Information Sciences 654 (2024) 119833

Contents lists available at ScienceDirect

Information Sciences

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

DR-GAT: Dynamic routing graph attention network for stock recommendation

Zengyu Lei [a], Caiming Zhang [a,b], Yunyang Xu [a], Xuemei Li [a,*]

[a] School of Software, Shandong University, Jinan 250101, China
[b] Shandong Provincial Laboratory of Future Intelligence and Financial Engineering, Yantai 264005, China

ARTICLE INFO

Keywords:
Stock recommendation
Graph attention networks
Dynamic stock relation
Dynamic routing

ABSTRACT

Investors are increasingly interested in using financial technology to guide investment decisions. The phenomenon of "stock linkage" or "leading-lag" is often observed between related stocks on the stock market. Based on this feature of the stock market, complex relationships between stocks influence investment decisions. Existing methods use static prior relational data (e.g., industry relations and Wiki relations) to build corporate relation graph, while learning relational features using the same graph for all stocks. This description and use of corporate relationships ignored the dynamic nature of association relationships in time and space, limiting the ability to learn latent relationships between stocks. To address these issues, we propose a Dynamic Routing Graph Attention Network (DR-GAT) for stock recommendation. We propose a novel similarity measurement method called stock price trend similarity. To track the evolution of intercorporate relationships over time, we dynamically construct relation graph attention networks using prior knowledge and stock price trend similarity. Moreover, a newly designed relation graph router (RGR) can route each stock to an optimal relationship graph based on its volatility. Extensive experiments demonstrate the superiority of our DR-GAT method. It outperforms state-of-the-art methods achieving an average return ratio of 152% and 162% on NASDAQ and NYSE, respectively.

1. Introduction

Stock market capital has grown rapidly, attracting an increasing number of investors. As stock investments often high profits and high risks, investors need effective models for analyzing the current market situation, predicting future market trends, and selecting stocks with profit potential. Deep learning provides a promising prospect for financial time series prediction [1]. The stock market's uncertainty and the stock price's volatility, however, pose challenges to stock prediction.

The traditional Auto Regression-based models [2–5] can predict future data prediction by fitting historical data, but the stochastic processes represented by such models cannot accurately model stock price fluctuations in the real world. To address this issue, deep neural networks, and especially Recurrent Neural Network (RNN) [6–8], have become a main approach for stock price prediction by mapping the data to a high-dimensional space for data fitting. However, Recurrent Neural Network (RNN) can only vertically propagate and aggregate information from adjacent time nodes. When making stock predictions, each stock is usually treated as an independent sample, ignoring the rich dynamic signals between related stocks. In fact, there are obviously complex correlations

* Corresponding author.
E-mail addresses: leizengyu@mail.sdu.edu.cn (Z. Lei), czhang@sdu.edu.cn (C. Zhang), xuyunyang@mail.sdu.edu.cn (Y. Xu), xmli@sdu.edu.cn (X. Li).

https://doi.org/10.1016/j.ins.2023.119833
Received 29 March 2023; Received in revised form 12 August 2023; Accepted 30 October 2023

