Fluctuation and Noise Letters
Vol. 23, No. 2 (2024) 2440006 (27 pages)

DOI: 10.1142/S0219477524400066

World Scientific
www.worldscientific.com

Graphical Deep Learning Prediction Model for Stock Risk Management

Haewon Byeon[*,**], Shyamsunder Chitta[†,††], Shavkatov Navruzbek Shavkatovich[‡,‡‡], Ghulam Jillani Ansari[§,§§], Majed Alhaisoni[¶,¶¶] and Yu-Dong Zhang[‖,‖‖]

[*]*Department of Digital Anti-Aging Healthcare*
Inje University, Gimhae, Republic of Korea 50834

[†]*Symbiosis Institute of Business Management Hyderabad*
Symbiosis International (Deemed University), India

[‡]*Department of Corporate Finance and Securities*
Tashkent Institute of Finance, Tashkent, Uzbekistan

[§]*Department of Information Sciences*
University of Education, Lahore, Pakistan

[¶]*Computer Sciences Department*
College of Computer and Information Sciences
Princess Nourah Bint Abdulrahman University
Riyadh, Saudi Arabia

[‖]*Department of Informatics*
University of Leicester, Leicester, UK
[**]*bhwpuma@naver.com*
[††]*shyam.chitta@sibmhyd.edu.in*
[‡‡]*shavkatov_navruzbek@tfi.uz*
[§§]*ghulamjillani@ue.edu.pk*
[¶¶]*MMAlhaisoni@pnu.edu.sa*
[‖‖]*yudongzhang@ieee.org*

Received 30 August 2023
Accepted 9 October 2023
Published 20 December 2023

Communicated by Muhammad Attique Khan

Forecasting the future movements of stock market indexes by utilizing historical transaction data is a prominent concern within the realm of finance. The application of graph convolutional networks to incorporate the interrelationships among various indices' patterns is a highly advanced subject within this field. Addressing the inconsistency between historical and future dynamic graphs in current graph convolution-based index prediction, we propose a method called G-Conv that constructs a graph structure based on constituent stocks of the indices for index trend prediction. This approach extracts traditional quantitative features along with deep features from one-dimensional convolutional networks as characteristics of prediction samples. The method produces index trend predictions by constructing a graph structure using constituent stock data of indices and applying graph convolution to different index sample

[**]Corresponding author.

features. The proposed methodology's efficacy is verified by utilizing 42 widely employed indicators in the A-share market. The experimental findings demonstrate that when utilizing mean absolute error (MAE) and mean squared error (MSE) as the loss functions for model training, G-Conv outperforms classic methods such as GC-CNN and ADGAT. Specifically, G-Conv reduces the average prediction errors by 5.10% and 4.20% respectively, as evaluated by the two error criteria. Additionally, G-Conv exhibits favorable generalization performance.

Keywords: Stock market prediction; graph convolutional networks; index trends; constituent stocks; financial forecasting.

1. Introduction

An important topic in the realm of financial investment is forecasting stock market movements using previous transaction data [1]. Accurately gauging the valuation of an individual stock proves challenging, owing to the ever-fluctuating essence inherent in stock prices and their vulnerability to unforeseen factors like corporate operations and public opinion. Stock market indices are indicators compiled according to specific principles that reflect the overall trend of a basket of stocks. In comparison to individual stocks, indices are less influenced by factors like the operational status of individual companies, resulting in better predictability [2]. Setting and meeting the right standards with your internal and external stakeholders will enable you to gain their trust, inspire their confidence and consistently and on schedule offer value to your end customers. The capacity of a team to reach a high level of predictability will depend on circumstances specific to each organization, yet there are standardized practices that every business should use but probably does not. Techniques begin with understanding and expanding on the principles of scrum while utilizing certain metrics that will aid in your teams' ongoing improvement and increased predictability. One of the most challenging tasks in software development is achieving a high degree of predictability.

Presently, there are numerous indices in the domestic securities market. Predicting index trends not only aids investors in mitigating investment risks and enhancing returns but also offers insight for formulating economic policies. Stock market indices serve as a gauge for the state of the market. It reflects stock market conduct as represented by the movements of all the stocks. They serve as a yardstick to assess the success of a portfolio. They aid in more effective resource allocation in businesses. Additionally, they support future stock prices and business cycle forecasting. The benchmark indexes of the Indian stock market are CNX NIFTY and BSE SENSEX. Both indexes display market activity. Sectoral Indices indicate a certain economic sector. Stocks' market capitalization serves as the basis for market-cap-based indexes. A stock's market capitalization provides information about its worth. It is the result of the market price at the time and the number of outstanding shares in the firm. Free Float Market Capitalization Weighted is the process used to calculate market capitalization.

The SADIE (spatial analysis by distance indices) data analysis methodology is an effective tool for assessing organism patterns (in terms of patches and gaps) and evaluating the randomness of the patterns. SADIE was recently created to assess spatial patterns of organisms for either geographically referenced sample units or individuals. The most common application of SADIE for spatially referenced sampling units is to quantify patterns by the total distance that individuals must be moved between sampling units for the data to be as regular as possible (i.e., there are the same number of individuals in each sampling unit). To examine the long- and short-term relationships among regional stock indexes, time series approaches are more suitable. The best time series approaches for examining long- and short-term links include cointegration, causal and innovation analyses. Strong economic links, coordinated policy and trade between the relevant areas may cause their stock indexes to be indirectly linked.

In that sequence, time series analysis approaches conventional machine learning methods, and deep learning methods have been the three main phases of the decades-long research into forecasting stock market movements [3]. The four phases of the stock market cycle are accumulation, mark-up (run-up), distribution and mark-down (run-down). You will benefit from having a solid understanding of each phase's operation if you analyze charts for trading and use technical analysis. As soon as a new firm is launched on the public exchanges, it goes through the accumulation period, which is the first stage of the stock lifecycle. Strong resistance has built at the top of the charts throughout the accumulation period, creating a range. A month to many years may pass throughout the time span. The type of the business and the success of the firm are key factors. When the accumulation phase has overcome its opposition, the run-up phase starts to define new and higher stock values. The traders or investors who missed their chance to accumulate in the first phase are now purchasing the stock vehemently in this phase. As a result, a large number of shares for that particular stock are trading during the run-up period. The amassed share traders start to sell the security during the distribution phase, the third stage of the stock market cycle. The reversal stage is another name for this phase. The decreasing stage of the stock lifecycle is the final stage, which is not a good moment for investors and long traders to enter. Due to the traders' purchases of equities during the distribution phase, there is a sell-off during this period. By maintaining a small number of buyers for the stock, the demand for security drops quickly.

The Auto Regressive Moving Average (ARMA) model is the most traditional approach in time series analysis [4]. However, this methodology has gradually been replaced by later ones, with fewer pertinent investigations in recent years, because time series models are unable to capture nonlinear temporal interactions. Decision trees [5], random forests [6], support vector machines [7] and shallow neural networks [8] are a few examples of traditional machine learning techniques. An increasingly common machine learning method used in quantitative trading and investment is decision trees. These algorithms have developed into a useful resource for traders and investors wanting to get a competitive advantage in the markets. They may be used

to examine past financial data and uncover patterns that guide investment decisions. These algorithms divide a dataset recursively into smaller subsets, with each split denoting a choice or a forecast. The result is a tree-like structure, with each internal node standing in for a choice and each leaf node for a forecast. Decision tree algorithms may be used to examine past financial data and spot trends that might help investors make better choices.

A supervised classification technique used in machine learning, the random forest algorithm takes an ensemble approach to solve the problem of overfitting in decision trees. It comprises several decision trees built at random using characteristics from the dataset. By combining the results from the decision trees and using the most common forecast, the random forest's final prediction is established. The random forest considers the conclusion that is reached the most times through the multiple decision trees to be the ultimate result.

One of the most well-liked supervised learning algorithms, Support Vector Machine, or SVM, is used to solve Classification and Regression issues. The SVM algorithm's objective is to establish the optimal line or decision boundary that can divide n-dimensional space into classes, allowing us to quickly classify fresh data points in the future. A hyperplane is the name given to this optimal decision boundary. SVM selects the extreme vectors and points that aid in the creation of the hyperplane. Support vectors, which are used to represent these extreme instances, form the basis for the SVM method.

These techniques continue to be used extensively in the world of investments because they may identify nonlinear correlations in time-series data. However, conventional machine learning approaches have inherent limitations in forecasting stock market trends since they require manual feature extraction from trading data and are unable to find deep-level connections between time-series data. Artificial neural networks (ANNs) and support vector machines (SVM), two examples of machine learning models, have been frequently utilized to forecast financial time series with remarkably accurate results. Due to their capacity to simulate complicated nonlinear topologies, deep learning models have more recently been used to tackle this issue. Deep learning can represent difficult real-world data correctly and more effectively than typical machine learning models by identifying strong features that capture the necessary data. The best of both worlds is offered by the deep reinforcement learning technique, which enables agents to learn deep features from training data without requiring a labeled data set and allows for the customization of certain reward functions.

