Information Sciences 643 (2023) 119236

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Stock ranking prediction using a graph aggregation network based on stock price and stock relationship information

Guowei Song [a], Tianlong Zhao [a], Suwei Wang [a], Hua Wang [b], Xuemei Li [a,*]

[a] *School of Software, Shandong University, Jinan 250101, China*
[b] *School of Information and Electrical Engineering, Ludong University, Yantai 264025, China*

A R T I C L E   I N F O

Keywords:
Stock return ranking
Graph-based learning
Stock price relation
Graph convolution

A B S T R A C T

The volatility of stock prices makes it difficult to predict stock price trends correctly. This volatility is affected by many factors, including other stocks related to it. Stock prediction based on graph learning uses various graph neural networks to learn how stocks interact to provide more information. However, they tend to adopt statically defined stock relations based on prior knowledge (such as industry relations and Wiki relations), making it difficult to capture the interplay between stocks over time. In addition, their predictions mostly rely on a single stock relationship, while many types of stock relationships affect the volatility of stock prices in a complex and intertwined manner. A new price similarity relation graph is first constructed using the multi-view stock price similarity to capture dynamic stock relationships. Based on three stock graphs (price similarity, Wiki and industry), we further propose a multi-relational graph attention ranking (MGAR) network. In MGAR, the multi-graph aggregation is achieved by applying adaptive learning mechanisms, thereby forming effective relation embeddings. When combined with the captured price trend embedding, MGAR model gives a ranking list of future returns and chooses K stocks with the best returns to trade so that the return on investment is maximized. Extensive experiments demonstrate that MGAR method outperforms state-of-the-art stock predicting solutions, achieving average returns of 164% and 236% on two real datasets, respectively.

1. Introduction

As an important way of investment in the securities market, stocks have attracted the attention of many investors. Finding an effective stock investment strategy to predict the future price and trend of stocks has always been a difficult problem that needs to be solved urgently in the investment market. As a typical time series data, the trend of stock series is affected by a variety of factors, such as policies, economic development, interest rates, capital flows and investor sentiment, which makes it very challenging to predict stocks correctly.

In recent years, artificial intelligence methods, such as Support Vector Machine (SVM) [1], Extreme Gradient Boosting (XGBoost) [2], Long Short-term Memory (LSTM) [3,4] and Gated Recurrent Unit (GRU) [5,6], are widely used in stock predictions. Some existing stock price prediction studies [7,8] have achieved good prediction results, but there is still much room for improvement.

* Corresponding author at: School of Software, Shandong University, Jinan 250101, China.
E-mail addresses: 202035276@mail.sdu.edu.cn (G. Song), tianlongzhao@mail.sdu.edu.cn (T. Zhao), wangsuwei1@163.com (S. Wang), hwa229@163.com (H. Wang), xmli@sdu.edu.cn (X. Li).

https://doi.org/10.1016/j.ins.2023.119236
Received 21 August 2022; Received in revised form 22 April 2023; Accepted 23 May 2023

Available online 29 May 2023

One of the reasons is that these algorithms do not adequately consider the rich relations among stocks. For example, companies in the same industry are sometimes affected by the same industry development policies and macroeconomic policies, which leads to some similarities in their price movements. Additionally, the stock trends of some companies will also have a certain relation if they are part of upstream and downstream supply and demand chains [9]. Relations between the stocks of these companies will affect the price changes of relevant stocks. Therefore, fully considering the relation between stocks as correctly as possible is helpful for stock forecasting [10]. Many scholars regard stocks and their relationships as nodes and edges in a graph, and use various models based on graph neural network to learn the interaction between stocks, so as to provide more effective information for stock prediction. For example, Graph Neural Network (GNN) [11], Graph Convolution Network (GCN) [12,13] and Graph Attention Network (GAT) [14] can propagate and aggregate information between neighboring nodes according to the degree of association between nodes, and show outstanding performance in many stock forecasting tasks [15]. However, the stock prediction model based on graph learning still has some shortcomings, mainly in the construction and use of graphs.

In terms of relation graph construction, most existing stock relation-based prediction research relies on prior knowledge (such as industry attributes and supply chain relations, etc.) to define the relationship between listed companies and build a relation graph, such as the stock industry relation graph or the Wiki relation graph [16]. Normally, the stock price performance of two companies with an industry relationship usually follows a similar trend; the movement of stock prices of two companies with a Wiki relationship should have a pass-through effect. The prediction research based on stock relation graph takes the historical price embedding and the relationship strength between stocks as the node features and edge weights of the graph, respectively, so that the related stock effects can be aggregated while modeling the temporal evolution of stock price characteristics. However, the stock market is uncertain, policy adjustments, corporate strategic transformations, and corporate responses to social or stock market emergencies may all trigger changes in the companies' relationships. For example, company A and company B have an upstream and downstream relationship in the supply chain. If company A subsequently conducts business related to company B, then their relationship will change from a partnership to a competitor relationship. The existing static construction graph based on prior knowledge can not capture the dynamic relationship changes between enterprises, and even provide a misleading model with incorrect relationships. Therefore, it is necessary for stock prediction to build a graph that can capture the possible changes in the relationship between stocks over time. It has been confirmed in several studies [10,17] that the prices of stocks with related relationships exhibit a significant "lead-lag" or "rise or fall together" pattern. Moreover, the efficient market hypothesis [18] states out that stock prices "fully reflect" all available information. This characteristic of the stock market shows that the price relationship between stocks implies all the relationships that determine the trend of stock price, and because the stock price changes with time, the relationship between stocks based on stock price mining is no longer static. Compared with the static stock relationship, the dynamic stock relationship can better guide the stock ranking prediction.

In the use of relation graphs, algorithms [16,19,20] apply a certain stock relation graph to predict stocks. However, the sparsity of a single relation graph makes it difficult to represent a comprehensive company relationship. At the same time, graph neural networks can only aggregate a unilateral correlation feature of neighboring stocks, and can not learn the complex stock relationship. Although Feng et al. [16] construct the industry relation graph and Wiki relation graph of the National Association of Securities Dealers Automated Quotations (NASDAQ) and New York Stock Exchange (NYSE) stock data sets at the same time, they use the designed Temporal Graph Convolution (TGC) module the predict the two kinds of graphs respectively and don't integrate them effectively. Their prediction results are not satisfactory due to the lack of relationship information caused by the sparsity of the graph. In fact, several types of stock relationships will have an impact on stock price volatility, including industry relationships and supply-demand relations. These effects are complex and intertwined. Therefore, the stock prediction based on graph learning should build a variety of different types of stock relation graphs and integrate multi-dimensional information between stocks or companies, so as to make the captured features more in line with the actual situation. Although the use of multiple relationships will introduce noise, the attention mechanism can mitigate the impact of noise to a certain extent.

The existing research methods based on graph learning have the following problems: 1) The static relation graph can't update the stock relationship that may change over time. 2) The sparseness of a single relation graph makes it impossible to aggregate comprehensive company relationships. To this end, we propose an end-to-end model called Multi-relational Graph Attention Ranking (MGAR) network, which can effectively combine stock relations and stock prices to predict stock rankings. The model is divided into three parts: feature extraction layer, relation extraction layer and prediction layer, as shown in Fig. 1. This paper first constructs a dynamic price similarity relation graph utilizing multi-view stock price series that is used to learn the price relationship between stocks over time, as shown in Fig. 2. The feature extraction layer of the MGAR model extracts the price trend embedding of stocks from the original stock price series as node features for three types of stock graphs (price similarity relation graph, industry relation graph and Wiki relation graph). A characteristic of this layer is that it can reduce data complexity while retaining price trend characteristics of the stock. In the relation extraction layer, the newly designed Multi-relational Graph Attention (MGA) network first uses a two-layer GCN network to train three types of stock relation graphs, then extracts three types of stock relation embeddings, and then uses the adaptive attention mechanism to learn their weights to get the final stock relation embedding. The effective combination of multiple stock relation graphs improves the problem of insufficient information or even errors caused by single and sparse stock relationships. Finally, the stock relation embedding and the price trend embedding are concatenated into the prediction layer to predict the return ranking of the group of stocks. In order to verify the proposed method, we conduct numerous experiments on two stock datasets of NASDAQ and NYSE. The experiments show that the proposed method performs better than existing stock ranking methods. The main contributions of the paper are summarized as follows:

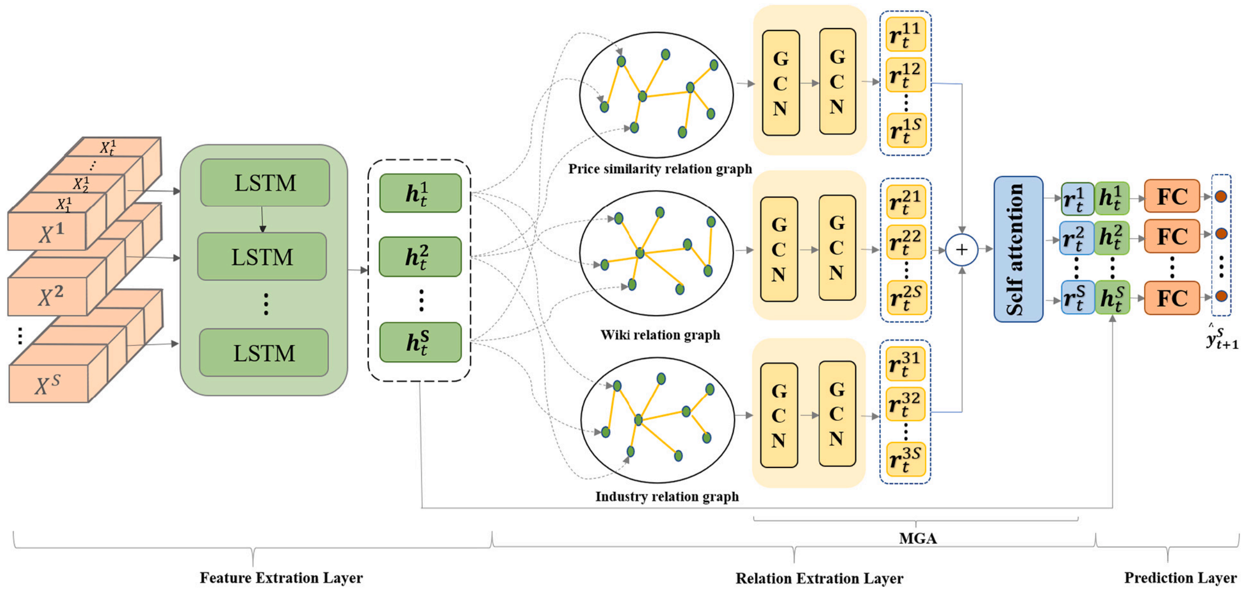


Fig. 1. Stock ranking model based on multi-relational graph attention network.

(1) Using Dynamic Time Warping (DTW) algorithm and neural network to learn price relationships between stocks, creating a price similarity relation graph based on multiple views, and introducing it into stock prediction based on graph learning, which can dynamically describe the relationship between stocks over time.

(2) A multi-relational graph attention module (MGA) is designed to benefit from the complementarity between the three stock relation graphs (stock industry relation, Wiki relation and price similarity relation) in order to produce an effective stock relation embedding that solves the problem of insufficient effective information caused by the sparsity of the stock relation graph.

(3) A multi-relational graph attention ranking (MGAR) network using stock historical price information and various stock relation information is proposed for predicting stock returns ranking in a learning-to-rank manner, and its effectiveness of the model is demonstrated through experiments.

The rest of this paper is organized as follows: Section 2 reviews the work related to stock prediction. Section 3 introduces the construction of stock relation graphs. Section 4 details our proposed MGAR. Section 5 introduces the datasets and shows the experimental results. Finally, the full text is summarized in Section 6.

2. Related work

In this section, we review and evaluate existing stock predicting methods, including classical machine learning-based modeling methods and graph-based learning modeling methods. Furthermore, we explain the problems with the above stock predicting methods, which motivates us to improve existing methods.