Available online 7 November 2023

between stocks, such as upstream and downstream relationships in the industrial chain and related industries. Companies' stock prices are affected by overall market conditions and those of their related companies. Recursive Neural Networks (RecNNs) not only have the ability to learn temporal features but also can describe feature correlations by constructing tree-like structures [9]. They have demonstrated promising performance in learning correlated features through hierarchical structures [10]. However, corporate relations among stocks are complex, and representing the intricate relationships between stocks as tree structures can lead to significant information loss. As a result, Recursive Neural Networks have not been extensively utilized in learning stock relational features. The Graph Neural Network (GNN) builds a graph based on the relationships between samples, and aggregates the features of neighboring nodes to update the central node's features. To capture the relationship between stocks, GNN-based algorithms [11,12] represent the features of stocks as nodes and construct edges between related stocks to enable information propagation between stocks. Defining the relationship between stocks is a challenging task. Several algorithms exist that use shareholding relationships between companies to define the edges of the corporation relationship graph [13]. Feng et al. defined the relationships between stocks by using publicly available corporate relationship data [14]. FinGAT [15] modeled the hierarchical correlation of stocks within and between sectors to capture the relational features of stocks. Despite these models' successes in capturing the interrelationship between stocks, the mining of new types of corporation relationship data and the creation of relation graphs need improvement. Firstly, the ability to acquire prior knowledge such as "shareholding relationships" and "membership relationships", as well as the ability to express stock relationships through prior knowledge, is limited. It is hard to express in a model the impact of market uncertainty on stocks and latent relationships between stocks flexibly and intuitively through prior relationship data. Secondly, the stock relationships based on prior knowledge such as "upstream and downstream relationships in the industrial chain", "internal relationships within sectors", and "inter-sector relationships" are pre-determined and difficult to expand, resulting in the long-term following of the same pattern when conducting relational feature learning. The market is inherently uncertain, and a variety of factors, including policies, sudden social phenomena, and the overall market situation, can cause dynamic changes in the relevance degree between stocks. Static graphs based on prior knowledge are insufficient for capturing dynamic changes in the relationships between listed companies. The aforementioned challenges make it a challenging task to model corporate relations and utilize them to enhance the predictive capabilities of the model.

In the real market, each company establishes connections with many other companies through different types of relationship data (e.g., industry relationship data, stock price similarity data). Amazon, for instance, associates with Microsoft through the concept of "cloud computing" and with eBay through the concept of "e-commerce". The relation graph established using these prior relationship data is long-lasting. Such long-term corporate relation graphs can usually be effectively used for relational feature learning when the stock price series is relatively stable. Unexpected events tend to impact the effectiveness of long-term relationship graphs, however. This is because when the market environment changes, stocks react differently to sudden market events, causing their relationships to change. For example, at the onset of the COVID-19 pandemic, Amazon's stock price rose due to its "e-commerce" business [16]. Therefore, the business relationship between Amazon and eBay, in contrast with the business relationship with Microsoft, is more effective in stock prediction at present. If continuing to use long-term corporate relation graph to learn relational features, although some effective corporate relationships (i.e., Business relationships related to "e-commerce") are retained, ineffective corporate relationships (i.e., Business relationships related to "cloud computing") inevitably become noise in relational feature learning. Additionally, Xu et al. [16] mentioned that personal protective equipment companies may have accidentally shared information and similar future trends with e-commerce companies during the outbreak of the pandemic using hidden concepts related to the pandemic. However, this relevance will be lost after the pandemic has passed. The existing work fails to consider the dynamic evolution of corporate relations and lacks the ability to adapt more effective corporate relation graphs based on stock characteristics. In fact, it is imperative to consider short-term corporate relationships that arise and exist in the short term when dealing with a sudden event, and to construct a short-term corporate relationship graph which encompasses as many effective business relationships as possible. Additionally, different companies exhibit diverse reactions to market conditions. However, the current methods neglect this characteristic when learning relational features by employing the same pattern to learn relational features for all stocks. Consequently, some stocks inevitably miss out on more effective corporate relation graphs during the relational feature learning process.

To address the difficulty of representing implicit latent relationships in prior knowledge, as well as the dynamic nature of stock relationships that cannot be reflected in prior knowledge, this paper constructs a dynamic relation graph that changes with time by combining prior corporate relationship data with similarity in stock price trends. Stock prices may also fluctuate due to different responses to unexpected market events during the same period, causing changes in the intercorrelation between companies. Considering the changes in the intercorrelation between companies caused by the uncertainty of such market, we divide the constructed dynamic relation graph of companies into two types, a long-term corporate relation graph and a short-term corporate relation graph. The newly designed Relation Graph Router (RGR) is then used to automatically select a more effective relationship graph based on the degree of stock price fluctuations, while Graph Attention Network (GAT) is used to capture corporate relational features, which are incorporated into the prediction model as incremental information. The Dynamic Routing Graph Attention Network (DR-GAT), our proposed predictive model for capturing dynamic relational features through automatic routing and dynamic graph construction, is shown in Fig. 1. We conducted experiments on over 2852 stocks and 1174 trading days in the NYSE and NASDAQ real stock markets and found that DR-GAT outperforms other models in predicting stock rankings. The main contributions of this article are as follows:

- A novel dynamic routing graph attention network model DR-GAT is developed to recommend profitable stocks. Through DR-GAT, latent relationships between stocks can be dynamically captured from both spatial and temporal perspectives.

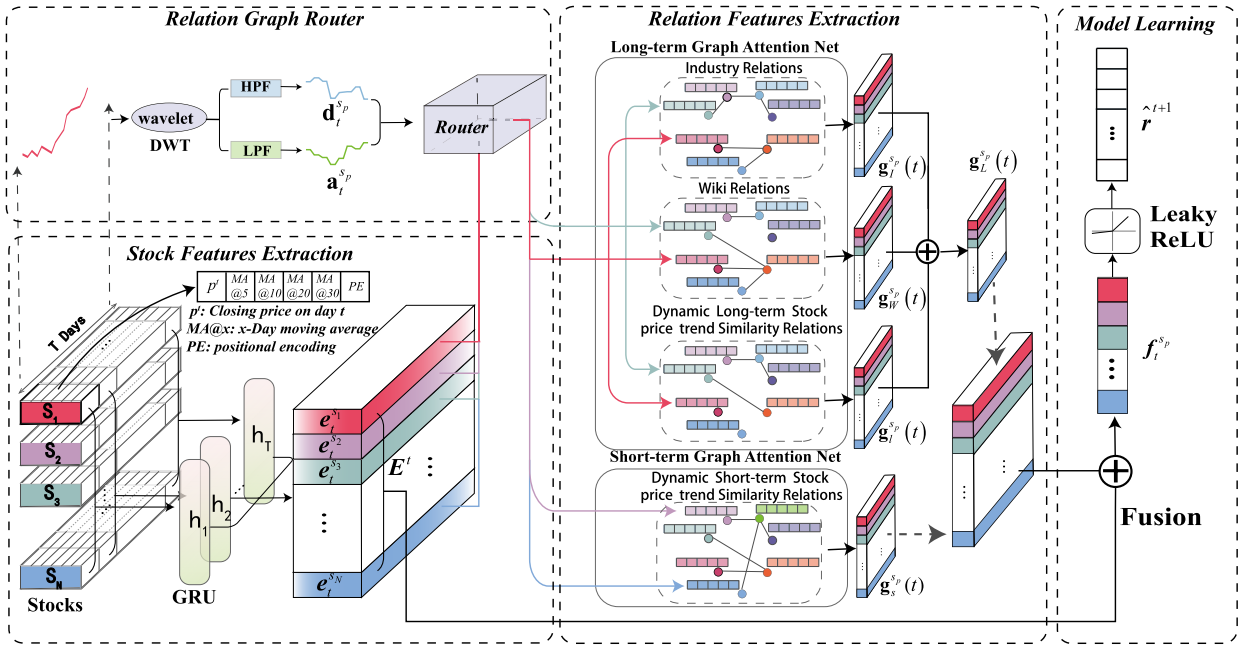


Fig. 1. The architecture of our proposed DR-GAT model.

- The proposed Relation Graph Router (RGR) module can automatically adapt an effective corporation relationship graph for each stock based on its price volatility, fully considering the impact of price fluctuations on intercorrelations among corporations during the process of learning stock relational features.
- We propose a method for dynamically learning stock relational features at the temporal level by combining stock price trend similarity and pre-defined prior relationship data to construct a dynamic corporation relationship graph. The stock price trend similarity compensates for the deficiency of prior relationship data, and the dynamic graph construction method captures the dynamic correlation between corporations. Dynamic corporation relationship graph constructed in this way can grasp market changes in real time and accurately.