As deep learning techniques have gained success in fields such as image analysis and natural language processing, an increasing number of researchers have begun applying deep learning models to stock market prediction [9]. Image analysis is the process of extracting useful information, primarily from digitally stored images, using a variety of processing techniques. However, each of these techniques may only be useful for a limited number of tasks, as there are currently no known methods of image analysis that are general enough to be used for a variety of tasks, as opposed to

a human's image-analysis skills. On the other hand, natural language processing (NLP) is a branch of artificial intelligence in which computer methods are used to analyze and synthesize voice and natural language. Deep learning and neural network applications in natural language processing are just a few of the innovations that have led to the development of speech-to-speech translation engines and spoken dialogue systems, as well as the mining of social media for financial and health-related information and detecting sentiment and emotion toward goods and services. Recent developments in deep learning allow us to analyze a huge number of photos effectively; yet, the pace of manual human work has largely hampered the collection of such big datasets. To accommodate the natural language information for image analysis, medical pictures are often preserved with the radiological reports that go with them.

Deep learning models do not require manual feature extraction; they use raw sequential data as input and allow neural networks to automatically learn the data's features and make predictions. The term sequential data refers to data where individual points in the dataset depend on one another. These sorts of data are often difficult for conventional neural networks to manage. The researchers have called for new paradigms of data analysis to derive relevant insights from it as a result of the availability of the data. Predictions using such analysis may be useful in several fields, including fraud detection, cyber security and medical informatics. A further issue with too much information is that it gives users of web apps too many options. To make predictions and judgments, machine learning algorithms build models that can learn from the data without being explicitly programmed. Deep learning develops feature representations from the data itself, unlike task-specific machine learning algorithms. Deep learning was able to produce futuristic results for incredibly difficult problems across a variety of disciplines thanks to these characteristics; especially in the fields of health care, computer vision, information retrieval and natural language processing.

Various techniques commonly employed in the field encompass multi-layer feedforward neural networks [10], recurrent neural networks (RNNs) [11, 12], deep belief networks [13], convolution neural networks (CNNs) [14], hybrid models [15, 16] and graph convolutional neural networks (GCNs) [17–25]. Multi-layer feedforward neural networks and deep belief networks are utilized to develop mapping linkages between time-series data and potential updates through the incorporation of numerous fully connected layers. The Multilayer Feed-Forward Neural Network (MFFNN) is a multi-layered, linked artificial neural network with neurons that have weights attached to them. The activation functions are used to compute the output. It is one of the varieties of neural networks in which the signal flow is from input to output units and there are no loops, feedback or signal movement from output to hidden and input layer. The activation process of a neuron acts as a decision-making body at the output. Either linear or nonlinear decision limitations are learned by the neuron depending on the activation function. Additionally, it has a leveling effect on neuron output, preventing the cascade effect from causing neurons' output to grow

too large after numerous layers. The simplest feed-forward neural network could seem like a single-layer perceptron. This model multiplies inputs as they enter the layer by weights. The weighted input values are then added collectively to determine the final total.

Using a deep architecture, a Deep Belief Network (DBN) is a powerful generative model. Deep layered networks have problems with traditional neural networks, thus we develop Deep Belief Networks (DBNs) to solve these problems. A DBN consists of multiple layers of stochastic latent variables. Binary latent variables, often called feature detectors or hidden units, are a subclass of latent variables. DBN is a hybrid model for generating graphics. The two uppermost levels are undirected. Directed linkages connect the upper levels to the lower layers. DBN is a method for probabilistic unsupervised deep learning.

Nevertheless, the foremost obstacle encountered by these networks is the vanishing gradient problem, which arises as a result of the deep layers inside the optimization procedure. Convolutional neural networks (CNNs) employ pooling processes to effectively reduce the number of model parameters. By downsampling feature maps from the preceding layer, pooling layers produce new feature maps. This layer has two purposes: one is to lower computing costs by decreasing the number of parameters or weights, and the other is to prevent the model from remembering. The pooling layer separates the features of the convolution layer into groups dependent on the pooling size. Each group is expressed as a single value via downsampling. As a consequence, a new feature map with a reduced size has been created. The pooling layer attempts to retain the pixel values of the input data while expressing them in a smaller size.

Additionally, they utilize local convolutions to analyze and establish connections between various data sequences. Nevertheless, CNNs require locally ordered input data, a constraint unsuitable for modeling stock market indices with independent changing trends. Recurrent neural networks excel at modeling historical sequences and currently dominate deep models in stock market prediction. However, RNNs also cannot model multi-index sequences with non-Euclidean relationships. GCNs, emerging in recent years for predictive modeling of graph data, can establish relationships between non-Euclidean predictive variables, making them highly suitable for addressing the prediction of multi-sequence data with correlated trends. Any previous business transactions that were conducted before a specific period are referred to as historical data (historical transactions). This phrase may also be used to describe the transactions that the company manually entered before using accounting software. Transactional data is the most important since it enables several inferences about a person's personal life and real activity. The main components are a date, account or credit card number, beneficiary account information, a description of the purchase and the cost. The most private information may be viewed in even greater depth on a payment statement.

Using GCNs to predict stock market trends faces the fundamental challenge of how to construct graph structures between different indices. Graph generation

approaches can be categorized into two major types: static and dynamic. Static methods utilize inherent domain knowledge and attributes of stocks to calculate correlations between them. For instance, Ahmad *et al.* [18] establish a graph structure between stocks based on business association information among companies, Yun *et al.* [19] construct hypergraphs between stocks using industry relationships and fund holding data, and Zhang *et al.* [20] establish stock associations using shareholder ownership data. These methods require collecting specific stock-related data from areas like supply chains and shareholders. However, since indices lack such data, these methods are not applicable for constructing graph structures between indices.

Data structures called graphs are made up of nodes or vertices connected by edges. Graphs are used to represent the connections and interconnections between various components, which make it easier to simulate and assess complex systems. A nonlinear data structure is a graph. It consists of a group of nodes joined together by edges. A data field may be found in each node. The mapping between the graph's vertices and edges is represented by an adjacency matrix. It may depict the weighted undirected, weighted directed, undirected and directed graphs. A graph is represented as an adjacency list using linked lists. A linked list that contains the value of the node and a pointer to the node next to it is present for each node in a graph.

Dynamic methods, on the other hand, calculate correlations between stocks dynamically based on their historical trends, allowing for the discovery of non-obvious relationships. For example, Cosgun [21] and Thakkar and Chaudhari [22] dynamically calculate the graph structure between stocks using the Spearman rank correlation coefficient based on historical trend data. Chen *et al.* [23] employ Hawkes processes to unearth associations between different stocks. Chou *et al.* [24] and Zhao *et al.* [25] generate graph structures using dynamic optimization methods. The graphs obtained through these methods depend on historical data, but the high randomness of the stock market means that the correlations between indices can change with shifts in the economic situation. Consequently, historical graphs are often inadequate for predicting future trends. Establishing a non-dynamic graph structure between indices remains a current research challenge. Furthermore, as graph convolutional networks need to model multiple indices simultaneously, and since different indices have significant point differences, normalizing multiple indices with varying points to a common scale is also a problem that needs resolution.

Because dynamic graph data changes over time, it is important to manage historical data to obtain a snapshot graph at a certain period. Continuously changing dynamic graphs create a vast quantity of historical data. A historical graph is necessary to handle the continual changes in the vertices and edges that make up the graph to trace the history of changes in graphs or to search for graphs at a certain period in the past in a dynamic environment. Historical graphs store changes in vertices and edges that have occurred over time in the initial graph as well as changes in the final graphs, allowing us to analyze graph changes and view the graph's status at a given time. The Version Graph offers high space efficiency from a spatial

standpoint since it just stores the change history and not all graphs. Due to the necessity of progressively accessing the change history from the first graph to examine the graph data at the necessary time, it suffers from the problem of sluggish access speed.

In this study, the future price changes of commonly observed indices in the A-share market are used as the prediction target. Traditional quantitative features and one-dimensional convolutional deep features are employed as the predictive sample features. The goal of investing in the financial market is to reap more rewards; but, because of the complexity of the market and the numerous factors that impact it, forecasting future market dynamics is difficult. The manners of stock market investors may necessitate the examination of several related aspects and the extraction of relevant data for accurate forecasting. Fusion may be thought of as a strategy to combine data or qualities in general and improve prediction based on a combinational strategy that can support one another. Fusion may be viewed as the process of integrating several components that can increase performance and provide beneficial outcomes. It can be thought of as a change of one or more aspects to create an effective representation. Exploring and making use of the environment requires a variety of data, characteristics, techniques, parameters, etc.; this information may be combined to improve performance. Information fusion involves dealing with the processed and/or generated data, whereas data fusion is the fusion conducted on the raw form of data.

GCNs are used to model index relationships. Given the existing challenges, this study introduces several innovations. First, to address the absence of a normalization method for multi-index modeling data when extracting one-dimensional convolutional features, a normalization approach that combines linear normalization and Sigmoid transformation is proposed. Second, to tackle the inconsistency between historical and future dynamic graphs for indices, a method is introduced that constructs an index graph based on index constituent stock data and employs convolution. Third, the effectiveness of this approach is validated using 42 indices in the A-share market, and the results demonstrate a significant reduction in prediction error.