2.1. Stock prediction based on classical machine learning

Traditional time series prediction models, such as Autoregressive model (AR) and Autoregressive Integrated Moving Average model (ARIMA), cannot capture the nonlinear relationship of stock price series, so they are difficult to extract the effective features of the stock data. With the rapid development of machine learning, classical models such as XGBoost [2], Random Forest [21], and Prophet [22] are often used as stock prediction models and show good performance. With the rise of deep learning, the Recurrent Neural Network (RNN) and its variants [23,24] have shown good results in stock trend prediction due to their strong ability to process time-series information. Deep learning model has become a popular choice to replace the traditional time-series model for stock trend prediction [25]. For example, Fischer and Krauss [8] use LSTM model to predict stocks for the first time, and explain the advantages of LSTM for stock prediction. Nelson et al. [7] use the LSTM network and the RNN network to construct a model and find that the LSTM network is more suitable for stock prediction. The LSTM network adds the cell state to store long-term memory and capture the long-term dependence in sequences [16], which can effectively solve the problem of vanishing gradient encountered in long-term sequences. However, in the real stock market, stocks are widely interrelated through various relations, and the price change of one stock will be affected by other related stocks [26]. Part of the existing work often only considers the prediction of a single stock, and does not combine relations between stocks to predict the stock price. Akita et al. [27] show that grouping stocks in the same industry can benefit the model prediction. For example, the stock prices of companies that compete may have the opposite trend, while the stock prices of companies that trade with each other may have a similar trend. Stock prices can therefore be better predicted by considering the relationships between stocks.

2.2. Stock prediction based on graph learning

Essentially, the price volatility of stock is caused by its own market signal and the interference of related enterprises [28]. Due to the great achievements of convolution neural networks (CNNs) in the field of computer vision, many scholars begin to apply convolution to graph neural networks and propose GCN [10,29]. In recent years, some researchers have begun to use GCNs-based learning methods to obtain the correlation between stocks, instead of viewing stocks as independent of each other, and jointly predicting the trend of stocks through historical stock prices and stock relations. One node in the graph represents an enterprise, and each edge is established according to the predefined enterprise relation. In the paper of Chen et al. [20], based on information about the investment between companies, the company investment graph of the target company is constructed, and a graph convolutional neural network, which is based on a pipeline model and a joint model, is used to utilize the information of related companies. Sawhney et al. [30] construct a new space-time attention hypergraph network architecture, using hypergraph convolution and attention mechanism to capture the space-time dependence in the stock market. Gao et al. [19] construct a stock relation graph based on industry classification, extract the stock historical features and document features with stock attributes from the stock historical price sequences and stock description documents, and build a graph-based model to get the relation weights between related stocks over time. The studies mentioned above construct a static relation graph based on prior knowledge, which are unable to learn the stock relationship that may change over time. In addition, most of these studies only predict by constructing a single stock relation graph, which makes the extracted stock relation features not comprehensive enough. In fact, different stocks are widely related through various relations (such as industry, supply chain and competition). Some missing features in this single stock graph can be obtained from other types of graphs. Feng et al. [16] propose a stock relation ranking framework, construct two types of stock relations, namely industry relation and Wiki relation, and design a temporal graph convolution model to deal with the relation between stocks, using rank loss to predict stock return ranking. Temporal graph convolution model can handle the effects between different stocks by time-sensitive encoding of stock relations. However, the industry relation graph and Wiki relation graph constructed by this method are sparse, and since each relation graph is still used for prediction separately, the two stock relations are not effectively fused.

In summary, current research on the stock prediction can be divided into two broad categories. The first is based on the prediction of stock price sequences. The stock price time series characteristics are used to predict the trend or price of the stock. Because the trend of a single stock is not only related to its own historical stock price, but also affected by related stocks, it is difficult to learn the complex interactive relation between stocks based on the time series characteristics of prices alone, so the prediction effect is often unsatisfactory. The second is stock prediction based on graph learning. Since graph neural networks have evolved rapidly, more and more scholars have built graphs such as the corporate investment graph, Wiki graph, and industry graph based on the relationships between stocks, and design a neural network framework based on graph learning to predict stock rankings. However, the relationship between stocks may change over time, and the static graph can not capture the relationship changes between enterprises. And the stock relationship is extremely complex, if only one relation graph type is considered, there will be a problem of insufficient effective information, resulting in unstable prediction effects. From a new perspective, a stock price similarity relation graph is built using multiple views to determine the price connections between them. With the newly developed MGA module, multiple stock relation graphs can be combined effectively. Table 1 summarizes the scientific gaps of relevant methods in terms of prediction methods, technology used, datasets, preprocessing method, evaluation measures, advantages and disadvantages of this technology.

3. Stock relation graph

This section first constructs price similarity relation graph based on historical price information of stocks and then constructs Wiki relation graph and industry relation graph based on multi-hot encoding [16] for different kinds of relations between stocks, such as Wiki relation and industry relation, respectively, where the “multi-hot encoding” uses binary vectors of 0 and 1 with a length of N to represent different stock relation types.

As the efficient market hypothesis [18] suggests, in such a market, the current price of a security obviously “fully reflects” all available information. That is, the price of stocks reflects all available information, so the price relationship between stocks should include many complex relationships that influence the stock trend. Therefore, the stock price relationship is one of the most important factors to consider when predicting its ranking. The paper calculates the price similarity between stocks, finds stocks with similar prices, and uses MGA module to mine possible price relationships between stocks (such as leading and lagging, or rising and falling together, etc.).

Price similarity relation graph. In order to better evaluate the similarity of the two stock prices, this paper uses the time series of five views commonly used in stock market price information to describe the stock price. Among the 5 views are stock opening price, closing price, highest price, lowest price and trading volume, hoping to measure the similarity between stocks more comprehensively. DTW [34,35] algorithm is based on the idea of dynamic programming. It is an algorithm to calculate the optimal match of a given two sequences, that is, by setting window constraints, the stock price of a certain period of time of one sequence can match the stock price of a certain period of time of another sequence, so as to find the best match between the two sequences and complete the similarity judgment between them. Since the time series of the five views correspond to similar values with different importance, we use a neural network to learn their weights. To sum up, in this paper, the DTW algorithm is used to calculate the similarity values of the above five natures of any two stocks, which are then put to the full connection layer to obtain the final price similarity relation graph, as shown in Fig. 2.

Table 1
Summarize the related work.

Predict methods	Algorithm	Datasets	Preprocessing	Evaluation measures	Advantages	Disadvantages
Prediction based on stock price series	ARIMA [31]	NIFTY100	Difference	MSE	Effectively combines autocorrelation and white noise of time series data	The data must be stable and cannot capture nonlinear relationships
	XGBoost [32]	CSI 300 Index	Standardization	MSE	The boosted tree model is implemented internally, which can automatically handle missing values	Compared with deep learning models, high-dimensional data features cannot be captured well
	Prophet [33]	YFINANCE	Denoising	MSE	Can effectively detect time series trend change points, seasonality, holidays and other events	Insufficient modeling ability for time series with no obvious periodicity or trend
	LSTM [7]	IBovespa Index	Standardization	MSE	Can capture the long-term dependence in the stock price sequence and solve the gradient disappearance problem encountered in the long-term sequence	Insufficient modeling ability for the dependency relationship between data at different times within the sequence
Stock prediction based on graph learning	Feng et al. [16]	NASDAQ, NYSE	Feature extraction	MSE, MRR, IRR	Construct Wiki and industry relation graphs, and effectively dealt with the impact of different stocks through the time-sensitive encoding of the stock relation	Only a single graph is used for prediction, and the model cannot effectively fuse multiple graphs
	Sawhney et al. [30]	NASDAQ, NYSE	Feature extraction	SR, IRR, NDCG	Spatial hypergraph convolution and attention are used to capture the spatiotemporal dependencies in the stock market	The static relation graph cannot update the stock relationship that may change over time.
	Gao et al. [19]	NASDAQ, NYSE	Feature extraction	MSE, MRR, IRR	Using stock historical price series and stock description documents, the relation weights of related stocks over time can be obtained	A single industry relation cannot better represent the stock relation, and there is insufficient effective information
	Proposed	NASDAQ, NYSE	Feature extraction	MSE, MRR, IRR	A dynamic price similarity relation graph is constructed to learn the price relationship between stocks over time, multiple types of stock relation graphs are constructed, and the attention mechanism is used to make full use of the complementarity between the constructed stock relation graphs.	The increase in the number of graphs leads to a slower running speed of the model

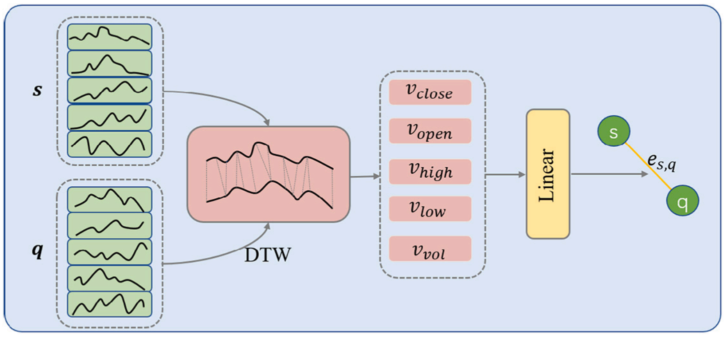


Fig. 2. The weight calculation of the edge between stock s and q in the price similarity relation graph.

Generally speaking, the dimensions of the four price data and the trading volume data of the same stock are typically different, and the prices of different stocks are often quite different as well. If the extracted feature vectors differ in their dimensions and dimension units, the results of the data analysis will be affected. As a result of normalization [36], raw data are scaled to a given range and in the same order of magnitude in order to measure the contribution of each feature under a uniform dimension [37]. Therefore, we first normalize the stock price series of the raw five views: closing price, opening price, highest price, lowest price and trading volume. The normalization method is to map the raw value to $[0, 1]$. Take the stock closing price as an example:

$$p_{\text{close}} = \frac{p - p_{\min}}{p_{\max} - p_{\min}} \tag{1}$$

Where, p_{max} and p_{min} are the maximum and minimum values of the stock closing price series respectively. Similarly, the opening price, the highest price, the lowest price, and the trading volume are calculated according to the above normalization method, respectively, and p_{open}, p_{high}, p_{low}, and p_{vol} can be obtained.

After normalizing the input five stock price sequences respectively, for stock price sequence of two stocks with time lengths of M and N, such as the normalized closing price sequence, $\boldsymbol{s} = \{s_1, s_2, \ldots, s_m\}$ and $\boldsymbol{q} = \{q_1, q_2, \ldots, q_n\}$, a warming path of the normalized closing price sequences of two stocks is expressed as $\boldsymbol{w} = \{w_1, w_2, \ldots, w_k\}, k \in [\max(m, n), m + n - 1]$, let $d(s_i, q_j)$ denote the Euclidean distance between two points s_i and q_j:

$$\begin{gathered} d(w_l) = d(s_i, q_j) = \left|s_i - q_j\right|, \\ i \in [1, m], j \in [1, n], l \in [1, k] \end{gathered} \tag{2}$$

DTW algorithm minimizes the distance between two stock price time series by constructing an optimal warming path. The calculation formula is as follows:

$$\text{DTW}(\boldsymbol{s}, \boldsymbol{q}) = \min \sum_{l=1}^{k} d(w_l) \tag{3}$$

Where, the parameter $k \in [\max(m, n), m + n - 1]$ represents the path length, and the optimal warming path is solved by dynamic programming algorithm:

$$\begin{gathered} \gamma(i, j) = d(s_i, q_j) + \\ \min\left(\gamma(s_i, q_{j-1}), \gamma(s_{i-1}, q_j), \gamma(s_{i-1}, q_{j-1})\right) \end{gathered} \tag{4}$$

Among them, $\gamma(0, 0) = 0$, $\gamma(i, 0) = \gamma(0, j) = \infty$. We use the DTW algorithm to determine similarity values for the two stocks using normalized closing prices:

$$v_{\text{close}} = DTW(\boldsymbol{s}, \boldsymbol{q}) \tag{5}$$

Similarly, v_{open}, v_{high}, v_{low} and v_{vol} can be obtained by calculating the opening price, the highest price, the lowest price and the trading volume respectively according to the above DTW algorithm.

$$e_{s,q} = \psi\left(W_s \boldsymbol{v} + \boldsymbol{b}_s\right) \tag{6}$$

Among them, $\boldsymbol{v} \in \mathrm{R}^D, D = 5$. ψ is Rectified Linear Unit ($Relu$) activation function, W_s and $\boldsymbol{b}_s$ are the weight matrix and bias term, respectively, which are parameters to be learned. $e_{s,q}$ represents the edge weight between stock s and q, the price similarity relation graph $G_{price} \in R^{U \times U}$ is constructed, where U is the number of stocks. In this way, each edge of the graph represents the price similarity of the two stocks corresponding to that edge.

