techniques. *Appl. Soft Comput.* **2025**, *170*, 112741. [CrossRef]
7. Setoudehtazangi, F.; Manouchehri, T.; Nematollahi, A.; Caporin, M. Time series clustering based on latent volatility mixture modeling with applications in finance. *Math. Comput. Simul.* **2024**, *223*, 543–564. [CrossRef]
8. Zolfaghari, M.; Gholami, S. A hybrid approach of adaptive wavelet transform, long short-term memory and ARIMA-GARCH family models for the stock index prediction. *Expert Syst. Appl.* **2021**, *182*, 115149. [CrossRef]
9. Singh, S.; Parmar, K.S.; Kumar, J. Development of multi-forecasting model using Monte Carlo simulation coupled with wavelet denoising-ARIMA model. *Math. Comput. Simul.* **2025**, *230*, 517–540. [CrossRef]
10. Behera, J.; Pasayat, A.K.; Behera, H.; Kumar, P. Prediction based mean-value-at-risk portfolio optimization using machine learning regression algorithms for multi-national stock markets. *Eng. Appl. Artif. Intell.* **2023**, *120*, 105843. [CrossRef]
11. Kuo, R.; Chiu, T.H. Hybrid of jellyfish and particle swarm optimization algorithm-based support vector machine for stock market trend prediction. *Appl. Soft Comput.* **2024**, *154*, 111394. [CrossRef]
12. Xu, Y.; Dai, Y.; Guo, L.; Chen, J. Leveraging machine learning to forecast carbon returns: Factors from energy markets. *Appl. Energy* **2024**, *357*, 122515. [CrossRef]
13. Beniwal, M.; Singh, A.; Kumar, N. Forecasting multistep daily stock prices for long-term investment decisions: A study of deep learning models on global indices. *Eng. Appl. Artif. Intell.* **2024**, *129*, 107617. [CrossRef]
14. Aydogan-Kilic, D.; Selcuk-Kestel, A.S. Modification of hybrid RNN-HMM model in asset pricing: Univariate and multivariate cases. *Appl. Intell.* **2023**, *53*, 23812–23833. [CrossRef]
15. Zhang, Z.; Liu, Q.; Hu, Y.; Liu, H. Multi-feature stock price prediction by LSTM networks based on VMD and TMFG. *J. Big Data* **2025**, *12*, 74. [CrossRef]
16. Gupta, U.; Bhattacharjee, V.; Bishnu, P.S. StockNet—GRU based stock index prediction. *Expert Syst. Appl.* **2022**, *207*, 117986. [CrossRef]
17. Liu, Z.; Duan, P.; Xue, X.; Zhang, C.; Yue, W.; Zhang, B. A dynamic hypergraph attention network: Capturing market-wide spatiotemporal dependencies for stock selection. *Appl. Soft Comput.* **2025**, *169*, 112524. [CrossRef]
18. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.u.; Polosukhin, I. Attention is All you Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
19. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 11106–11115. [CrossRef]
20. Ren, S.; Wang, X.; Zhou, X.; Zhou, Y. A novel hybrid model for stock price forecasting integrating Encoder Forest and Informer. *Expert Syst. Appl.* **2023**, *234*, 121080. [CrossRef]
21. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–14 December 2021; Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2021; Volume 34, pp. 22419–22430.
22. Liu, Y.; Wu, H.; Wang, J.; Long, M. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022; Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2022; Volume 35, pp. 9881–9893.
23. Nourbakhsh, Z.; Habibi, N. Combining LSTM and CNN methods and fundamental analysis for stock price trend prediction. *Multimed. Tools Appl.* **2023**, *82*, 17769–17799. [CrossRef]
24. Md, A.Q.; Kapoor, S.; A.V., C.J.; Sivaraman, A.K.; Tee, K.F.; H., S.; N., J. Novel optimization approach for stock price forecasting using multi-layered sequential LSTM. *Appl. Soft Comput.* **2023**, *134*, 109830. [CrossRef]

25. Xu, Y.; Liu, T.; Du, P. Volatility forecasting of crude oil futures based on Bi-LSTM-Attention model: The dynamic role of the COVID-19 pandemic and the Russian-Ukrainian conflict. *Resour. Policy* **2024**, *88*, 104319. [CrossRef]
26. Tao, Z.; Wu, W.; Wang, J. Series decomposition Transformer with period-correlation for stock market index prediction. *Expert Syst. Appl.* **2024**, *237*, 121424. [CrossRef]
27. Liu, Q.; Yahyapour, R. Nonlinear parsimonious modeling based on Copula–LoGo. *Expert Syst. Appl.* **2024**, *255*, 124774. [CrossRef]
28. Li, Y.; Zhuang, M.; Wang, J.; Zhou, J. Leveraging multi-time-span sequences and feature correlations for improved stock trend prediction. *Neurocomputing* **2025**, *620*, 129218. [CrossRef]
29. Lin, P.; Ma, S.; Fildes, R. The extra value of online investor sentiment measures on forecasting stock return volatility: A large-scale longitudinal evaluation based on Chinese stock market. *Expert Syst. Appl.* **2024**, *238*, 121927. [CrossRef]
30. Htun, H.H.; Biehl, M.; Petkov, N. Survey of feature selection and extraction techniques for stock market prediction. *Financ. Innov.* **2023**, *9*, 26. [CrossRef]
31. de Oliveira Carosia, A.E.; Coelho, G.P.; da Silva, A.E.A. Investment strategies applied to the Brazilian stock market: A methodology based on Sentiment Analysis with deep learning. *Expert Syst. Appl.* **2021**, *184*, 115470. . [CrossRef]
32. Ji, Z.; Wu, P.; Ling, C.; Zhu, P. Exploring the impact of investor's sentiment tendency in varying input window length for stock price prediction. *Multimed. Tools Appl.* **2023**, *82*, 27415–27449. [CrossRef]
33. Liu, W.J.; Ge, Y.B.; Gu, Y.C. News-driven stock market index prediction based on trellis network and sentiment attention mechanism. *Expert Syst. Appl.* **2024**, *250*, 123966. [CrossRef]
34. Tibshirani, R. Regression Shrinkage and Selection Via the Lasso. *J. R. Stat. Soc. Ser. B (Methodol.)* **2018**, *58*, 267–288.