2. Index Prediction Framework

The framework for index prediction in this paper comprises two main components: the feature extraction layer and the graph convolutional layer, as illustrated in Fig. 1. The model structure is depicted in the figure. The feature extraction layer is designed to extract quantitative and deep features from the raw index data of the samples. Before using an ML model to predict outcomes, a feature selection procedure should be carried out to choose important characteristics from the initial feature set. The feature selection procedure also enhances the performance of ML models by lowering the number of unnecessary variables, the cost of computation and the overfitting issue. If we just choose a few characteristics to use as input for an ML model, there may not be enough data to generate predictions. Due to the curse of dimensionality, a high number of features also lengthen the running time and impair

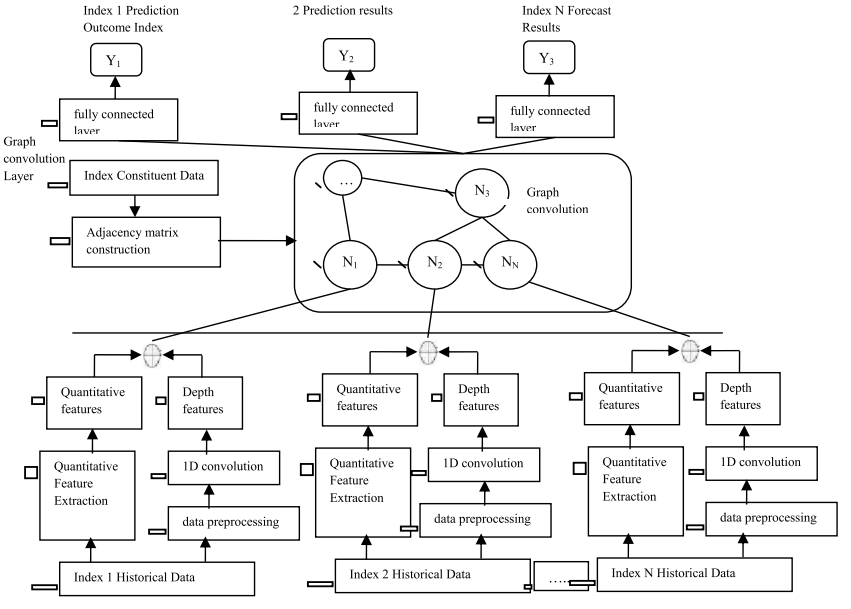


Fig. 1. Index forecasting framework.

generalization performance. To make accurate forecasts, only the most important factors that have an impact on the outcomes should be chosen.

This layer takes historical K-line data and valuation data of the indices as input and yields extracted sample features as output. Quantitative features are traditional financial indicators used in investment, including metrics like price-to-earnings ratio, moving averages and relative strength index. This aspect will be elaborated upon in Sec. 2.1. Deep features are features computed using a one-dimensional CNN. Several techniques for data characterization and feature extraction can create quantitative characteristics, which have been effectively employed as biomarkers. These characteristics provide details on a nodule, such as its size, pixel intensity, histogram-based data and texture data derived from wavelets or a convolution kernel. The last layers of a CNN, which come before the classification layer, are often where deep picture characteristics are retrieved. The CNN learns to detect distinct patterns and textures by practicing a variety of picture types. To handle challenging issues (such as picture colorization, classification, segmentation and detection), deep learning (DL) is utilized in the field of digital image processing. With the use of massive data and plenty of processing power, DL techniques like CNNs have pushed the envelope of what is feasible and primarily increased prediction performance. Super-human precision is being used to address issues that were formerly thought to be insurmountable. One excellent example of this is image categorization.

This component first transforms index points into price changes and normalizes the data to mitigate the impact of differences in index point values. Subsequently, a

one-dimensional CNN is employed to perform convolutions on the samples and derive deep features. The feature extraction layer will be discussed in Sec. 2.2.

The graph convolutional layer takes index constituent stock data and the deep features from the feature extraction layer as input and produces prediction labels for different indices as output. This layer initially constructs a graph structure among indices using index constituent stock data, with vertices representing indices and edges representing the correlation weights between indices. It then performs graph convolution operations on the sample features extracted in the feature extraction layer to facilitate the fusion of sample features across different indices. Finally, the index features, post graph convolution, are mapped to the price changes of the indices through a fully connected layer. The graph convolutional layer will be detailed in Sec. 3.

3. Feature Extraction Layer

The feature extraction layer comprises two components: quantitative feature extraction and deep feature extraction. Quantitative features refer to commonly used technical indicators in the field of financial investment, including metrics like price-to-earnings ratios and moving averages. Deep features are features automatically extracted from historical trading data using a one-dimensional CNN. These two types of features are detailed in Secs. 2.1 and 2.2, respectively.

The historical data used in this study includes index valuation data and trading data. Valuation data consists of daily-related metrics such as price-to-earnings ratio, price-to-book ratio and turnover ratio. Trading data comprises 5-min interval K-line data. K-line data is represented as $g = (g_1, g_2, \ldots, g_5)$, with each of the five elements representing open, high, low, close and volume values. Index trading data is represented as $G = \{g_1, g_2, \ldots, g_U\}$, and $Gj = \{g_{1j}, g_{2j}, \ldots, g_{Uj}\}$ represents the sequence composed of all jth elements, where U is the length of historical trading data. This study uses historical data from the past b_1 days to predict the average index price changes for the next b_2 days. Before performing feature extraction, the trading sequence data is partitioned into samples. The calculation method for the uth sample and its label is shown as follows:

$$x_u = \{x_u^1, x_u^2, \ldots, x_u^{b_1 \times k}\} = \{g_{(u-1)\times k+1}, \ldots, g_{(u-1)\times k+b_1 \times k}\}, \tag{1}$$

$$y_u = \frac{1}{b_2 g_{((u-1+b_1)\times k)}} \sum_{i=0}^{b_2-1} g_{((u+b_1+i)\times k)} - 1. \tag{2}$$

Among them, x_u^i is the ith K-line in the sample x_u, k is the number of K-lines per day and $k = 48$ for 5-min K-line data. After dividing the samples, the sample set containing labeled data is denoted by $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_u, y_u)\}$.

Table 1. Quantitative features.

Index Abbreviation	Index full name	Indicator description
P/E	P/E Ratio	3-day and 5-day averages
P/B	Price-to-book ratio	3-day and 5-day averages
—	Turnover rate	3-day and 5-day averages
ROC	Rate of change	Calculate the values for the 3rd and 5th days separately
WMA	Weighted moving average	Calculate the values for the 3rd and 5th days separately
HMA	Hull Moving Average	Calculate the values for the 3rd and 5th days separately
EMA	Exponential Moving Average	Calculate the values for the 3rd and 5th days separately
TEMA	Triple Exponential Moving Average	Calculate the values for the 3rd and 5th days separately
CMO	Chande Momentum Oscillator	Calculate the values for the 3rd and 5th days separately
WR	William index	Calculate the values for the 3rd and 5th days separately
RSI	Relative strength index	Calculate the values for the 5th and 9th days, respectively

3.1. *Quantitative feature extraction*

Drawing from methods outlined in [27–39], the quantitative features extracted in this study are presented in Table 1.

The values for the price-to-earnings ratio, price-to-book ratio and turnover ratio represent the average over the last 3 and 5 days. The rate of change signifies the percentage change between the closing price of the current day and the closing price N days ago. This includes the 3-day and 5-day rate of change, computed as follows:

$$\mathrm{ROC}_u(n) = \frac{x_u^{(b_1\times k)} - x_t^{((b_1-n)\times k)4}}{x_u^{((b_1-n)\times k)4}}. \tag{3}$$

Weighted Moving Average and Hull Moving Average are used to calculate the weighted average of the closing prices for the most recent n days. This includes the 3-day and 5-day moving averages. The calculation methods are as follows:

$$\mathrm{WMA}_u(x_u, n) = \frac{\sum_{j=1}^{n} j x_u^{((b_1-n+j)\times k)4}}{n}, \tag{4}$$

$$\mathrm{HMA}_u(x_u, n) = \mathrm{WMA}_u. \tag{5}$$

Exponential Moving Average (EMA) and Triple Exponential Moving Average (TEMA) are also types of moving average indicators. They differ from the weighted moving average in terms of how the weights are calculated. The calculation methods for these two indicators are presented as follows:

$$(x_u, n) = \alpha \times x_u^{(b_1\times k)4} + (1-\alpha) \times \mathrm{EMA}_{u-1}(x_{u-1}, n-1), \tag{6}$$

$$(x_u, n) = 3 \times (x_u, n) - 3 \times ((x_u, n), n) + \mathrm{EMA}_u(\mathrm{EMA}_u(\mathrm{EMA}_u(x_u, n), n), n), \quad (7)$$

where $\alpha = 2/(n+1)$ is the moving weight. The Chande Momentum Oscillator, Williams Indicator and Relative Strength Indicator are used to measure the overbought and oversold levels of buying and selling, and the calculation methods are as follows:

$$\mathrm{CMO}_u(n) = \frac{T_v - T_e}{T_v + T_e}, \quad (8)$$

$$\mathrm{WR}_u(n) = \frac{\max\{G_2\} - x_u^{(b_1 \times k)4}}{\max\{G_2\} - \min\{G_3\}}, \quad (9)$$

$$\mathrm{RSI}_u(n) = \frac{\mathrm{EMA}_u(V, n)}{\mathrm{EMA}_u(V, n) + \mathrm{EMA}_u(E, n)}, \quad (10)$$

where $T_v \wedge T_e$, respectively, represent the sum of momentum for rising and falling days within the past n days, G_2 and G_3 signify the sequences of highest and lowest points within the last n days, and V and E represent the sequences of upward and downward price changes within the past n days. This is indicative of the quantified feature vector x_u extracted from the sample o_u.