Industry relation graph. NASDAQ and NYSE stocks hierarchies are collected, and industry relations are extracted for each stock pair under the same industry node. Each stock has a sector and industry classification. NASDAQ and NYSE markets offer 112 and

130 industry classifications, respectively, for building industry relations. In both markets, the ratio of stock pairs with at least one type of industry relation is less than 10%, so the industry relation is relatively sparse [16].

Wiki relation graph. The first-order and second-order corporate relations are taken from Wikidata. There are 42 Wiki relation types in NASDAQ market and 32 Wiki relation types in the NYSE market respectively, which are used to build Wiki relation graph. The ratio of at least one stock pair based on Wiki relation type in two markets is 0.21% and 0.30% respectively [16].

4. Model

The MGAR model proposed includes feature extraction layer, relation extraction layer and prediction layer, as shown in Fig. 1, the same features are marked with the same color in the figure, and the color of the module is consistent with the result output by the module. Through the MGA module created in this paper, the industry relation graph, Wiki relation graph, and price similarity relation graph are effectively combined. The effective combination of the three relation graphs solves the problem of incomplete stock relation information caused by the use of a single stock relation, as well as the issue of insufficient effective information caused by the sparsity of the industry and Wiki relations graphs. The feature extraction layer adopts a LSTM model, and its function is based on the input historical stock price data X_t^s (a stock s in the stock group to be predicted) to train to determine the price trend embedding $\boldsymbol{h}_{\mathrm{t}}^s$, which will be the node feature of the stock relation graph; The relation extraction layer uses graph convolution to train the price similarity relation graph, Wiki relation graph and industry relation graph respectively, extracts the stock relation embeddings $\boldsymbol{r}_t^{1s}$, $\boldsymbol{r}_t^{2s}$ and $\boldsymbol{r}_t^{3s}$ in each relation graph and implements a self-attention mechanism for these three stock relation embeddings per stock. In the prediction layer, the price trend embedding $\boldsymbol{h}_{\mathrm{t}}^s$ and the relation embedding $\boldsymbol{r}_t^s$ obtained by the self-attention mechanism are concatenated into the fully connected layer to predict the stock return ranking.

4.1. Feature extraction layer

Stock price data has the characteristics of nonlinearity and high volatility [38], so our model first extracts the effective information from the raw stock price data through the feature extraction layer, and then reduces the complexity of the data while preserving its price trend features of the stock, thus better representing historical stock price information. As LSTM networks can store long-term time information, so as to solve the gradient disappearance and gradient explosion problems in the process of long sequence training, they are widely used to process stock data [8,16,19,39]. Therefore, we encode historical stock price data using LSTM network to obtain effective stock price trend embedding. LSTM consists mainly of three gates: forget gate, input gate and output gate [40]. The forget gate is responsible for filtering the cell state information of the previous neuron, the input gate prepares the cell state information to be updated, and the output gate obtains the hidden state from the cell state. Given the historical price sequence of stock s, $X_t^s = \left\{ \boldsymbol{x}_{t-T+1}^s, \boldsymbol{x}_{t-T+2}^s, \dots, \boldsymbol{x}_t^s \right\}, \boldsymbol{x}_t^s \in \mathrm{R}^K$ is the raw stock price historical sequence features, including closing price, 5, 10, 20, and 30-day price moving averages, representing the daily, weekly, and monthly trends of stock s, T is the length of the time window, $K = 5$. We input X_t^s into the LSTM network. The data transmission process is described as follows:

$$\begin{aligned}
\boldsymbol{z}_t^s &= \tanh\left(W_z\left[X_t^s;\boldsymbol{h}_{t-1}^s\right] + \boldsymbol{b}_z\right)\\
\boldsymbol{i}_t^s &= \sigma\left(W_i\left[X_t^s;\boldsymbol{h}_{t-1}^s\right] + \boldsymbol{b}_i\right)\\
\boldsymbol{g}_t^s &= \sigma\left(W_g\left[X_t^s;\boldsymbol{h}_{t-1}^s\right] + \boldsymbol{b}_g\right)\\
\boldsymbol{c}_t^s &= \boldsymbol{g}_t^s \odot \boldsymbol{c}_{t-1}^s + \boldsymbol{i}_t \odot \boldsymbol{z}_t^s\\
\boldsymbol{o}_t^s &= \sigma\left(W_o\left[X_t^s;\boldsymbol{h}_{t-1}^s\right] + \boldsymbol{b}_o\right)\\
\boldsymbol{h}_t^s &= \boldsymbol{o}_t^s \odot \tanh_t\left(\boldsymbol{c}_t^s\right)
\end{aligned} \tag{7}$$

Among them, $W_z, W_i, W_g, W_o \in R^{U\times K}$ are the parameters to be learned, U is the number of LSTM hidden layer units; $\boldsymbol{b}_z, \boldsymbol{b}_i, \boldsymbol{b}_g, \boldsymbol{b}_o \in \mathrm{R}^U$ are bias terms, $tanh$ and σ are two different activation functions, where σ uses the Leaky Rectified Linear Unit ($LeakyRelu$) activation function. After recursively processing the historical price sequence, the feature extraction layer takes the output of the last hidden layer as the price trend embedding of the stock s, that is:

$$H_t = LSTM\left(\boldsymbol{X}_t\right) \tag{8}$$

Among them, $\boldsymbol{X}_t$ represents the historical sequence characteristics of stock prices of all stocks at time t, $H_t = \left[\boldsymbol{h}_t^1, \boldsymbol{h}_t^2, \dots \boldsymbol{h}_t^S\right] \in R^{S\times U}$ represents the price trend embedding of all stocks, S represents the number of stocks, and U represents the embedding size, that is, the number of hidden units.

4.2. Relation extraction layer (MGA module)

A stock's future price isn't just determined by its historical price, but also by other stocks. Other related stocks' price changes will also affect this stock's price. In the relation extraction layer, an effective relation embedding is obtained by using the MGA module to learn various interactions between stocks based on the historical price and relation information of the stocks.

Graph Convolutional Network (GCN) is an advanced deep learning method that can effectively capture the interdependence between instances in a graph describing complex data. It is widely used in the field of stock prediction [10,41] and has achieved remarkable results. Then, after obtaining the price similarity relation graph G_{price}, the constructed Wiki relation graph G_{wiki} and industry relation graph G_{industry} [16], we introduce a graph convolutional network to deal with stock relations. Through the use of the relation between stocks in the stock relation graph, the features of adjacent stock nodes are aggregated using the convolution operation of the graph. That is, the information of each stock node is distributed to its neighbor nodes, and a new representation is generated for each node, so as to learn the cross influence between them. For the processing of price similarity relation graph, the input of GCN includes two aspects: the price trend embedding H_t (node features of the graph) and the stock price similarity relation graph adjacency matrix A_{price} (the edge weights of the graph). The details are as follows:

$$\widetilde{A}_{\text{price}} = A_{\text{price}} + \lambda I_N \tag{9}$$

$$R_{\text{price}} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\widetilde{A}_{\text{price}}\,\tilde{D}^{-\frac{1}{2}}\sigma\left(\tilde{D}^{-\frac{1}{2}}\widetilde{A}_{\text{price}}\,\tilde{D}^{-\frac{1}{2}}H_t W_r^1\right)W_r^2\right) \tag{10}$$

Among them, A_{price} is the adjacency matrix of the self-connected undirected graph G_{price}, I_N is the identity matrix, λ is an adjustment item, which is set to 1 in this paper. D is the degree matrix, $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$, W_r^l is the parameter to be learned in the l layer of graph convolution, σ is the activation function, this paper uses the $Relu$ activation function. $R_{\text{price}} = \left\{\boldsymbol{r}_t^{11}, \boldsymbol{r}_t^{12}, \dots \boldsymbol{r}_t^{1S}\right\}$ is the price relation embedding output by GCN on the price similarity relation graph, and $\boldsymbol{r}_t^{1s}$ represents the price relation embedding of stock s trained in GCN on the price similarity relation graph at time t. Similarly, the Wiki relation embedding $R_{\text{wiki}} = \left\{\boldsymbol{r}_t^{21}, \boldsymbol{r}_t^{22}, \dots \boldsymbol{r}_t^{2S}\right\}$ and the industry relation embedding $R_{\text{industry}} = \left\{\boldsymbol{r}_t^{31}, \boldsymbol{r}_t^{32}, \dots \boldsymbol{r}_t^{3S}\right\}$ can be obtained. In a one-layer graph convolution operation, each stock node can only aggregate the features of its neighboring stock nodes, while a two-layer graph convolution operation aggregates the node's second-order neighbor features (neighbor's neighbor features). Specifically, the constructed graph undergoes a two-layer graph convolution operation. When another layer is superimposed on the basis of the graph convolution of the first layer, the features of the adjacent nodes of each stock node are repeatedly aggregated this time, and at this point, the neighbor nodes of each stock have already aggregated the features of their neighbor nodes (from the first layer of graph convolution), so that each stock node has aggregated the features of its neighbor nodes and second-order neighbor nodes. The three-layer graph convolution aggregates the third-order neighbor features of stock nodes, making the aggregated information too redundant. Hence, we put the three relational graphs into two layers of graph convolution training respectively, this allows us not only to learn the influence of stocks related to the target stock, but also to mine the influence of stocks with one layer of indirect relation (second-order neighbors) through two-layer graph convolution, so as to make the learned stock relation more comprehensive and to form three kinds of stock relation embeddings with higher accuracy.

Since the feature information for the three stock relational embeddings is derived from the three types of stock relational graphs, they will influence the prediction results of the associated stocks from different aspects. In order to effectively fuse the three stock relation embeddings, we introduce an attention mechanism to allow the model to adaptively capture the importance of the three stock relation embeddings and learn their interaction. An important aspect of the attentional mechanism is the ability to selectively learn the features of the input, to filter the important information from a given number of messages and to focus on these important messages, i.e., to assign different weights to different messages, and the size of the weights represents the importance of the message [42]. Self-attention is a type of attention mechanism where the query(Q), key(K), and value(V) are all sourced from the same input sequence. The obtained three stock relation embeddings are concatenated:

$$R_c = \left[R_{\text{price}}, R_{\text{wiki}}, R_{\text{industry}}\right] \tag{11}$$

Map R_c to query space, key space and value space, and the relevant calculation formula is as follows:

$$\begin{aligned} Q &= W_q R_c = \left[\boldsymbol{q}_1, \boldsymbol{q}_2, \dots, \boldsymbol{q}_n\right] \\ K &= W_k R_c = \left[\boldsymbol{k}_1, \boldsymbol{k}_2, \dots, \boldsymbol{k}_n\right] \\ V &= W_v R_c = \left[\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n\right] \end{aligned} \tag{12}$$

Next, the attention distribution of each position is calculated, and the corresponding results are weighted and summed:

$$s\left(\boldsymbol{q}_i, \boldsymbol{k}_j\right) = \frac{\boldsymbol{q}_i \boldsymbol{k}_j^T}{\sqrt{D_k}} \tag{13}$$

$$\alpha_{ij} = \text{softmax}\left(s\left(\boldsymbol{q}_i, \boldsymbol{k}_j\right)\right) = \frac{\exp\left(s\left(\boldsymbol{q}_i \boldsymbol{k}_j\right)\right)}{\sum_{t=1}^{n}\exp\left(s\left(\boldsymbol{q}_i, \boldsymbol{k}_j\right)\right)} \tag{14}$$

$$\boldsymbol{r}_i = \sum_{j=1}^{n} \alpha_{ij} \cdot \boldsymbol{v}_j \tag{15}$$

Calculate the importance weight for different stock relation embeddings, and use the self-attention mechanism to get the final relation embedding:

$$R = \text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{D_k}}\right)V \tag{16}$$

Among them, W_q, W_k and W_v are the parameters to be learned, $\sqrt{D_k}$ is the scale, α_{ij} is the attention weight, which is used to measure the importance of different graph features. The $Softmax$ operation normalizes the calculation result to a probability distribution, and then multiplies it by the matrix V to get the representation of the sum of the weights. We concatenate the stock relation embeddings obtained by training the three relational graphs, retaining all stock relation feature information, and using self-attention to adaptively assign different weights to the feature information of the concatenated relation embedding to obtain the final relation embeddings $R = \{\boldsymbol{r}_t^1, \boldsymbol{r}_t^2, \dots \boldsymbol{r}_t^S\}$, where $\boldsymbol{r}_t^s$ represents the final relation embedding obtained by the stock s with respect to the relation extraction layer at time t.