35. Fischer, T.; Sterling, M.; Lessmann, S. Fx-spot predictions with state-of-the-art transformer and time embeddings. *Expert Syst. Appl.* **2024**, *249*, 123538. [CrossRef]
36. Cho, K.; van Merrienboer, B.; Bahdanau, D.; Bengio, Y. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arXiv* **2014**, arXiv:1409.1259.
37. Mnih, V.; Heess, N.; Graves, A.; Kavukcuoglu, K. Recurrent Models of Visual Attention. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2014; Volume 27.
38. Kitaev, N.; Łukasz, K.; Levskaya, A. Reformer: The Efficient Transformer. *arXiv* **2020**, arXiv:2001.04451
39. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In Proceedings of the 39th International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022; Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., Sabato, S., Eds.; PMLR: Cambridge, MA, USA, 2022; Proceedings of Machine Learning Research; Volume 162, pp. 27268–27286.
40. Obaid, K.; Pukthuanthong, K. A picture is worth a thousand words: Measuring investor sentiment by combining machine learning and photos from news. *J. Financ. Econ.* **2022**, *144*, 273–297. [CrossRef]
41. Yang, K.; Cheng, Z.; Li, M.; Wang, S.; Wei, Y. Fortify the investment performance of crude oil market by integrating sentiment analysis and an interval-based trading strategy. *Appl. Energy* **2024**, *353*, 122102. [CrossRef]
42. Yao, G.; Feng, X.; Wang, Z.; Ji, R.; Zhang, W. Tone, sentiment and market impact: Based on financial sentiment dictionary. *J. Manag. Sci. China* **2021**, *24*, 26–46. [CrossRef]
43. Jiang, F.; Meng, L.; Tang, G. Media Text Sentiment and Stock Return Forecasts. *China Econ. Q.* **2021**, *21*, 1323–1344. [CrossRef]
44. Xu, Y.; Liang, C.; Li, Y.; Huynh, T.L. News sentiment and stock return: Evidence from managers' news coverages. *Financ. Res. Lett.* **2022**, *48*, 102959. [CrossRef]
45. Zhang, Y.; Yan, B.; Aasma, M. A novel deep learning framework: Prediction and analysis of financial time series using CEEMD and LSTM. *Expert Syst. Appl.* **2020**, *159*, 113609. [CrossRef]
46. eFinance: Python Financial Data Interface Library. Available online: https://github.com/Micro-sheep/efinance (accessed on 14 May 2025).
47. Ma, C.; Yan, S. Deep learning in the Chinese stock market: The role of technical indicators. *Financ. Res. Lett.* **2022**, *49*, 103025. [CrossRef]
48. Geopolitical Risk (GPR) Index. Available online: https://www.matteoiacoviello.com/gpr.htm (accessed on 14 May 2025).
49. Daily Infectious Disease Equity Market Volatility Tracker. Available online: https://www.policyuncertainty.com/infectious_EMV.html (accessed on 14 May 2025).
50. Google Trends. Available online: https://trends.google.com/trends/ (accessed on 14 May 2025).
51. Baidu Index. Available online: https://index.baidu.com/v2/index.html (accessed on 14 May 2025).
52. Eastmoney Guba. Available online: https://guba.eastmoney.com/ (accessed on 14 May 2025).
53. ChinaNews. Available online: https://www.chinanews.com/ (accessed on 14 May 2025).
54. China Internet Information Center. Available online: http://www.china.com.cn/ (accessed on 14 May 2025).
55. China Securities Regulatory Commission News Releases. Available online: http://www.csrc.gov.cn/csrc/xwfb/index.shtml (accessed on 14 May 2025).

56. Bazhuayu Web Crawler Software. Available online: https://www.bazhuayu.com/ (accessed on 14 May 2025).
57. Liu, Q.; Yahyapour, R.; Liu, H.; Hu, Y. A novel combining method of dynamic and static web crawler with parallel computing. *Multimed. Tools Appl.* **2024**, *83*, 60343–60364. [CrossRef]