3.2. *Deep feature extraction*

3.2.1. *Data preprocessing*

To mitigate the influence of differences in index values, the sample's index values y_u^{ij} are first transformed into percentage changes as follows:

$$y_u^{ij} = \begin{cases} \dfrac{y_u^{ij} - y_u^{(i-1)4}}{y_u^{(i-1)4}}, j = 1 \dfrac{y_u^{ij} - y_u^{(i-1)j}}{y_u^{(i-1)j}}, j = 5 \dfrac{y_u^{ij} - y_u^{i1}}{y_u^{i1}} j = 2, 3, 4. \end{cases} \quad (11)$$

As shown in Eq. (11), the calculation of percentage changes differs for different components of the K-line. For instance, the opening price is compared to the previous moment's closing price, while the volume is compared to the previous moment's trading volume. Similarly, the highest, lowest and closing prices are calculated relative to the current moment's opening price. After converting to percentage changes, the samples are normalized using the method detailed in the equation:

$$x_t^{nij} = \frac{x_t^{\prime ij} - x^j}{\mathrm{pre}(X'_j, \alpha) - \mathrm{pre}(X'_j, 1-\alpha)}. \quad (12)$$

Among them, Y'_j is the set of all y'^{ij} in the sample set, y^j is the mean value of elements in Y'_j, $\mathrm{pre}(X'_j, \alpha)$ is defined as an element in Y'_j, and satisfies Y'_j. The pieces with a proportion of α are smaller than $\mathrm{pre}(X'_j, \alpha)$, where $0.5 < \alpha < 1$ is a variable parameter used to control the normalization range of the data. The larger the value, the more dispersed the data after the execution of formula (12); on the contrary, the

data is more concentrated towards 0. According to the characteristics of the K-line rise and fall data, after normalization using formula (12), the data is mainly focused between -1 and 1. However, a tiny portion of the data lies outside of this band. To reduce this part of the model training process, the influence of data on the gradient, perform Sigmoid transformation on the result of the formula (12), as follows:

$$y_u^{ij} = 2\,\mathrm{Sigmod}(y_u^{ij}) - 1 = \frac{f^{y_u^{ij}} - 1}{f^{y_{ij}^{ij}} + 1}. \tag{13}$$

After formula (13) is executed, the data is normalized to be between -1 and 1. The normalized sample is denoted as y_u, used as the input of one-dimensional convolution.

3.2.2. *One-dimensional convolution*

After normalizing the samples, use a one-dimensional convolutional network to extract sample depth features. The network input is the sample y_u, whose index is at time u, and the output is the convolutional depth feature o_t'. The one-dimensional convolutional network structure is shown in Fig. 2.

As shown in Fig. 2, the input of the one-dimensional convolutional layer is a matrix of $\tau_0 \times n_i^0$, where $\tau_0 = b_1 \times k$ is the length of the data in the sample, and $n_i^0 = 5$ is the number of input channels. The one-dimensional convolutional network includes p convolutional pooling layers. The jth convolutional pooling layer has n_o^j convolution kernels, and the size of each convolution kernel is $n_k^j \times n_i^{j-1} - 1$. During the convolution process, each seed slides the input data along the time direction and calculates the convolution in every sliding step. The calculation method of the nth row and the f-column in the convolution result ξ_j^{tmp} of the jth layer is shown as follows:

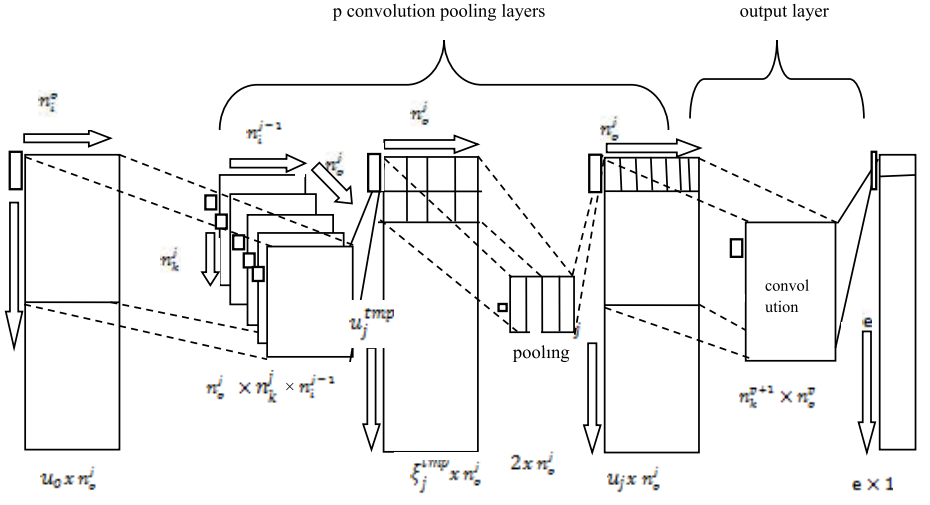


Fig. 2. One-dimensional convolutional network structure.

$$(\xi_j^{tmp})_f^g = \sum_{c=f}^{f+n_k^j n_i^{j-1}} \sum_{e=1}^{g} (\xi_{j-1})_c^e (c_j^g)_{c-f+1}^e, \tag{14}$$

where $(\xi_{j-1})_c^e$ is the value of d in line b of the jth layer convolution input data, $(c_j^g)_{c-f+1}^e$ is the fth convolution kernel of the jth layer. The value of column d at row $c-d+1$ in. After all the convolution kernels are convoluted, the output result of $\xi_j^{tmp} \times n_o^j$ dimension is obtained, where $\xi_j^{tmp} = \xi_{j-1} - n_k^j + 1$. After receiving the convolution result, use maximum pooling to reduce the amount of data. The calculation method of pooling is as follows:

$$(\xi_j)_f^g = \max. \tag{15}$$

Among them, $(\xi_j)_f^g$ is the value of the nth row and column g of the jth layer pooling output result, which is the more significant value of the convolution result ξ_j^g in the time dimension of the two adjacent data. The data dimension after pooling is $\tau_j \times n_o^j$, where $\tau_j = \xi_j^g$. After p convolution and pooling layers, a convolutional layer is used to map the output of the convolutional pooling layer to the depth feature o_t' of the sample. After the quantization feature is connected with the depth feature, the part out of the model is obtained and is represented by o_t'. The quality of the tth sample of index I and the elements of all the tth pieces of the index are denoted as O_t.

4. Graph Convolution Layers

To reflect the trend correlation of different indices in the forecasting process, the graph convolutional layer uses the graph convolutional network to fuse quantitative and deep features. This section first introduces index graph construction based on index constituent data and then presents index feature fusion based on graph convolution.

4.1. *Index graph construction*

The primary problem of using graph convolutional neural networks for exponential feature fusion is constructing a graph structure between indices. A graph is represented by $H = \{W, F\}$, where W is the set of vertices and F is the set of edges. In the model of this paper, the vertex represents the index, and the edge represents the relationship between the indices. Edges are represented by an adjacency matrix $B = (b_{ij})_{N \times N}$, where N is the number of vertices, and b means the association weight between vertices i and j. Since the index is obtained by weighting multiple constituent stocks, two indices with the same constituent stocks are related in trend, so this paper uses the proportion of the same constituent stocks in the index to calculate the adjacency matrix. Assuming that the constituent stocks and weights of index I are represented by the set $D_i = \{(d_1, x_1)(ind_2, x_2), \ldots, (ind_{n_i}, x_{n_i})$ the calculation

Algorithm 1. Construction Algorithm of Graph Adjacency Matrix Based on Index Component Stock Data

Input: constituent stock data D_i and D_j of index I and index j;
Output: the weight of index I and index j in the adjacency matrix;

1. Let $b_{ij} = 0$;
2. Execute for $\forall(\text{ind}_i, x_i) \in D_i$:
3. Execute for $\forall(\text{ind}_j, x_j) \in D_j$:
4. If $\text{ind}_i = \text{ind}_j$, then:
5. $b_{ij} = bj + x_j$
6. Return b_{ij};

method of edge aid of index I and index j in the adjacency matrix is as shown in Algorithm 1.