4.3. Prediction layer

Prediction layer produces a return ranking list for a given group of stocks. Analysis above suggests that price trend embedding contains the time series characteristics of each stock itself; stock relation embedding aggregates the characteristics of the stock itself and its related stocks. As can be seen, stock relation embedding and price trend embedding both provide specific effective information. The prediction layer connects the price trend embedding $\boldsymbol{h}_t^s$ acquired from feature extraction layer with the relation embedding $\boldsymbol{r}_t^s$ from MGA module. The spliced embedding is weighted and mapped to the sample label through a full connection layer, and then the nonlinear expression ability of the prediction layer is increased through a $LeakyRelu$ activation function. The predicted stock return score of stock s at time $t+1$ $\hat{y}_{t+1}^s$ is as follows:

$$\hat{y}_{t+1}^s = \sigma\left(W_p\left[\boldsymbol{r}_t^s; \boldsymbol{h}_t^s\right] + \boldsymbol{b}_p\right) \tag{17}$$

Among them, W_p and $\boldsymbol{b}_p$ are parameters to be learned, for all S stocks, a ranking list $\hat{\boldsymbol{y}}_{t+1} = \left\{\hat{y}_{t+1}^1, \hat{y}_{t+1}^2, \cdots, \hat{y}_{t+1}^S\right\} \in R^S$ can be obtained, σ is the activation function, this paper uses the $LeakyRelu$ function. We use pointwise regression loss and pairwise ranking-aware loss to optimize our model in order to obtain the return ranking among stocks and select the highest return securities to trade [16]:

$$\begin{aligned} &L\left(\hat{\boldsymbol{y}}_{t+1}, \boldsymbol{y}_{t+1}\right) = \left\|\hat{\boldsymbol{y}}_{t+1} - \boldsymbol{y}_{t+1}\right\|^2 \\ &+ \alpha \sum_i \sum_j \max\left\{0, -\left(\hat{y}_{t+1}^i - \hat{y}_{t+1}^j\right)\left(y_{t+1}^i - y_{t+1}^j\right)\right\} \end{aligned} \tag{18}$$

Among them, $\hat{\boldsymbol{y}}_{t+1}$ and $\boldsymbol{y}_{t+1}$ represent the predicted ranking list and the real ranking list for day $t+1$, respectively. $\hat{y}_{t+1}^i$ and y_{t+1}^i represent the predicted return and the real return of the stock i on day $t+1$, respectively. α is a hyperparameter that tunes the two loss terms. The first loss term is a calculation of the difference between the predicted return and the true return so that all predicted stock returns have a high level of accuracy. The second loss term is the pairwise ranking-aware loss, which calculates the relative ranking error for each stock pair, i.e. optimizes the predicted returns to have the same relative order as the true value, so that the predicted return ranking list has an accurate ranking order.

The algorithm flow of MGAR model is shown in Algorithm 1.

Algorithm 1 Model training process.

```
Input: $X_t = [\boldsymbol{x}_{t-T+1}, \boldsymbol{x}_{t-T+2}, \dots, \boldsymbol{x}_t]$ represents historical price series, $\boldsymbol{x}_t \in \mathbb{R}^N$ represents N price nature at day $t$.
Output: $\hat{\boldsymbol{y}}_{t+1} = [\hat{y}_{t+1}^1, \hat{y}_{t+1}^2, \dots, \hat{y}_{t+1}^S]$, $\hat{y}_{t+1}^j (j = 1, 2, \dots, S)$ represents the ranking score of stock $j$ at day $t+1$, the number of stocks is represented by $S$.
1:Initialize parameters of LSTM and GCN
2:for $j = 1$ to $S$ do
3:  $\boldsymbol{h}_t^j = LSTM\left(X_t^j\right)$, price trend embedding of the $j$th stock.
4:  Get $\boldsymbol{r}_t^{1j}$: The price relation embedding of the $j$th stock obtained by GCN.
5:  Get $\boldsymbol{r}_t^{2j}$: The Wiki relation embedding of the $j$th stock obtained by GCN.
6:  Get $\boldsymbol{r}_t^{3j}$: The industry relation embedding of the $j$th stock obtained by GCN.
7:  $\boldsymbol{r}_t^j = Attention([\boldsymbol{r}_t^{1j}, \boldsymbol{r}_t^{2j}, \boldsymbol{r}_t^{3j}])$
8:  $\hat{y}_{t+1}^j = MLP([\boldsymbol{r}_t^j, \boldsymbol{h}_t^j])$
9:end for
10:Get $\hat{\boldsymbol{y}}_{t+1} = [\hat{y}_{t+1}^1, \hat{y}_{t+1}^2, \dots, \hat{y}_{t+1}^S]$.
11:Calculate the loss function according to eq (18).
12:Update parameters of LSTM and GCN by gradient descent.
```

5. Experiments and analysis

In this section, we introduce the dataset, evaluation metrics, comparison methods, and experimental parameters. To assess the performance of the model comprehensively, we compare the MGAR model to several current state-of-the-art models on the data sets [16] of the NASDAQ and NYSE markets. Then, the ablation experiment confirms the effectiveness of the designed different modules. We test the results of graph convolution of different layers in MGA module graph, and verify the effectiveness of the connection between price trend embedding and relation embedding in the prediction layer. After that, the changes in returns of the MGAR model adopting different trading strategies are examined. We also explore the impact of the setting of hyperparameters on the model.

Table 2
Division of training set, validation set and test set.

Market	#Stocks	Training set 01/02/2013 12/31/2015	Validation set 01/04/2016 12/30/2016	Testing set 01/03/2017 12/08/2017
NASDAQ	1026	756	252	237
NYSE	1737	756	252	237

5.1. Dataset

In this subsection, we detail stock historical price data and stock relation data in the two markets.

Stock historical price data. We use stocks in NASDAQ and NYSE markets. The stock historical sequence is between January 2, 2013 and December 8, 2017, including 1026 and 1737 stocks, respectively. We set the prediction frequency to the daily level, that is, our goal is to predict the stock return ranking list of the next trading day based on the historical trading data of the past t days. Since the prediction is for return ranking, set the stock's ground-truth ranking score to 1-day return. In order to obtain stock price trends in different intervals such as weekly, monthly, etc., the closing price, 5-day, 10-day, 20-day and 30-day moving averages are used as features. Then, the sequence data are trained according to three intervals, the training set (January 2, 2013 to December 31, 2015), the validation set (January 4, 2016 to December 30, 2016) and the test set (January 3, 2017 to December 8, 2017), as shown in Table 2, the days of training, validation and test sets are 756, 252 and 237 days respectively.

Stock relation data. For the industry relation, the NASDAQ and NYSE markets have 112 and 130 industry classifications, respectively, which are used to construct industry relations; For the Wiki relation, the NASDAQ and NYSE markets have 42 and 32 kinds of company relations between the stock pairs, respectively, which are used to construct Wiki relation graph [16]. On the training set of the NASDAQ and NYSE markets, price similarity relation graph is constructed using the five view time series data of opening price, closing price, highest price, lowest price and trading volume.

5.2. Experimental setup

In this subsection, we will introduce the trading strategy of the model, the evaluation metrics of the model, the comparison model and the parameter settings of the model training in detail.

5.2.1. Evaluation metrics

In order to accurately predict the return of stocks and rank the return of stocks, we use metrics such as Mean Squared Error (MSE), Mean Reciprocal Rank (MRR) and Investment Return Ratio (IRR) to evaluate the performance of the model in the test set interval (January 3, 2017 to December 8, 2017). Considering that both risk and return need to be measured in real investment, we introduce Sharpe Ratio (SR) and Maximum Drawdown (MDD) to compare the risk control ability of the model. The following is a detailed description of the above evaluation metrics.

-MSE: The truth value is subtracted from the predicted value and then squared and averaged. It is used to evaluate the difference between the ground truth and predicted value of all stocks for each trading day.

$$MSE = \frac{1}{m}\sum_{i=1}^{m}\left(y_i - \hat{y}_i\right)^2 \tag{19}$$

-MRR: This is used to evaluate the accuracy of stock rankings by comparing the ranking of correct results in search results. We query the predicted top1, which is ranked at position K in the real result k_i, that is, $rank$ is k_i, then the score is counted as $\frac{1}{k_i}$, the average score of all queries is MRR.

$$MRR = \frac{1}{N}\sum_{i}^{n}\frac{1}{k_i} \tag{20}$$

-IRR: It reflects the return on investment and is the most important metric, which is calculated by adding the return of selected stocks on each test day, R_i represents the daily return.

$$IRR = P_0\sum_{i}^{n} R_i \tag{21}$$

-SR: Investors need to consider risks while obtaining benefits. The Sharpe ratio is an indicator that considers both returns and risks.

$$SR = \frac{E\left(R_P\right) - R_f}{\sigma_P} \tag{22}$$

Where $E(R_P)$ is the expected rate of return on a portfolio, R_f represents the interest rate without any risk, and σ_P is the standard deviation of returns.

-MDD: It is the range between the highest and lowest point of stock returns during a given trading period. It is used to measure the maximum loss experienced by investors during the investment period.

$$MDD = \max\left(\frac{D_i - D_j}{D_i}\right), \quad j > i \tag{23}$$

Where D_i is the net product value on day i.

5.2.2. *Baselines*

We select some of the most advanced models in recent years to compare with the MGAR model, including LSTM, Rank_LSTM, four variations of Relational Stock Ranking (RSR), Spatio Temporal Hypergraph Attention Network for Stock Ranking (STHAN-SR) and Time-aware Relational Attention Network (TRAN), and here is a brief description of them.

-LSTM: It is the vanilla LSTM [43], which is trained based on the price features constructed by historical sequence data to obtain sequential embedding, and then a full connection layer is used to predict the rate of return.

-Rank_LSTM: It uses the above LSTM and adopts the method of ranking prediction. That is, the MGAR model removes the relation extraction layer [19].

-RSR_E(industry): It is a graph-based stock ranking prediction method, based on Rank LSTM, the TGC module is added and explicit modeling method is adopted to mine relationship characteristics from the constructed industry relation graph [16].

-RSR_E(wiki): Similar to RSR_E(industry), it only uses the constructed Wiki relation graph to train and mine the relation features [16].

-RSR_I(industry): It is a variant of RSR_E, which uses implicit modeling instead of explicit modeling in the TGC module to mine relational features from the constructed industry graph [16].

-RSR_I(wiki): Similar to RSR_I(industry), it only uses the constructed Wiki relation graph to train and mine the relation features [16].

-STHAN-SR: It uses the relationship between stocks to build a hypergraph, and uses the designed spatial hypergraph convolution module and attention mechanism to learn the spatio-temporal dependence in stocks [30].

-TRAN: It is an improved model of RSR model. By using the interaction between the stock price series and the stock description document, the time-aware relationship attention is used to obtain the correlation intensity between stocks over time [19].

5.2.3. *Experimental parameter setting*

We normalize the historical sequence features for each stock to [0,1] to avoid the problem of using different units when calculating each feature [44]. We implement the proposed model with TensorFlow, use Adam to optimize, and set the learning rate as 0.0001. The number of GCN network hidden layer units in NASDAQ and NYSE is 30 and 64 respectively. In NASDAQ market, the sequential input length in LSTM is 16, the number of units in the hidden layer is 64, and the α in the objective function is set to 0.1; In NYSE market, the sequential input length in LSTM is 8, the number of hidden layer units is 32, and the α in the objective function is set to 10. We run the model and other methods 10 times and report their average test performance. The parameter setting of the comparative experiment adopts the same parameters as the original author [16,19].