As shown in Algorithm 1, the percentage of indices A and J that share components can be calculated using the edge and adjacency matrices. This method of computation has the meaning that is described below. If the stocks that make up indices I and j are identical, then a change in indices I will weigh and indices j will weigh-off. If the proportion is large, the impact on index j will be more significant when index I changes. The convolution processes will exhibit increased robustness when utilizing the same convolution kernel settings, hence resulting in a stronger correlation. The sum of specific gravity is calculated. The adjacency matrix b_{ij} calculated by this method is an asymmetric matrix, and $0 \leq b_{ij} \leq 1$.

4.2. *Graph convolution*

The index graph structure was constructed in the previous section to model the trend correlation between different indices. Based on this graph, this section fuses the characteristics of other indices through the graph convolution operation to integrate the index trend correlation in the prediction results. The generalized graph convolution definition is shown as follows:

$$A_u = B_u^U O_u X. \tag{16}$$

Among them, B_u^U is the transpose of the adjacency matrix at time u, calculated by Algorithm 1. All exponential sample features are extracted by the feature extraction layer at time u, and the size is $N \times f$. X is an $e \times o$ dimensional convolution kernel parameter, which needs to be dynamically updated during model training. As per the formal definition of the adjacency matrix, the ith row in matrix At signifies the impact of index I on the other indices. Similarly, when the matrix At is transposed and convolved, the ith row of the resulting matrix represents the influence of various indices o_u^i. Equation (16) represents a convolution operation of the graph, which incorporates the power of directly connected indices. Still, there is usually an association between the index connected by multiple steps in the chart, and this

association needs to be realized by multi-step convolution, as follows:

$$(A_u)_\mu = Q_u^\mu P_u X. \tag{17}$$

Among them, $Q_u = B_u^U / rowsum(B_u^U)$ represents the normalized transposed adjacency matrix, and the convolution considers the influence between the indices whose distance is μ steps. To fuse the results of different convolution steps in the final convolution result, the convolution on a single convolution kernel is defined as the sum of multi-step convolutions, as follows:

$$(A_u)_s = \sum_{q=0}^{\psi} Q_u^\mu P_u X. \tag{18}$$

To enhance the convolutional model's capacity for universality, a multi-convolution kernel convolution is introduced, which is formulated according to Eq. (18) and presented as follows:

$$(A_u)_{\text{out}} = \sum_{\infty=1}^{\pi} \sum_{\mu=0}^{\psi} Q_u^\mu P_u X_\infty. \tag{19}$$

Among them, π is the number of convolution kernels, and X_x is the parameter of the xth convolution kernel. After performing graph convolution on the input feature P_u, the $N \times o$ convolution feature $(A_u)_{\text{out}}$ is obtained. Compared with P_u, this feature incorporates different exponential trend correlations. After receiving the output of the convolutional layer, use the fully connected layer to map the convolutional part $(A_u)_{\text{out}}$ to the prediction result of each index, as follows:

$$x_u^i = \tanh((A_u^i)_{\text{out}} Y_{\text{out}}^i). \tag{20}$$

Among them, x_u^i is the prediction result of index I sample t, $(A_u^i)_{\text{out}}$ is the ith line of $(A_u)_{\text{out}}$, and without is the output layer parameter of index i.

5. Experimental Setup

To verify the model's effectiveness in this paper, the method is simulated using the historical data of the A-share index. This section first introduces the design of the experiment, including the selection of data sets and the construction and training methods of the model, then presents the comparison results with existing processes, and finally introduces the impact of the number of model training iterations. Stock predictions may be made in a variety of methods. Fundamental analysis and technical analysis are the two often employed techniques. Using the expertise of financial professionals, fundamental analysis is a subjective analytical technique. This approach is based on broad data, such as the financial and operational health of the organization. To make predictions and draw conclusions about the stock's future

Table 2. Index list.

Index name	Index code	Index name	Index code
The Shanghai Composite Index	1	Securities Central Enterprises	926
A share index	2	Securities Private Enterprises	938
SSE 380	9	Securities State-owned Enterprises	955
SSE 180	10	CSI Upstream	961
SSE 50	16	CSI Midstream	962
CSI 300	300	CSI Downstream	963
Fundamentals 50	925	CSI Energy	928
Basic 200	965	CSI materials	929
Shenzhen Component Index	399001	CSI optional	931
Small and medium plate refers to	399005	Securities Telecom	936
New index	399100	CSI Public	937
SZSE 100	399330	New energy	941
CSI 100	399903	Securities Agriculture	949
CSI 500	399905	Securities Land Enterprise	953
Oversized plate	43	Securities Consumer	806
Shanghai Stock Exchange	44	Zhongzheng Medicine	399933
Shanghai small cap	45	Securities Finance	399934
SSE SME	46	Securities Information	399935
CSI Emerging	964	CSI Coal	399998
Business growth	958	Pension Industry	399812
CSI bonus	922	Well-off index	901

direction, experts rely on this macro information in addition to their knowledge, wisdom and experience. The primary probability technique, leading indicator method, cross-probability method and Delphi method are some of the traditional approaches. The process of extracting potentially useful, fixed and regular stock prebarium models from a huge quantity of data is known as data mining.

5.1. *Experimental design*

In this paper, the indexes listed on the Shenzhen Stock Exchange and the Shanghai Stock Exchange are selected as the prediction objects, and 42 commonly used indexes are chosen as the experimental data after removing the indexes with a short-listing time and a lot of missing data, including the central market indexes and industry indexes. The list of indexes is as shown in Table 2.

The experimental data are 5-minute K-line data from June 2009 to May 2022. After obtaining the original data, divide the data into samples, calculate the corresponding labels and then extract quantitative and deep features, respectively. The experiment uses the K-line data of the past five days as model input to predict the average rise and fall of the closing price in the next 1 to 4 days, with a total sample size of 3,108. The samples before 2022 are used for model training and testing in Secs. 4.2 and 4.3, and the ratio of training and testing is 7:3. The samples from January to May 2022 are used for the prediction result analysis in Sec. 4.4.

Use the PyTorch deep learning framework to build the deep learning model described in this paper, and then train the model parameters using the Adam method, which is an optimized version of the stochastic gradient descent technique. Because the prediction of the rise and fall of the index in this study falls within the category of regression issues, the loss function that was used throughout the process of training the model for this paper was a mean absolute error (MAE) and a mean square error (MSE). The following is an explanation of how each of the two indicators is calculated:

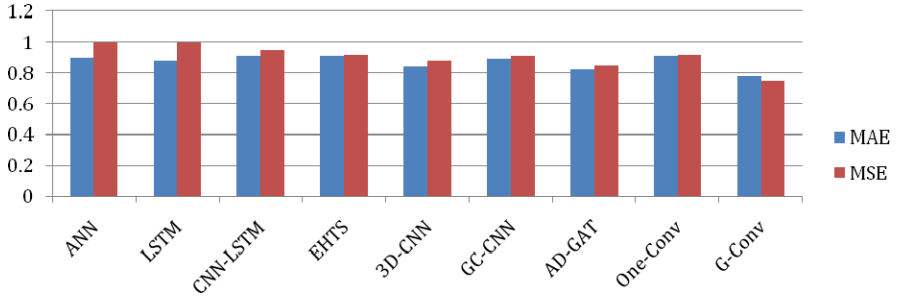
$$\delta_{\text{MAE}} = \frac{1}{n}\sum_{i=1}^{n}|x_i - x_i|, \tag{21}$$

$$\delta_{\text{MSE}} = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_i)^2. \tag{22}$$

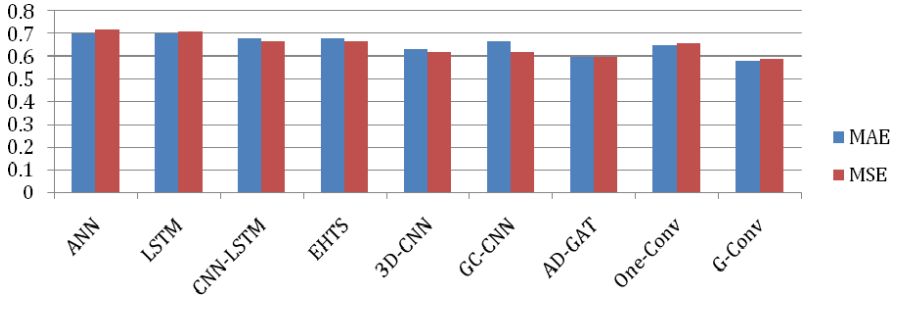
Among them, x_i is the predicted result of the sample x_i, and i is the actual label. After the model training is completed, use the trained model to calculate the prediction results for the test set; finally, the average prediction error of the statistical model on the test set.

5.2. *Comparison of experimental results of different methods*

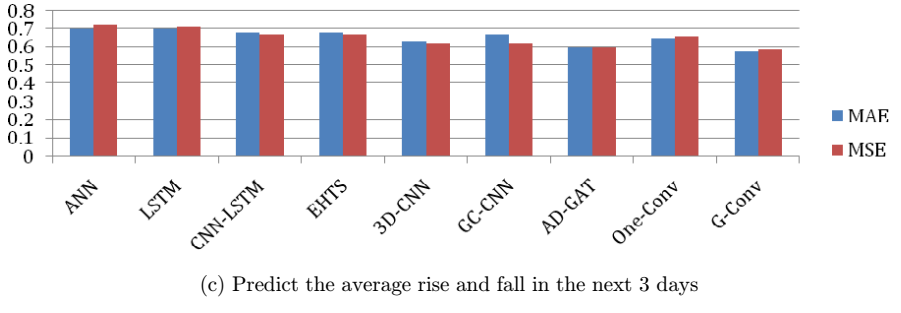
Currently, the predominant techniques employed for the prediction of stocks and indices include fully connected neural networks (ANN [8]), cyclic neural networks (LSTM [12]), convolutional neural networks (3D-CNN [14]), convolution-circular hybrid neural networks (CNN-LSTM [15], EHTS [16]), graph convolutional neural networks (GC-CNN [21], AD-GAT [25]) and others. This study aims to validate the efficacy of the proposed approach by conducting a comparative analysis with the aforementioned methodologies. In the context of the experimental design, when manual feature extraction is necessary, the retrieved features encompass a set of 22-dimensional characteristics, which consist of value, moving average and momentum features [26]. The artificial neural network (ANN) is composed of two hidden layers, each consisting of a single neuron that functions as the output. The input layer of the LSTM consists of 22 neurons, whereas the output of the LSTM unit is connected to a fully connected layer. Additionally, the LSTM's output layer comprises a solitary neuron. The 3D Convolutional Neural Network (3D-CNN) processes a dataset consisting of 240,542 units of three-dimensional (3D) input and produces an output of 142 vectors. The CNN-LSTM model utilizes a combination of input elements, including the Hankel matrix derived via daily factor reconstruction, convolutions from a CNN and an LSTM module, to produce an output. The employed methodology of EHTS involves the utilization of CNNs and long short-term memories to extract location- and time-specific information. Subsequently, multi-layer neurons are employed to transform these extracted characteristics into prediction outputs. The Spearman rank correlation coefficient and gradient descent are employed by GC-CNN and AD-GAT to compute the correlation weight across indices.



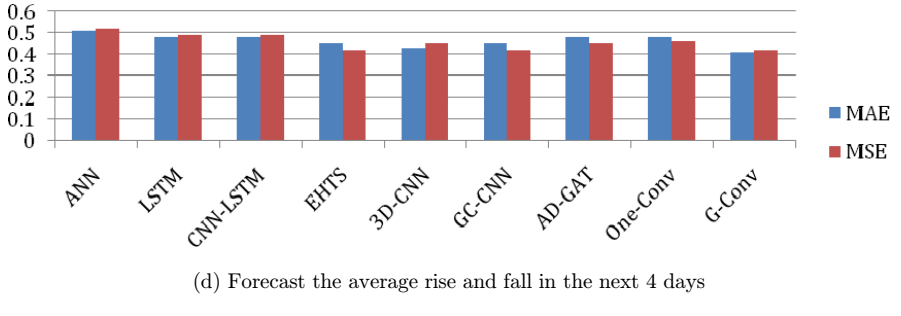
(a) Predict the average rise and fall in the next day



(b) Predict the average rise and fall in the next 2 days



(c) Predict the average rise and fall in the next 3 days



(d) Forecast the average rise and fall in the next 4 days

Fig. 3. Prediction results of different methods.

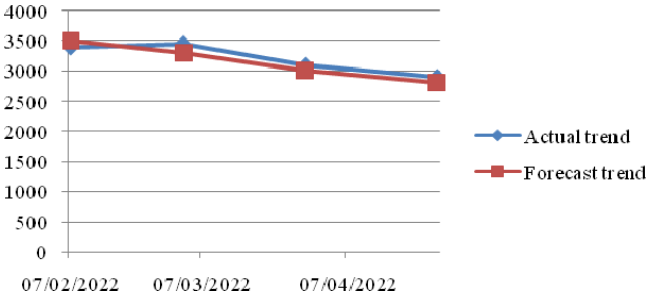
Additionally, CNN and graph attention networks are utilized to integrate information across indices, hence obtaining prediction outcomes.

Four prediction tasks were selected in the experiment, respectively predicting the average rise and fall of the index in the next 1 to 4 days. For each prediction task, the model is trained using the two loss functions of formulas (21) and (22), and the average regression error of the model on the corresponding test set is calculated. Figure 3 shows the average prediction error of 42 indices, where G-Conv is the method proposed in this paper, and One-Conv is the result of the way in this paper after removing the graph convolution.

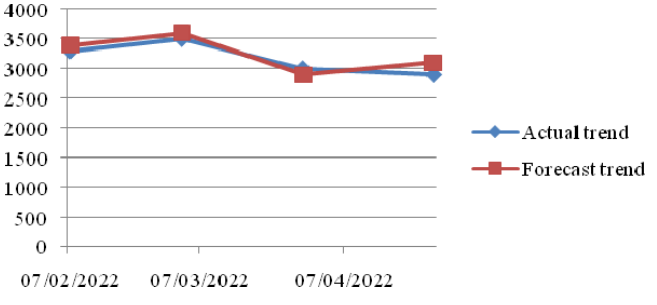
As can be observed in Fig. 3, the average prediction errors of various approaches for various prediction tasks vary widely. Among these methods, the four methods of 3D-CNN, GC-CNN, AD-GAT and G-Conv perform noticeably better than other methods. The following is an explanation of the primary causes for this. When training and making predictions, ANN, LSTM, CNN-LSTM, EHTS and One-Conv, along with other approaches, employ historical data from a single index. Since these methods do not take into account the correlation between indices, their overall performance is subpar. In contrast, the training of the model in 3D-CNN, GC-CNN, AD-GAT and G-Conv is accomplished by jointly utilizing historical data from several indices. The errors of the many indices are averaged out during training to keep the overall error rate as low as possible, which allows for the elimination of the need for a single index. Because overfitting is an issue when training on historical data, these three techniques perform better than others.

When fusing features from different indices, 3D-CNN only considers the associations between neighboring indices during model training and does not consider the associations of a specific index with all other indices. In contrast, GC-CNN, ADGAT and G-Conv fuse the features of different indices by establishing an inter-exponential graph structure and connecting the correlation weights between indices through convolution operations, so these two methods perform better than 3D-Conv. CNN. Comparing GC-CNN and AD-GAT, the performance of the two methods is close but worse than that of GConv. The main reasons are as follows. GC-CNN and AD-GAT use index trend correlation and gradient descent optimization to establish inter-index weights. Both methods are dynamic methods to generate correlation weights, and the calculated graph structure will change with the index trend. In the stock market with high randomness, the graph structure calculated based on historical trend correlation has excellent uncertainty in predicting future trends, and some indexes will perform very poorly. However, the G-Conv in this paper uses the inherent constituent stock data of the index to construct a graph structure, and its graph structure will not change with the trend. Therefore, this method is more suitable for predicting future index trends.

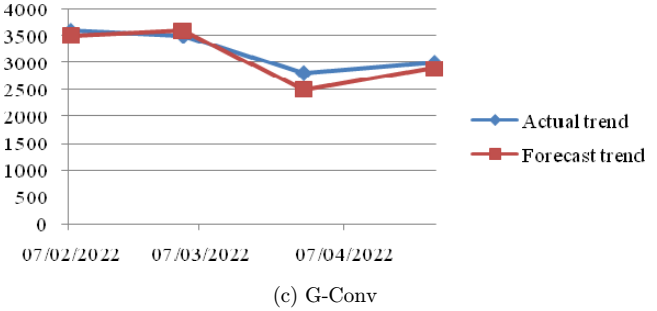
Table 3 lists our method's percentage reduction in forecast error compared to AD-GAT, the current best-performing process, including the average forecast error for all indices and the forecast error for the best and worst-performing indices. From Table 3, it can be concluded that the average forecast error is reduced by about 4.65%,



(a) LSTM



(b) AD-GAT



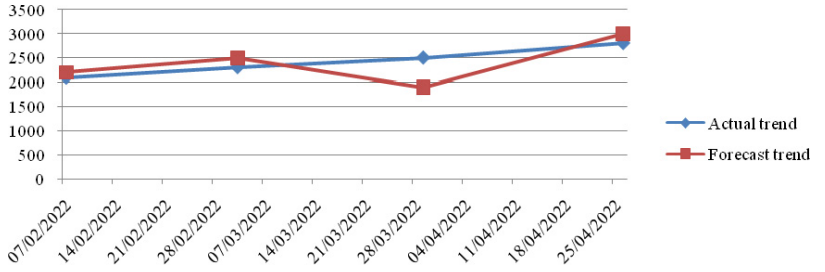
(c) G-Conv

Fig. 4. Analyses of Shanghai Composite Index forecast results.