5.2.4. *The hypothesis and limitations of the develop method*

Since the stock market is affected by many factors, in the real stock trading market, if the top K stocks are predicted on the day t, it is difficult to buy or sell the top K stocks at the closing price on the day $t+1$. Thus, we assume the following: investors will be able to buy the top K stocks at the closing price the same day (day t) after obtaining the ranking of the predicted returns of all stocks by utilizing the model, and then sell them at the closing price the following trading day (day $t+1$), repeating the process. IRR is calculated based on the following assumptions: (1) Investors spend the same amount of money on each trading day. (2) Investors can buy at the closing price on day t and sell at the closing price on day $t+1$. (3) Transaction costs are not taken into account.

5.3. *Experimental results*

For the purpose of proving the performance of the proposed MGAR model, we conduct a series of experiments, including an ablation experiment, a comparative experiment, MGA module of different graph convolution layers, validation of price trend embedding splicing, a model evaluation of different strategies, as well as a parameter sensitivity analysis.

5.3.1. *Comparative experiment*

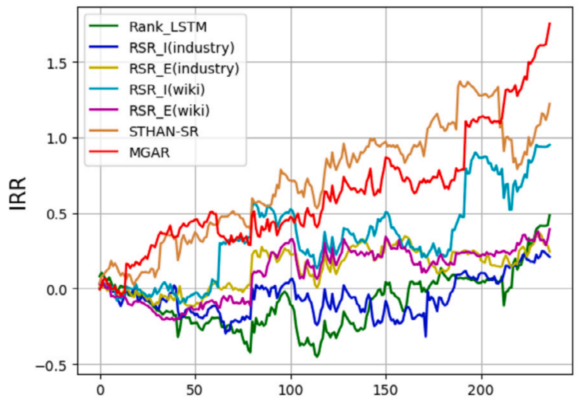
Aiming at the data sets of NASDAQ and NYSE markets, we compare all the above baseline models on MSE, MRR and IRR. MGAR model achieves the best results on multiple indicators, and the experimental results are shown in Table 3.

In both markets, Rank_LSTM outperforms LSTM by a large margin in both markets, especially in terms of return, since the optimization goal set by the rank-based model is no longer just MSE, but also optimizes for return ranking. For the RSR model, in terms of considering industry relations, whether based on explicit modeling (RSR_E (industry)) or implicit modeling (RSR_I (wiki)), the prediction effect is not obvious in the NASDAQ market, but it shows better results in the NYSE market. This may be because the industry relationship reflects the longer-term correlation between stocks. Compared with the NYSE market, the NASDAQ market fluctuates greatly, and it is affected by short-term factors [16]. In terms of considering Wiki relations, in both markets, the two modeling methods of RSR (RSR_E(wiki) and RSR_I(wiki)) are better than Rank_LSTM, which verifies that modeling considering stock

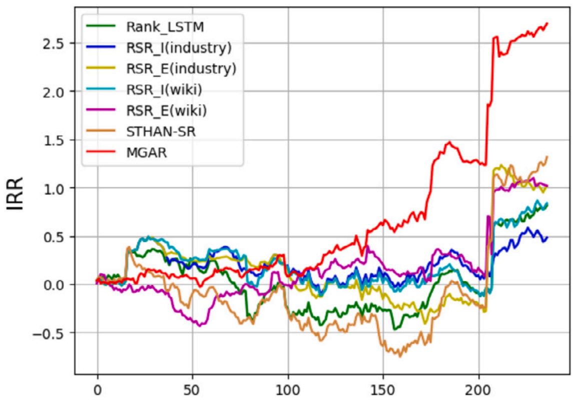
Table 3
Experimental results of different models.

	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
LSTM	3.81e-4 ± 2.20e-6	3.64e-2 ± 1.04e-2	0.13 ± 0.62	2.31e-4 ± 1.43e-6	2.75e-2 ± 1.09e-2	-0.90 ± 0.73
Rank_LSTM	3.79e-4 ± 1.11e-6	4.17e-2 ± 7.50e-3	0.68 ± 0.60	2.28e-4 ± 1.16e-6	3.79e-2 ± 8.82e-3	0.56 ± 0.68
RSR_E(industry)	3.82e-4 ± 2.69e-6	3.16e-2 ± 3.45e-3	0.20 ± 0.22	2.29e-4 ± 2.77e-6	4.28e-2 ± 6.18e-3	1.00 ± 0.58
RSR_E(wiki)	3.80e-4 ± 7.20e-7	3.94e-2 ± 8.15e-3	0.81 ± 0.85	2.29e-4 ± 2.77e-6	4.28e-2 ± 6.18e-3	0.96 ± 0.47
RSR_I(industry)	3.80e-4 ± 7.90e-7	3.17e-2 ± 5.09e-3	0.23 ± 0.27	2.26e-4 ± 5.30e-7	4.51e-2 ± 2.41e-3	1.06 ± 0.27
RSR_I(wiki)	3.79e-4 ± 6.60e-7	4.09e-2 ± 5.18e-3	1.19 ± 0.55	2.26e-4 ± 1.37e-6	4.58e-2 ± 5.55e-3	0.79 ± 0.34
STHAN-SR	1.45e-3 ± 1.35e-4	**4.83e-2 ± 4.78e-3**	1.10 ± 0.53	2.93e-4 ± 2.13e-6	4.35e-2 ± 7.46e-5	0.61 ± 0.81
TRAN	3.79e-4 ± 3.90e-7	3.81e-2 ± 4.37e-3	0.92 ± 0.25	2.26e-4 ± 2.30e-7	4.91e-2 ± 4.82e-3	1.38 ± 0.35
MGAR	**3.77e-4 ± 5.20e-6**	4.18e-2 ± 6.42e-3	**1.64 ± 0.35**	2.26e-4 ± 1.31e-7	**4.98e-2 ± 3.41e-3**	**2.36 ± 0.24**

Bold values are the best results for the evaluation metrics in the table, the results of LSTM, Rank_LSTM, RSR_E(industry), RSR_I(industry), RSR_E(wiki) and RSR_I(wiki) come from Feng et al. [16], the results of TRAN come from Gao et al. [19].



(a) NASDAQ



(b) NYSE

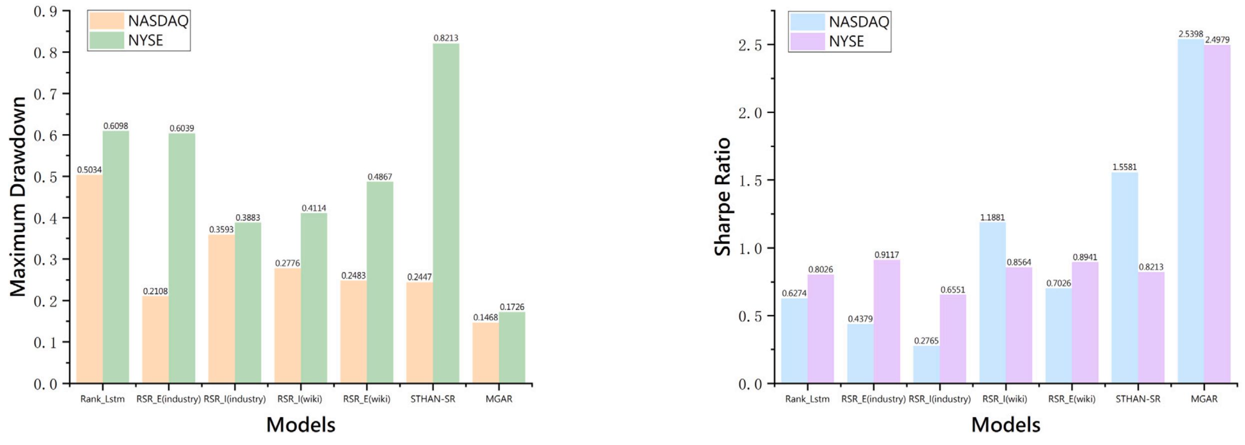
Fig. 3. IRR performance of different models in NASDAQ and NYSE.

relations is helpful for stock ranking prediction. In the NASDAQ market, the STHAN-SR model achieves the best results in the MRR index compared with all the comparison methods, which may be due to the fact that the hypergraph establishes a collective high-level relationship among multiple stocks and alleviates the ranking fluctuations caused by the volatile NASDAQ stock market. Because of its static graph based on a priori knowledge, it can not capture the possible relationship changes in the stock market, so its effect on IRR indicators is not significant. Based on the stock industry graph, the TRAN model uses the constructed stock description document and uses the constructed Time-aware Relational Attention (TRA) unit to learn the relationship strength between stocks. Due to the supplement of time-aware stock description information, better results are achieved compared with the above model. However, because the model still uses an industry graph based on prior knowledge and the stock relationship is single, it can not fully describe the complex interaction between stocks, so the prediction performance of the model is still not ideal.

When compared to the current state-of-the-art ranking-based model TRAN, the MGAR model improves IRR by 78.3% and 71.0%, respectively, in NASDAQ and NYSE markets. In terms of MRR, in the NASDAQ market, we can see that of all the comparison methods, our model is only lower than STHANSR, this may be because the hypergraph builds a collective high-level relationship between multiple stocks, alleviating the volatility of the rankings brought about by stocks in the volatile NASDAQ market. In the NYSE market, our model has the best performance. This shows that the MGAR model can more accurately predict the return ranking, which is the main reason why it can achieve higher returns. In terms of MSE, we can see that the MSE difference between each model is small, but our model is 0.02 lower than the optimal model for NASDAQ. In the NYSE market, it is the same as the optimal model, but it shows good stability.

In stock recommendation model, the most important evaluation metric is the return, which is also the metric that investors pay the most attention to. Fig. 3 shows the IRR curves of each model in the test set for 237 days, it can be seen that the IRR curves of each model are very unstable and fluctuate greatly, because the absolute values of the daily returns range from 0 to 0.98, and once the predicted ranking changes, it will lead to large deviations in the returns [16].

In addition to paying attention to returns, investors need to consider risks as well. In Fig. 4, we calculate the Sharpe ratio and maximum drawdown of each model to evaluate the risk control ability of the model. In NASDAQ and NYSE markets, both metrics achieve the best results out of all models. The maximum drawdowns of the MGAR model are 14.7% and 17.3%, respectively, which are 6.4% and 21.5% lower than the best results from other models (21.1% of RSR_E(industry) and 38.8% of RSR_I(industry)), respectively; MGAR's Sharpe ratios are 2.5398 and 2.4979, which are 63.0% and 174.0% higher than the best results of other models



(a) Maximum Drawdown (b) Sharpe Ratio

Fig. 4. Maximum drawdown and Sharpe ratio performance of each model.

Table 4
Results of three MGAR variant models on two datasets.

	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
wo/g_price&att	3.79e-4 ± 6.60e-7	3.16e-2 ± 3.12e-3	0.58 ± 0.39	2.26e-4 ± 2.39e-6	3.94e-2 ± 5.55e-3	0.95 ± 0.42
wo/g_price	3.79e-4 ± 4.23e-7	3.84e-2 ± 2.32e-3	1.06 ± 0.37	2.26e-4 ± 1.37e-6	4.53e-2 ± 5.51e-3	1.41 ± 0.39
wo/att	3.78e-4 ± 4.93e-7	3.66e-2 ± 5.18e-3	1.42 ± 0.43	2.26e-4 ± 3.36e-7	4.46e-2 ± 4.73e-3	1.69 ± 0.34
MGAR	**3.77e-4 ± 5.20e-6**	**4.18e-2 ± 6.42e-3**	**1.64 ± 0.35**	2.26e-4 ± 1.31e-7	**4.98e-2 ± 3.41e-3**	**2.36 ± 0.24**

Bold values are the best results for the evaluation metrics in the table.