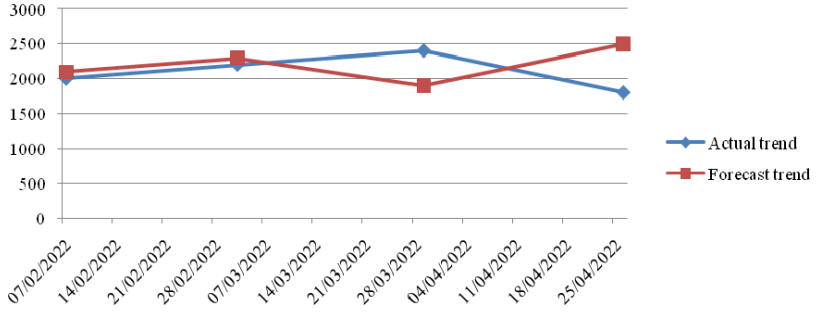
and the forecast errors of the best and worst-performing indices are reduced by 5.26% and 6.12%, respectively. This shows that G-Conv can not only reduce the average prediction error but also have a better effect on improving the poor performance index.

5.3. *Analysis of prediction results*

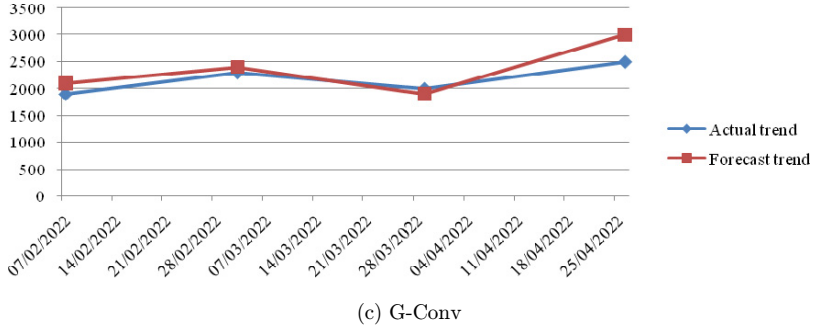
To further analyze the forecasting effects of different methods, Figs. 4 and 5 show the actual and predicted trends of the Shanghai Composite Index and the CSI Energy



(a) LSTMs



(b) AD-GAT



(c) G-Conv

Fig. 5. Analysis of forecast results of CSI Energy Index.

Index from February to April 2022, respectively. During this period, the Shanghai Composite Index generally showed a downward trend, while the CSI Energy Index generally showed an upward trend. Since the results of ANN and LSTM are similar, AD-GAT is representative and performs best among the convolutional methods, so this section only gives the prediction results of the three forms of LSTM, AD-GAT and GConv.

It is apparent from Figs. 4 and 5 that the prediction results of AD-GAT and G-Conv are relatively smooth; that is, the two types of methods are weak in fitting short-term fluctuations, but they can predict the overall trend well. The main

reasons are as follows. The movements of different indexes will have a strong correlation in the short term, and the direction of the index is the result of the joint influence of the trends of other indexes. Accurate stock market models can give investors tools for improved data-based decision making. These models can aid traders in lowering investing risk and choosing the best stocks. Furthermore, using unconventional data, such as previous stock prices and news, is made possible by the development of sophisticated models. These retail and institutional investors would be able to continuously build portfolios with low risk and high return if stock market behavior predictions were more accurate. It is extremely challenging to make predictions in the financial markets due to a variety of elements and uncertainties (such as general economic circumstances, global events, socio-political issues and natural disasters). This has been a key driver behind the application of artificial intelligence approaches to create models for stock market forecasting that are more precise. After performing graph convolution on multiple indices, it can avoid overfitting the model to a single index, making the forecast curve smoother and reducing the forecast error. Further analysis of the difference between the forecast results and the original trend shows that the forecast results of the three methods all have a certain degree of lag, which shows that the lag problem still needs to be solved in index trend forecasting. But in contrast, G-Conv's performance is closer to the actual trend, which shows that the method proposed in this paper significantly improves prediction hysteresis.

6. Conclusion

In the realm of financial investment, one of the most important concerns is determining how accurately one can forecast the movement of a stock market index by making use of previous data. This study offers a method for constructing a graph structure using index constituent stocks as the basis for graph convolution index prediction. The objective of this approach is to address the issue of inconsistency that exists between historical and future dynamic graph structures in the current method of graph convolution index prediction. This method first extracts traditional quantitative features and one-dimensional convolution depth features as a prediction sample feature, then uses the index constituent stock data to construct a graph structure between the indexes, and finally uses graph convolution to achieve deep feature fusion, and maps the convolution results to the index rise and fall. Correlated with the existing approaches, the approach in this paper can effectively reduce the prediction error.

Although the model proposed in this paper can reduce the forecast error of the index, there are still some problems to be solved in the forecast of the stock market index trend. The experimental analysis in this paper shows that the overfitting phenomenon in the model training process is still a common problem in index trend prediction. Improving the model's generalization ability and reducing the impact of overfitting on the prediction of unknown samples are the key issues that need to be solved in the future of exponential projection. Future research can reduce the impact of model overfitting from the following aspects. One is to replace the short-term

forecast with the long-term trend prediction of the index to reduce the impact of the short-term fluctuation of the index; the other is to use experimental methods to artificially generate samples to increase the sample size of the model training; the third is to use techniques such as confrontation training to optimize the model training process, to improve the generalization ability of the model.

ORCID


Ghulam Jillani Ansari https://orcid.org/0000-0002-8985-1383

References

[1] J. Wang, J. He, C. Feng, L. Feng and Y. Li, Stock index prediction and uncertainty analysis using multi-scale nonlinear ensemble paradigm of optimal feature extraction, two-stage deep learning and Gaussian process regression, *Appl. Soft Comput.* **113**(Part A) (2021) 107898, https://doi.org/10.1016/j.asoc.2021.107898.

[2] L. Cagliero, J. Fior and P. Garza, Shortlisting machine learning-based stock trading recommendations using candlestick pattern recognition, *Expert Syst. Appl.* **216** (2023) 119493, https://doi.org/10.1016/j.eswa.2022.119493.

[3] V. M. Ngo, H. H. Nguyen and P. Van Nguyen, Does reinforcement learning outperform deep learning and traditional portfolio optimization models in frontier and developed financial markets?, *Res. Int. Bus. Financ.* **65** (2023) 101936, https://doi.org/10.1016/j.ribaf.2023.101936.

[4] H. Han, C. Yang, B. Jiang, C. Shang, Y. Sun, X. Zhao, D. Xiang, H. Zhang and Y. Shi, Construction of chub mackerel (*Scomber japonicus*) fishing ground prediction model in the northwestern Pacific Ocean based on deep learning and marine environmental variables, *Mar. Pollut. Bull.* **193** (2023) 115158, https://doi.org/10.1016/j.marpolbul.2023.115158.

[5] A. A. Oyedele, A. O. Ajayi, L. O. Oyedele, S. A. Bello and K. O. Jimoh, Performance evaluation of deep learning and boosted trees for cryptocurrency closing price prediction, *Expert Syst. Appl.* **213**(Part C) (2023) 119233, https://doi.org/10.1016/j.eswa.2022.119233.

[6] N. Alam, J. Gao and S. Jones, Corporate failure prediction: An evaluation of deep learning vs discrete hazard models, *J. Int. Financ. Markets, Institut. Money* **75** (2021) 101455, https://doi.org/10.1016/j.intfin.2021.101455.

[7] A. Ali Salamai, Deep learning framework for predictive modeling of crude oil price for sustainable management in oil markets, *Expert Syst. Appl.* **211** (2023) 118658, https://doi.org/10.1016/j.eswa.2022.118658.

[8] Z. Zhou, Intelligent prediction method for power generation based on deep learning and cloud computing in big data networks, *Int. J. Intell. Netw.* **4** (2023) 224–230, https://doi.org/10.1016/j.ijin.2023.08.004.

[9] M. Senthil Murugan and T. Sree Kala, Large-scale data-driven financial risk management & analysis using machine learning strategies, *Meas. Sens.* **27** (2023) 100756, https://doi.org/10.1016/j.measen.2023.100756.

[10] T. Kristóf and M. Virág, EU-27 bank failure prediction with C5.0 decision trees and deep learning neural networks, *Res. Int. Bus. Financ.* **61** (2022) 101644, https://doi.org/10.1016/j.ribaf.2022.101644.

[11] H. Maqsood, I. Mehmood, M. Maqsood, M. Yasir, S. Afzal, F. Aadil, M. Mohamed Selim and K. Muhammad, A local and global event sentiment based efficient stock exchange

forecasting using deep learning, *Int. J. Inf. Manag.* **50** (2020) 432–451, https://doi.org/10.1016/j.ijinfomgt.2019.07.011.

[12] A. Y. Raya-Tapia, F. J. López-Flores and J. M. Ponce-Ortega, Incorporating deep learning predictions to assess the water-energy-food nexus security, *Environ. Sci. Policy* **144** (2023) 99–109, https://doi.org/10.1016/j.envsci.2023.03.010.

[13] S. Ghimire, R. C. Deo, D. Casillas-Pérez and S. Salcedo-Sanz, Boosting solar radiation predictions with global climate models, observational predictors and hybrid deep-machine learning algorithms, *Appl. Energy* **316** (2022) 119063, https://doi.org/10.1016/j.apenergy.2022.119063.

[14] Y. Wang, Z. Luo and J. Luo, Research on predicting the diffusion of toxic heavy gas sulfur dioxide by applying a hybrid deep learning model to real case data, *Sci. Total Environ.* **2023** (2023) 166506, https://doi.org/10.1016/j.scitotenv.2023.166506.

[15] A. Thakkar and K. Chaudhari, Fusion in stock market prediction: A decade survey on the necessity, recent developments, and potential future directions, *Inf. Fusion* **65** (2021) 95–107, https://doi.org/10.1016/j.inffus.2020.08.019.

[16] Y. Ma, Y. Wang, W. Wang and C. Zhang, Portfolios with return and volatility prediction for the energy stock market, *Energy* **270** (2023) 126958, https://doi.org/10.1016/j.energy.2023.126958.

[17] I. Almalis, E. Kouloumpris and I. Vlahavas, Sector-level sentiment analysis with deep learning, *Knowl.-Based Syst.* **258** (2022) 109954, https://doi.org/10.1016/j.knosys.2022.109954.

[18] S. Ahmad, I. Shakeel, S. Mehfuz and J. Ahmad, Deep learning models for cloud, edge, fog, and IoT computing paradigms: Survey, recent advances, and future directions, *Comput. Sci. Rev.* **49** (2023) 100568, https://doi.org/10.1016/j.cosrev.2023.100568.

[19] K. K. Yun, S. W. Yoon and D. Won, Prediction of stock price direction using a hybrid GA-XGBoost algorithm with a three-stage feature engineering process, *Expert Syst. Appl.* **186** (2021) 115716, https://doi.org/10.1016/j.eswa.2021.115716.

[20] W. Zhang, Z. Chen, J. Miao and X. Liu, Research on Graph Neural Network in Stock Market, *Procedia Comput. Sci.* **214** (2022) 786–792, https://doi.org/10.1016/j.procs.2022.11.242.

[21] C. Cosgun, Machine learning for the prediction of evaluation of existing reinforced concrete structures performance against earthquakes, *Structures* **50** (2023) 1994–2003, https://doi.org/10.1016/j.istruc.2023.02.127.

[22] A. Thakkar and K. Chaudhari, A comprehensive survey on deep neural networks for the stock market: The need, challenges, and future directions, *Expert Syst. Appl.* **177** (2021) 114800, https://doi.org/10.1016/j.eswa.2021.114800.

[23] Z. Chen, M. Ma, T. Li, H. Wang and C. Li, Long sequence time-series forecasting with deep learning: A survey, *Inf. Fusion* **97** (2023) 101819, https://doi.org/10.1016/j.inffus.2023.101819.

[24] Y.-C. Chou, C.-T. Chen and S.-H. Huang, Modeling behavior sequence for personalized fund recommendation with graphical deep collaborative filtering, *Expert Syst. Appl.* **192** (2022) 116311, https://doi.org/10.1016/j.eswa.2021.116311.

[25] X. Zhao, J. W. Lai, A. F. W. Ho, N. Liu, M. E. H. Ong and K. H. Cheong, Predicting hospital emergency department visits with deep learning approaches, *Biocybern. Biomed. Eng.* **42**(3) (2022) 1051–1065, https://doi.org/10.1016/j.bbe.2022.07.008.

[26] T. Yin, C. Liu, F. Ding, Z. Feng, B. Yuan and N. Zhang, Graph-based stock correlation and prediction for high-frequency trading systems, *Pattern Recogn.* **122** (2022) 108209, https://doi.org/10.1016/j.patcog.2021.108209.

[27] X. Li, Q. Liu and Y. Wu, Prediction on blockchain virtual currency transaction under long short-term memory model and deep belief network, *Appl. Soft Comput.* **116** (2022) 108349, https://doi.org/10.1016/j.asoc.2021.108349.
[28] M. Ehteram, A. N. Ahmed, Z. S. Khozani and A. El-Shafie, Graph convolutional network – Long short term memory neural network- multilayer perceptron- Gaussian progress regression model: A new deep learning model for predicting ozone concertation, *Atmos. Pollut. Res.* **14**(6) (2023) 101766, https://doi.org/10.1016/j.apr.2023.101766.
[29] L. Yu and M. Li, A case-based reasoning driven ensemble learning paradigm for financial distress prediction with missing data, *Appl. Soft Comput.* **137** (2023) 110163, https://doi.org/10.1016/j.asoc.2023.110163.
[30] F. Xia and R. Chatterjee, Multicategory choice modeling with sparse and high dimensional data: A Bayesian deep learning approach, *Decis. Support Syst.* **157** (2022) 113766, https://doi.org/10.1016/j.dss.2022.113766.
[31] M. M. Navarro, M. N. Young, Y. T. Prasetyo and J. V. Taylar, Stock market optimization amidst the COVID-19 pandemic: Technical analysis, K-means algorithm, and mean-variance model (TAKMV) approach, *Heliyon* **9**(7) (2023) e17577, https://doi.org/10.1016/j.heliyon.2023.e17577.
[32] E. Vecchi, G. Berra, S. Albrecht, P. Gagliardini and I. Horenko, Entropic approximate learning for financial decision-making in the small data regime, *Res. Int. Bus. Financ.* **65** (2023) 101958, https://doi.org/10.1016/j.ribaf.2023.101958.
[33] P.-Y. Wu, T. Johansson, M. Mangold, C. Sandals and K. Mjörnell, Estimating the probability distributions of radioactive concrete in the building stock using Bayesian networks, *Expert Syst. Appl.* **222** (2023) 119812, https://doi.org/10.1016/j.eswa.2023.119812.
[34] S. Ahmed, M. M. Alshater, A. El Ammari and H. Hammami, Artificial intelligence and machine learning in finance: A bibliometric review, *Res. Int. Bus. Financ.* **61** (2022) 101646, https://doi.org/10.1016/j.ribaf.2022.101646.
[35] A. S. Qambar and M. M. M. Al Khalidy, Development of local and global wastewater biochemical oxygen demand real-time prediction models using supervised machine learning algorithms, *Eng. Appl. Artif. Intell.* **118** (2023) 105709, https://doi.org/10.1016/j.engappai.2022.105709.
[36] Z. Yi, F. Salem, M. C. Menon, K. Keung, C. Xi, S. Hultin, M. R. H. Al Rasheed, L. Li, F. Su, Z. Sun, C. Wei, W. Huang, S. Fredericks, Q. Lin, K. Banu, G. Wong, N. M. Rogers, S. Farouk, P. Cravedi, M. Shinde, R. Neal Smith, I. A. Rosales, P. J. O'Connell, R. B. Colvin, B. Murphy and W. Zhang, Deep learning identified pathological abnormalities predictive of graft loss in kidney transplant biopsies, *Kidney Int.* **101**(2) (2022) 288–298, https://doi.org/10.1016/j.kint.2021.09.028.
[37] A. M. Ozbayoglu, M. U. Gudelek and O. B. Sezer, Deep learning for financial applications: A survey, *Appl. Soft Comput.* **93** (2020) 106384, https://doi.org/10.1016/j.asoc.2020.106384.
[38] O. Ogunfowora and H. Najjaran, Reinforcement and deep reinforcement learning-based solutions for machine maintenance planning, scheduling policies, and optimization, *J. Manuf. Syst.* **70** (2023) 244–263, https://doi.org/10.1016/j.jmsy.2023.07.014.
[39] H. N. Bhandari, B. Rimal, N. R. Pokhrel, R. Rimal, K. R. Dahal and R. K. C. Khatri, Predicting stock market index using LSTM, *Mach. Learn. Appl.* **9** (2022) 100320, https://doi.org/10.1016/j.mlwa.2022.100320.
[40] Y. Li and Y. Pan, A novel ensemble deep learning model for stock prediction based on stock prices and news, *Int. J. Data Sci. Anal.* **13** (2022) 139–149.
[41] X. Li, Y. Li, H. Yang, L. Yang and X. Liu, DP-LSTM: differential privacy-inspired LSTM for stock prediction using financial news (2019), arXiv:1912.10806.

[42] W. Zhong and F. Gu, Predicting local protein 3D structures using clustering deep recurrent neural network, *IEEE/ACM Trans. Comput. Biol. Bioinform.* (2020).
[43] T. Mudiyanselage, X. Xiao, X. Zhang and Y. Pan, Deep fuzzy neural networks for biomarker selection for accurate cancer detection, *IEEE Trans. Fuzzy Syst.* (2019).
[44] A. W. Li and G. S. Bastos, Stock Market Forecasting Using Deep Learning and Technical Analysis: A Systematic Review, *IEEE Access* **8** (2020) 185232–185242, doi: 10.1109/ACCESS.2020.3030226.