(1.5581 of STHAN-SR and 0.9117 of RSR_E(industry)). This is because a comprehensive analysis of stock relationships from multiple perspectives can effectively uncover a more comprehensive graph of stock relationship. If the relevant stock price of a stock fluctuates greatly, our model may be able to learn about relevant information and avoid some possible risks.

5.3.2. Ablation experiment

In order to further evaluate the effectiveness of the price similarity relation graph proposed in Section 3 as well as the self-attention mechanism of the MGA module proposed in Section 4, we transform MGAR into the following three models and carry out experiments on NASDAQ and NYSE stock markets.

1.wo/g_price&att: MGAR removes price similarity relation graph and self-attention mechanism.

2.wo/g_price: MGAR removes price similarity relation graph.

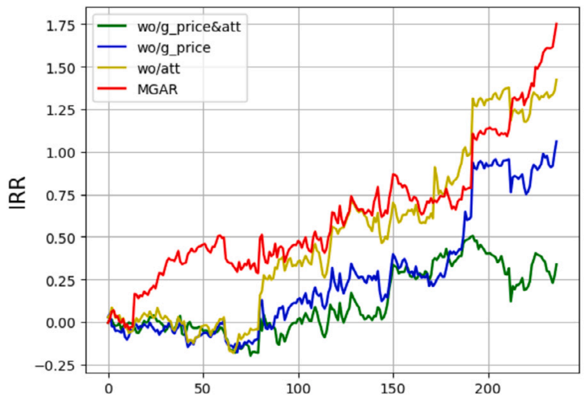
3.wo/att: MGAR removes the self-attention mechanism.

Using these methods, 10 experiments are conducted on NASDAQ and NYSE markets, and the average performance is recorded in Table 4.

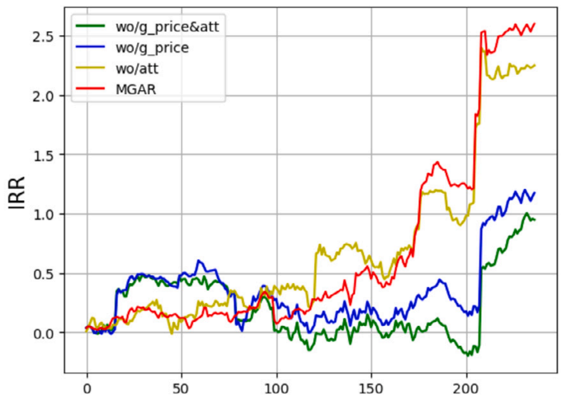
As shown in Table 4, in NASDAQ and NYSE markets, the IRR of the wo/g_price&att model are only 0.58 and 0.95, and the IRR of the wo/att model are 1.42 and 1.69, respectively, which are 144.8% and 77.9% higher than the wo/g_price&att model. MRR also increases to varying degrees, demonstrating that the self-attention module can learn the interaction of different stock relations and achieve effective stock relation embedding. Compared to the wo/g_price&att model, the IRR of the wo/g_price model increases by 82.8% and 48.4% respectively in the two markets, and the MRR also improves. This indicates that the stock price similarity relation graph constructed by considering the similarity between the historical price information of multiple views complements the effective relation information feature, that is, the model can mine the price relation between stocks from the stock price similarity relation graph. On the two datasets, the MSE, MRR and IRR of the MGAR model are all achieve the best, indicating that the combination of the constructed stock price similarity relation graph and the self-attention module can produce better results. Fig. 5 illustrates the IRR results of the three MGAR variant models on the test set for the two datasets. As can be seen, the MGAR model performs better over each time period in the test sets.

5.3.3. Selection of layers in the module

The number of graph convolutional layers in the MGA module is set from 1 to 4, and the results are shown in Table 5. As shown in Table 5, the MSE values are very similar when the layer is set to 1 to 4. When the number of layers increases, both MRR and IRR increase at first, then decrease on both data sets. Both MRR and IRR are best when layer is set to 2. One layer of graph convolution only aggregates the characteristics of its neighbor nodes. In fact, the stock may also be indirectly affected by second-order neighbors



(a) NASDAQ



(b) NYSE

Fig. 5. IRR performance of three MGAR variant models on two datasets.

Table 5
Results of GCN layers 1 to 4 on two datasets.

	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
One layer	3.77e-4 ± 8.17e-7	3.97e-2 ± 3.63e-3	1.27 ± 0.26	2.26e-4 ± 6.38e-7	4.92e-2 ± 4.21e-3	1.44 ± 0.45
Two layers	**3.77e-4 ± 5.20e-6**	**4.18e-2 ± 6.42e-3**	**1.64 ± 0.35**	2.26e-4 ± 1.31e-7	**4.98e-2 ± 3.41e-3**	**2.36 ± 0.24**
Three layers	3.78e-4 ± 1.19e-7	4.06e-2 ± 2.14e-3	1.31 ± 0.37	2.26e-4 ± 7.29e-7	4.71e-2 ± 2.11e-3	1.34 ± 0.42
Four layers	3.78e-4 ± 1.07e-7	3.83e-2 ± 5.19e-3	1.02 ± 0.43	2.27e-4 ± 1.88e-6	4.59e-2 ± 4.83e-3	1.17 ± 0.36

Bold values are the best results for the evaluation metrics in the table.

Table 6
The results of the prediction layer using different input methods on the two datasets.

	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
MGAR_R	3.81e-4 ± 5.68e-6	3.76e-2 ± 4.78e-3	1.24 ± 0.51	2.29e-4 ± 2.46e-6	4.52 e-2 ± 3.61e-3	1.51 ± 0.54
MGAR	**3.77e-4 ± 5.20e-6**	**4.18e-2 ± 6.42e-3**	**1.64 ± 0.35**	**2.26e-4 ± 1.31e-7**	**4.98e-2 ± 3.41e-3**	**2.36 ± 0.24**

Bold values are the best results for the evaluation metrics in the table.

(neighbors' neighbors). From the third layer to the fourth layer, the predicted results gradually become worse, because with the increase of layers, the aggregate neighbor node information becomes redundant, making it difficult to model the stock relation and learn the effective relationship characteristics. Minor ranking changes, however, can have a major impact on IRR. Therefore, when the layer is set to 2, the neighbor information can be better aggregated, and the model performs best.

5.3.4. Validation of price trend embedding splicing

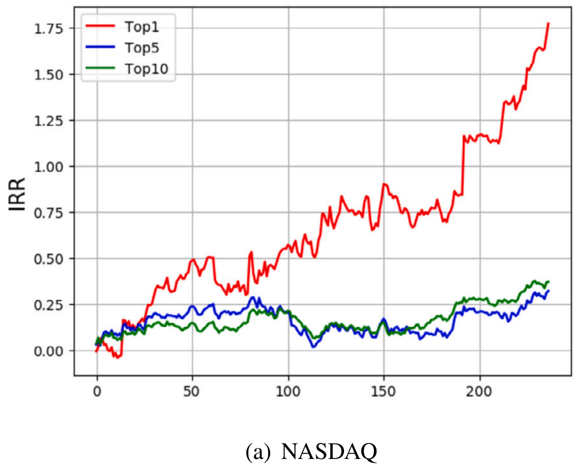
In order to verify the effectiveness of splicing price trend embedding and relation embedding in the prediction layer, we conduct related experiments. In the prediction layer, MGAR_R only uses relation embedding to predict the return score of stocks, and Table 6 shows the results.

Table 6 shows the results of the prediction layer using different input methods on the two datasets. In comparison with the MGAR model, all the indicators of the MGAR_R model have decreased. Among them, MSE's performance is poor, as the price trend embedding contains the characteristics of stock price, which are very important for predicting the accuracy of stock return score. Relation embedding contains information about the stock and its related stocks, allowing the model to learn a more accurate ranking. The low accuracy of the stock return score, however, affects the ranking accuracy, while reducing the IRR. This shows that considering stock prices and concatenating price trend embedding and relation embedding will improve the accuracy of the model.

5.3.5. Model evaluation under different strategies

In real life, in order to avoid risks, investors often adopt a portfolio strategy, that is, buy multiple stocks for trading. For example, Top5 is to select the top-5 stocks with predicted ranking scores and buy them evenly, and calculate the average IRR of the 5 stocks. Therefore, we test the MGAR model on two datasets to select top-1, top-5, and top-10 stocks for backtesting. The IRR curves for the three investment strategies are in Fig. 6, we can observe the following points:

In both markets, Top1 has a higher IRR value than Top5 and Top10. In NYSE market, the returns of each time period of the test set are basically Top1 $>$ Top5 $>$ Top10. This is because our task is to rank according to the return rate of stocks. A higher ranking



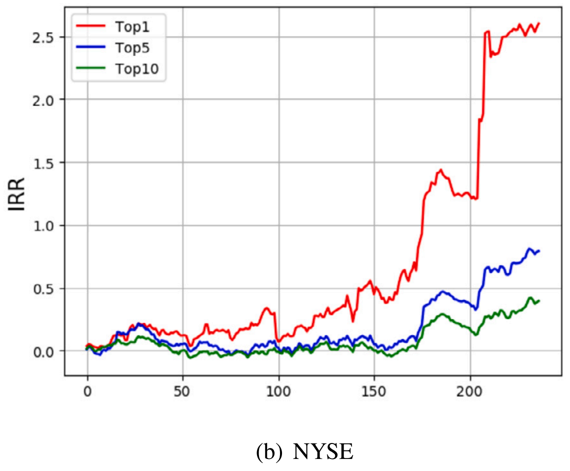
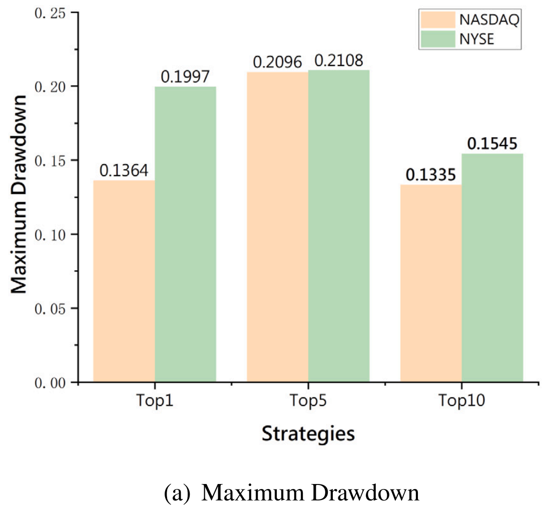


Fig. 6. IRR performance of different strategies in NASDAQ and NYSE.



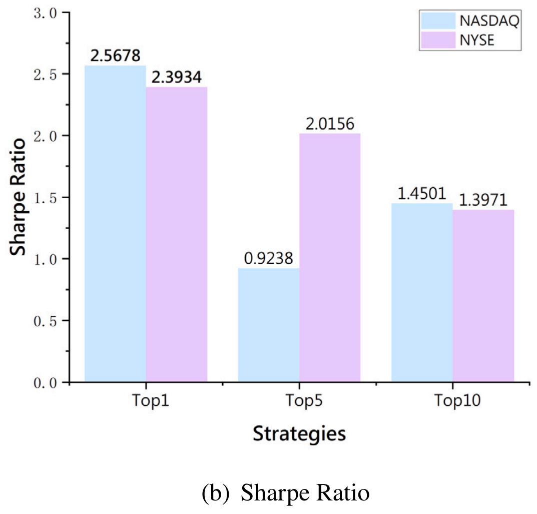


Fig. 7. Maximum drawdown and Sharpe ratio for Top1, Top5, and Top10 strategies.

means a higher return on investment. In NASDAQ market, at the beginning, the Top1's return is lower than the Top5 and Top10, after around day 100 of the test set, Top5 lags behind Top10 by a small margin, which is different from the results of NYSE market, this is because NASDAQ market is a more volatile market [16], and when the predicted ranking is far from the actual, it will lead to a large deviation in returns.

Fig. 7 shows that on the two datasets, Sharpe ratio of Top1 is higher than that of Top5 and Top10, which indicates that Top1 is a better strategy when comprehensively balancing risks and benefits. In both markets, the maximum drawdown of Top10 is lower than that of Top1 and Top5, indicating that the strategy of multiple stock portfolios can effectively alleviate the sharp volatility of returns. The reason for the maximum drawdown of Top5 $>$ the maximum drawdown of Top1 may be that the number of stocks selected is small and cannot effectively alleviate the volatility of stock returns.

Further, to assess the IRR performance of our model, we chose the stock with the highest return during the test period (such as top5) on the two data sets used in this experiment to compare with MGAR model, and the results are shown in Fig. 8. In the two markets, the IRR performance of Top1 of MGAR model is marginally higher than that of the Top 1 stock selected during the test period. Although our method does not consider the transaction fee, this performance is quite good, which further proves the effectiveness of MGAR. Top5 and Top10 have a much lower IRR than the top 5 and top 10 stocks selected from the sample, since it is very difficult to predict the top 5 or 10 stocks with the highest return. In addition, it indicates that stock ranking still has a lot of room for improvement.

5.3.6. Parameter sensitivity analysis

In order to see the sensitivity of the model to parameters, we change the objective function α and stock sequence input length T for parameter sensitivity analysis.

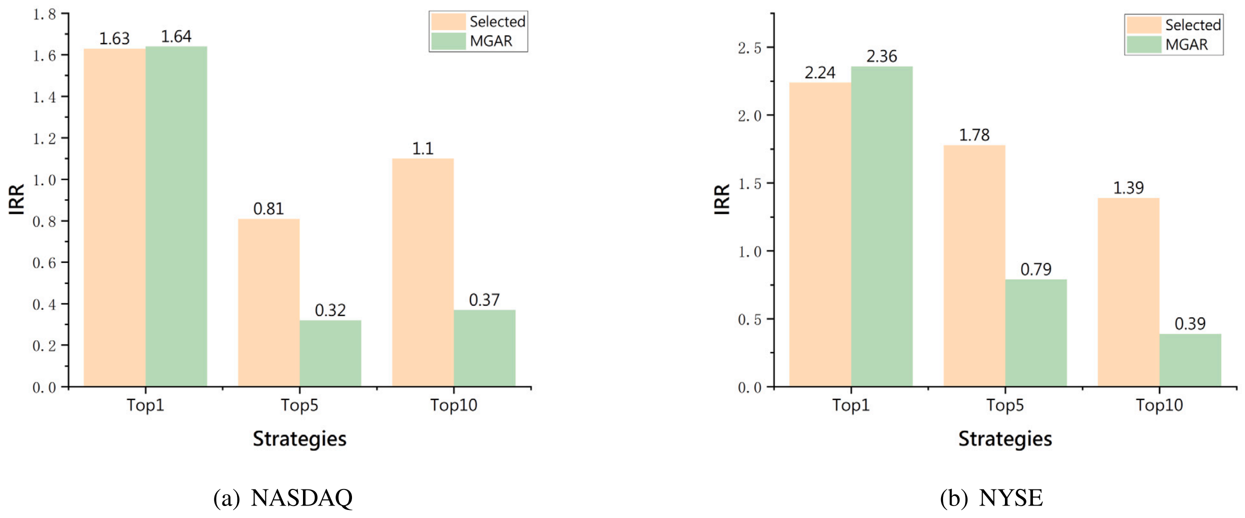


Fig. 8. Performance of MGAR compared to ideal strategies in both markets.

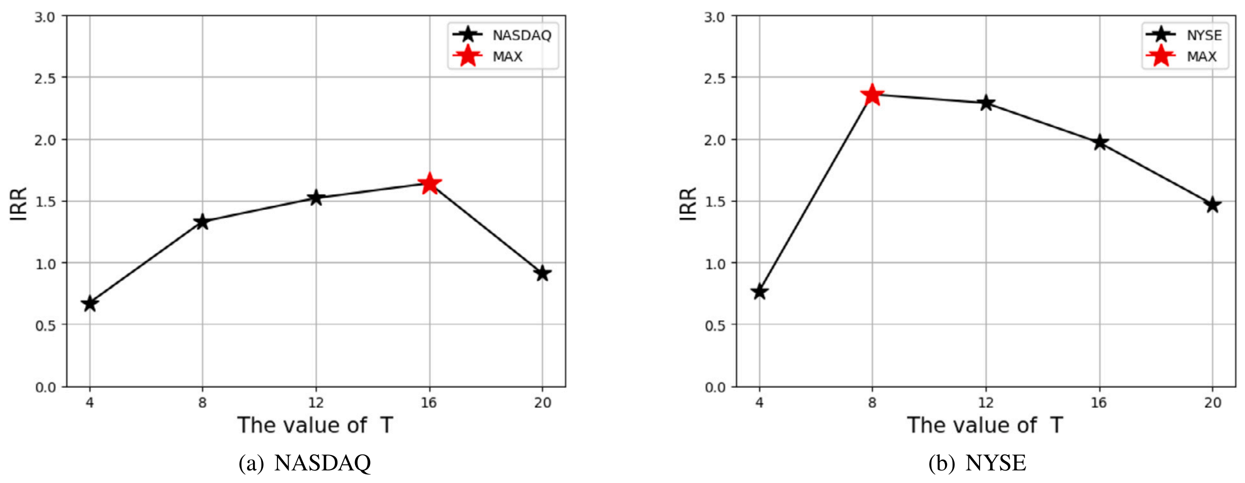


Fig. 9. The objective function of the MGAR model under different T, the IRR performance of the two datasets.

(1) Input length of stock price sequence T: In order to verify the performance of the MGAR model under different stock price sequence input lengths T, we conduct experiments in NASDAQ and NYSE markets respectively to observe the corresponding IRR performance. The T values are 4, 8, 12, 16 and 20, respectively, and the results are shown in Fig. 9. Our model's IRR performance varies with sequence length input T.

In NASDAQ market, when T is 16, the IRR of the model achieves the maximum value, and in NYSE market, when T is 8, the IRR achieves the maximum value, which are 1.64 and 2.36, respectively. On the two datasets, when T is 4, the worst results of the selected T values are obtained, which are 0.67 and 0.76 respectively. This may be because the length of the sequence is too short and the information of historical prices is insufficient. With the increase of T value, more useful price information is embedded into the sequence to improve the IRR. However, when the length reaches a certain larger value, as the sequence length gets greater, more redundant information is added, and some useless information is embedded in the sequence. Thus, the IRR values increase first and then decrease.

(2) α of the objective function: In order to verify the performance of the MGAR model under different α values, experiments are performed on the two datasets to observe the corresponding IRR performance. The α values are taken as 0, 0.1, 1, 5, 10 and 20, respectively, and the results are shown in Fig. 10. In NASDAQ market, when $\alpha = 0$, the IRR is 0.88, and when $\alpha = 0.1$, the IRR achieves the maximum value of 1.64, and when $\alpha = 1$, 5, 10, the corresponding IRR drops to 1.31, 1.10, 0.75, when $\alpha = 20$, the IRR is 0.77, which is not much changed from the IRR of $\alpha = 10$. In NYSE market, IRR also shows a trend of first rising and then falling with the increase of α. When $\alpha = 0$, 0.1, 1, and 5, the IRR gradually increases to 0.76, 0.80, 1.86, and 1.97, respectively. When $\alpha = 10$, the IRR reaches the maximum value of 2.36. When $\alpha = 20$, the IRR is 1.41, and there is a large decline.

As an increasing α is applied in the objective function, the pairwise ranking-aware loss also increases gradually, and its objective is to ensure that the predicted ranking of stocks is as close as possible to the actual ranking, so that the predicted stocks will also have a more accurate ranking order, whereby the IRR will also increase gradually. Once α reaches a certain value, the pointwise

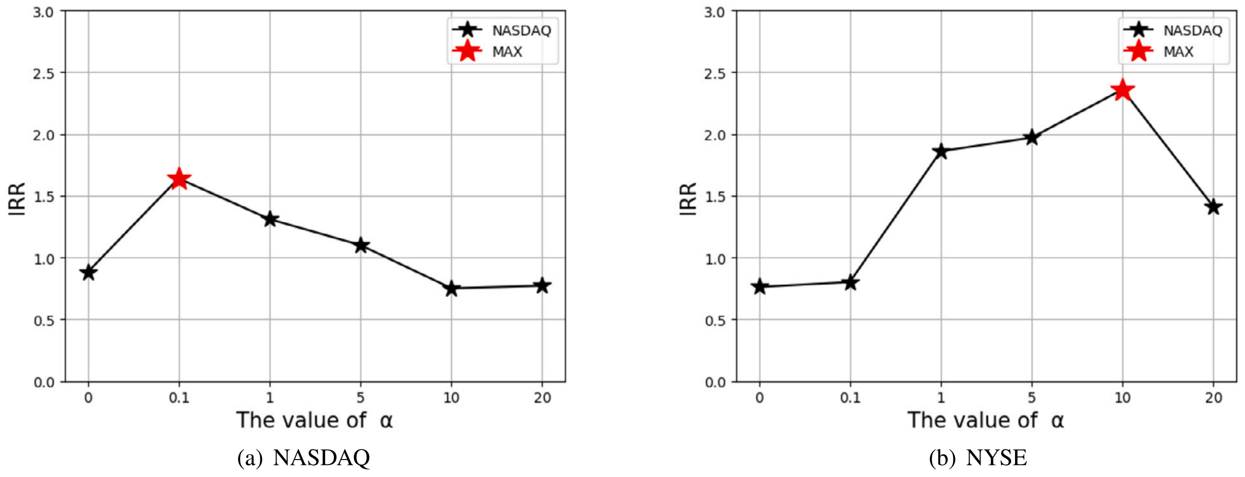


Fig. 10. The objective function of the MGAR model under different α, the IRR performance of the two datasets.

regression loss subsists in a small proportion and is virtually ignored, resulting in a decline in the accuracy of stock ranking score prediction, which impacts the stock's ranking error, and thus impacts its final IRR.

5.3.7. Summary of experimental results

A large number of experiments are conducted to prove the effectiveness of the proposed model, including comparative experiments and ablation experiments, etc. In the comparative experiment, Rank_LSTM is obviously better than LSTM, because pointwise regression loss and pairwise ranking-aware loss are combined in the loss function of the prediction layer, and ground-truth and ranking accuracy have been comprehensively optimized, indicating that the stock ranking prediction can significantly improve the prediction accuracy. On the two data sets, compared with other models, the MGAR model achieves greater improvements in various indicators. In particular, on the return indicator IRR, the average return rates of 164% and 236% are achieved respectively, which proves the effectiveness of the MGAR model. Among them, in the NASDAQ data set, other graph-based learning models do not improve significantly compared with Rank_LSTM, because the NASDAQ market is more volatile than the NYSE market [16], and the stock relationship is more complex. The static graph based on priori knowledge can not learn the changes of stock relationship that may occur in this volatile market. And if a single relationship graph is used, it is difficult to capture sufficient effective information. In this paper, we construct price similarity relation graph to aggregate stock price relationships in a time-sensitive manner. The designed MGA module effectively fuses the three relation graphs, which can capture more representational stock relation embeddings, thereby improving the accuracy of stock predicting. Our model also benefits from accurate stock rankings to minimize investment risk, achieving excellent performance on both maximum drawdown and Sharpe ratio results. Furthermore, ablation experiment shows that the performance of the model is significantly reduced without providing price similarity relation graph, which illustrates that price similarity relation graph are valuable informative complements to industry relation graph and Wiki relation graph. Without using the attention module, the model cannot effectively fuse the three relation graphs. After removing the stitching of the price trend embedding, the model performance degrades due to the lack of augmentation of the stock's own price information. The results of these tests prove the effectiveness of each module.

Although this paper has integrated three types of relation graphs to mine relationship information and uses attention mechanism to remove introduced noise, there are many factors affecting stocks that make stock relationships complex and variable, making it difficult to simulate and quantify them. Industry relation graph and Wiki relation graph constructed based on priori knowledge in this paper may even give incorrect information. Furthermore, using multiple stock graphs results in a lot of memory usage due to the large number of stocks. More stock graphs are constructed, the slower the model runs, which makes it difficult to achieve fast and timely stock trading in the real investment environment.

6. Summary and prospect

In this paper, the stock prediction problem is set up as return ranking prediction in order to get the maximum return. A variety of relationship graphs are constructed in this paper to model the comprehensive impact of various complex relationships among stocks on stock price changes. Exploiting new stock relationships and obtaining useful information from the interweaving of different relationships is a very challenging task. Based on the fact that all relevant information will be reflected in the stock price [18], we construct a new stock relation graph based on historical stock price information of multiple views, called a price similarity relation graph. It can effectively explore the price relationship between related stocks (such as rising and falling stocks) and can dynamically supplement information to the static relation graph. It is another effective stock relationship, including wiki relationships and industry relationships. The MGAR stock ranking prediction model uses these three types of relationship graphs exactly. The LSTM network is used in the feature extraction layer of the MGAR model to extract price trend embedding from historical stock price series. The newly designed MGA module mines information from each stock relation graph using graph convolution. Due to the difficulty of manually setting the weights of the three relations in the training, we introduce the attention mechanism adaptive learning to achieve effective information fusion. As a final step, the price trend embedding and relation embedding are spliced together and input into the full connection layer to predict the stock return ranking.

The experimental results on several real data sets show that the model can effectively learn the price relation between stocks over time through the constructed price similarity graph; by using three types of stock relation graphs, we can mitigate the problem of insufficient effective information caused by the sparsity of a single stock relation graph; utilizing the complementarity of the three relational graphs, the newly designed MGA module achieves effective relation embedding.

Stocks can have many types of relationships with each other. We will explore other ways to further improve and expand the relation graph between stocks in the future. The model structure should also be improved to deal with the problem of running time due to the complexity of the stock graph. Considering that public opinion influences the future trend of stock prices [45], we will also investigate integrating financial news and stock review data into the prediction model to enhance its performance.

CRediT authorship contribution statement

Guowei Song: Software, Investigation, Writing – original draft. **Tianlong Zhao:** Software, Conceptualization, Writing and editing. **Suwei Wang:** Visualization, Investigation. **Hua Wang:** Conceptualization, editing. **Xuemei Li:** Conceptualization, Methodology, Writing – review and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

We thank the anonymous reviewers for their constructive comments. This work was supported in part by the National Natural Science Foundation of China (Grant No. 62072281, No. 62007017).

References

[1] Z. Liu, Z. Dang, J. Yu, Stock price prediction model based on rbf-svm algorithm, in: 2020 International Conference on Computer Engineering and Intelligent Control (ICCEIC), 2020, pp. 124–127.
[2] K.K. Yun, S.W. Yoon, 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.
[3] Residual Long Short-Term Memory Network with Multi-Source and Multi-Frequency Information Fusion: An Application to China's Stock Market, Inf. Sci. 622 (2023) 133–147, https://doi.org/10.1016/j.ins.2022.11.136.
[4] Multi-step-ahead stock price index forecasting using long short-term memory model with multivariate empirical mode decomposition, Inf. Sci. 607 (2022) 297–321, https://doi.org/10.1016/j.ins.2022.05.088.
[5] U. Gupta, V. Bhattacharjee, P.S. Bishnu, Stocknet—gru based stock index prediction, Expert Syst. Appl. 207 (2022) 117986, https://doi.org/10.1016/j.eswa.2022.117986.
[6] Adaptive stock trading strategies with deep reinforcement learning methods, Inf. Sci. 538 (2020) 142–158, https://doi.org/10.1016/j.ins.2020.05.066.
[7] D.M. Nelson, A.C. Pereira, R.A. De Oliveira, Stock market's price movement prediction with lstm neural networks, in: 2017 International Joint Conference on Neural Networks (IJCNN), Ieee, 2017, pp. 1419–1426.
[8] T. Fischer, C. Krauss, Deep learning with long short-term memory networks for financial market predictions, Eur. J. Oper. Res. 270 (2018) 654–669, https://doi.org/10.1016/j.ejor.2017.11.054.
[9] B. Zhang, Z. Li, C. Yang, Z. Wang, Y. Zhao, J. Sun, L. Wang, Similarity analysis of knowledge graph-based company embedding for stocks portfolio, in: 2021 IEEE 6th International Conference on Smart Cloud (SmartCloud), IEEE, 2021, pp. 84–89.
[10] J. Ye, J. Zhao, K. Ye, C. Xu, Multi-graph convolutional network for relationship-driven stock movement prediction, in: 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, 2021, pp. 6702–6709.
[11] D. Matsunaga, T. Suzumura, T. Takahashi, Exploring graph neural networks for stock market predictions with rolling window analysis, arXiv preprint, arXiv:1909.10660, 2019, https://doi.org/10.48550/arXiv.1909.10660.
[12] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, arXiv preprint, arXiv:1609.02907, 2016, https://doi.org/10.48550/arXiv.1609.02907.
[13] A novel graph convolutional feature based convolutional neural network for stock trend prediction, Inf. Sci. 556 (2021) 67–94, https://doi.org/10.1016/j.ins.2020.12.068.
[14] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint, arXiv:1710.10903, 2017, https://doi.org/10.48550/arXiv.1710.10903.
[15] Y. Wang, Y. Qu, Z. Chen, Review of graph construction and graph learning in stock price prediction, Proc. Comput. Sci. 214 (2022) 771–778, https://doi.org/10.1016/j.procs.2022.11.240.
[16] F. Feng, X. He, X. Wang, C. Luo, Y. Liu, T.-S. Chua, Temporal relational ranking for stock prediction, ACM Trans. Inf. Syst. 37 (2019) 1–30, https://doi.org/10.1145/3309547.
[17] K. Hou, Industry information diffusion and the lead-lag effect in stock returns, Rev. Financ. Stud. 20 (2007) 1113–1138, https://doi.org/10.1093/revfin/hhm003.

[18] E.F. Fama, Efficient capital markets: a review of theory and empirical work, J. Finance 25 (1970) 383–417, https://doi.org/10.2307/2325486.

[19] J. Gao, X. Ying, C. Xu, J. Wang, S. Zhang, Z. Li, Graph-based stock recommendation by time-aware relational attention network, ACM Trans. Knowl. Discov. Data 16 (2021) 1–21, https://doi.org/10.1145/3451397.

[20] Y. Chen, Z. Wei, X. Huang, Incorporating corporation relationship via graph convolutional neural networks for stock price prediction, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 1655–1658.

[21] S. Al-Janabi, A.F. Alkaim, A nifty collaborative analysis to predicting a novel tool (drflls) for missing values estimation, Soft Comput. 24 (2020) 555–569, https://doi.org/10.1007/s00500-019-03972-x.

[22] B.K. Jha, S. Pande, Time series forecasting model for supermarket sales using fb-prophet, in: 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), IEEE, 2021, pp. 547–554.

[23] Z.A. Kadhuim, S. Al-Janabi, Codon-mrna prediction using deep optimal neurocomputing technique (dlstm-dsn-woa) and multivariate analysis, Results Eng. 17 (2023) 100847, https://doi.org/10.1016/j.rineng.2022.100847.

[24] C. Yuan, X. Ma, H. Wang, C. Zhang, X. Li, Covid19-mlsf: a multi-task learning-based stock market forecasting framework during the Covid-19 pandemic, Expert Syst. Appl. 217 (2023) 119549, https://doi.org/10.1016/j.eswa.2023.119549.

[25] R. Gao, X. Zhang, H. Zhang, Q. Zhao, Y. Wang, Forecasting the overnight return direction of stock market index combining global market indices: a multiple-branch deep learning approach, Expert Syst. Appl. 194 (2022) 116506, https://doi.org/10.1016/j.eswa.2022.116506.

[26] C. Cui, X. Li, J. Du, C. Zhang, X. Nie, M. Wang, Y. Yin, Temporal-relational hypergraph tri-attention networks for stock trend prediction, arXiv preprint, arXiv:2107.14033, 2021, https://doi.org/10.48550/arXiv.2107.14033.

[27] R. Akita, A. Yoshihara, T. Matsubara, K. Uehara, Deep learning for stock prediction using numerical and textual information, in: 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), IEEE, 2016, pp. 1–6.

[28] R. Cheng, Q. Li, Modeling the momentum spillover effect for stock prediction via attribute-driven graph attention networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 1, 2021, pp. 55–62.

[29] Y. Wan, C. Yuan, M. Zhan, L. Chen, Robust graph learning with graph convolutional network, Inf. Process. Manag. 59 (2022) 102916, https://doi.org/10.1016/j.ipm.2022.102916.

[30] R. Sawhney, S. Agarwal, A. Wadhwa, T. Derr, R.R. Shah, Stock selection via spatiotemporal hypergraph attention network: a learning to rank approach, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 497–504.

[31] B.K. Meher, I.T. Hawaldar, C.M. Spulbar, F.R. Birau, Forecasting stock market prices using mixed arima model: a case study of Indian pharmaceutical companies, Invest. Manag. Financ. Innov. 18 (2021) 42–54, https://doi.org/10.21511/imfi.18(1).2021.04.

[32] X. Zhang, W. Chen, Stock selection based on extreme gradient boosting, in: 2019 Chinese Control Conference (CCC), IEEE, 2019, pp. 8926–8931.

[33] C. Saiktishna, N.S.V. Sumanth, M.M.S. Rao, J. Thangakumar, Historical analysis and time series forecasting of stock market using fb prophet, in: 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS), IEEE, 2022, pp. 1846–1851.

[34] A. Mueen, N. Chavoshi, N. Abu-El-Rub, H. Hamooni, A. Minnich, J. MacCarthy, Speeding up dynamic time warping distance for sparse time series data, Knowl. Inf. Syst. 54 (2018) 237–263, https://doi.org/10.1007/s10115-017-1119-0.

[35] A. Sharabiani, H. Darabi, S. Harford, E. Douzali, F. Karim, H. Johnson, S. Chen, Asymptotic dynamic time warping calculation with utilizing value repetition, Knowl. Inf. Syst. 57 (2018) 359–388, https://doi.org/10.1007/s10115-018-1163-4.

[36] S. Al-Janabi, M. Mohammad, A. Al-Sultan, A new method for prediction of air pollution based on intelligent computation, Soft Comput. 24 (2020) 661–680, https://doi.org/10.1007/s00500-019-04495-1.

[37] D. Singh, B. Singh, Investigating the impact of data normalization on classification performance, Appl. Soft Comput. 97 (2020) 105524, https://doi.org/10.1016/j.asoc.2019.105524.

[38] H. Rezaei, H. Faaljou, G. Mansourfar, Stock price prediction using deep learning and frequency decomposition, Expert Syst. Appl. 169 (2021) 114332, https://doi.org/10.1016/j.eswa.2020.114332.

[39] T. Liu, X. Ma, S. Li, X. Li, C. Zhang, A stock price prediction method based on meta-learning and variational mode decomposition, Knowl.-Based Syst. (2022) 109324, https://doi.org/10.1016/j.knosys.2022.109324.

[40] S. Al_Janabi, A. Yaqoob, M. Mohammad, Pragmatic method based on intelligent big data analytics to prediction air pollution, in: Y. Farhaoui (Ed.), Big Data and Networks Technologies, Springer International Publishing, Cham, 2020, pp. 84–109.

[41] S. Feng, C. Xu, Y. Zuo, G. Chen, F. Lin, J. XiaHou, Relation-aware dynamic attributed graph attention network for stocks recommendation, Pattern Recognit. 121 (2022) 108119, https://doi.org/10.1016/j.patcog.2021.108119.

[42] Z. Niu, G. Zhong, H. Yu, A review on the attention mechanism of deep learning, Neurocomputing 452 (2021) 48–62, https://doi.org/10.1016/j.neucom.2021.03.091.

[43] W. Bao, J. Yue, Y. Rao, A deep learning framework for financial time series using stacked autoencoders and long-short term memory, PLoS ONE 12 (2017) e0180944, https://doi.org/10.1371/journal.pone.0180944.

[44] S. Al-Janabi, A.F. Alkaim, Z. Adel, An innovative synthesis of deep learning techniques (dcapsnet & dcom) for generation electrical renewable energy from wind energy, Soft Comput. 24 (2020) 10943–10962.

[45] N. Seong, K. Nam, Predicting stock movements based on financial news with segmentation, Expert Syst. Appl. 164 (2021) 113988, https://doi.org/10.1016/j.eswa.2020.113988.