The remaining parts of this paper are organized as follows. Section 2 discusses related work. Section 3 introduces the algorithm details of the proposed DR-GAT. Section 4 reports the experimental results, and Section 5 summarizes the work of this paper.

2. Related work

Our work is directly related to the recent work on graph-based learning and multiple stocks prediction.

2.1. Stock prediction with graph-based learning

Recent studies in stock price prediction attempt to analyze and predict stock behavior by learning relationships between stocks. With their ability to represent relational data, graphs are an effective data structure for learning correlations between stocks. A graph consists of vertices and edges, mathematically represented as an adjacency matrix. In stock price prediction, stock sequential features are used as vertices, and edges are constructed based on the relationships between stocks.

The two primary tasks of graph-based learning for stock prediction are extracting features from the stock price series and constructing edges between stocks. Recently, there has been a trend in combining graph neural networks (GNNs) with recurrent neural networks (RNNs) for stock price series prediction. Specifically, the historical stock price series is fed into an RNN to extract features, and the resulting series embedding is then used as the vertices of a GNN. Xu et al. [17] combined the hierarchy structure with graph neural networks to model the relationships between stocks. Wu et al. [18] mapped the stock price sequences to a price graph and learned the relational features through the weights between nodes. When constructing edges, the usual practice is to rely on open-source knowledge graphs. Currently, many works use the combination of historical closing prices and moving average lines as the raw stock features input to LSTM [14] or GRU [15] for feature extraction. Meanwhile, HATS [19] inputs the combination of historical closing prices and price change rates to LSTM for feature extraction. When constructing the adjacency matrix, some of the work [14,19] involves constructing edges using publicly available knowledge bases containing various types of corporate relationships, such as industry, personnel, and industry relationships. FinGAT [15] hierarchically constructs intra-sector relation graphs and inter-sector relation graphs using industry data. For the constructed stock relation graph, Feng et al. [14] combined temporal convolution with graph convolution to perform graph learning and proposed a Temporal Graph Convolution (TGC) to capture the

correlation between stocks in a time-sensitive manner. The emergence of Graph Attention Networks (GAT) [20] has propelled further developments in graph-based stock prediction. GAT learns attention weights between nodes to obtain the contribution of related stock pairs to the target stock, thus enabling inductive learning. When aggregating information from adjacent nodes, GAT only considers the features of the target node and its adjacent nodes, independent of a fixed graph structure, making it suitable for the dynamic relationships in the stock market. FinGAT [15] utilizes Graph Attention Networks to learn the relation graph and aggregate the relational features of stocks, and its results demonstrate the promising prospect of Graph Attention Networks in stock price prediction. Even though graph-based stock prediction algorithms have achieved certain results, dynamic correlations between stocks usually cannot be manifested through open-source prior relationship data. Therefore, there is still considerable room for improvement in the graph construction aspect of existing work.

2.2. Multiple stocks prediction

When compared with predicting the future trends or prices of individual stocks, forecasting multiple stocks to obtain stock rankings can provide investors with better recommendations for profitable stocks. Fischer et al. [21] applied LSTM networks to make multiple stock predictions for all S&P 500 index component stocks, selected well-performing stocks for long positions based on the prediction results, and shorted poorly-performing stocks at the same time. However, although these methods achieved multi-stock prediction, they still isolated different stocks for prediction in practice, ignoring comparisons between stocks. Therefore, more advanced work organizes multiple stocks using high-dimensional tensors and introduces ranking algorithms to enable comparison between stocks during the prediction process. On the one hand, multi-stock prediction algorithms [22,14] that organize stock features through high-dimensional tensors can predict all stocks within the tensor simultaneously, improving prediction efficiency. On the other hand, organizing stock data using multi-dimensional tensors is advantageous for preserving relationships between companies and enabling comparisons between them in subsequent learning processes. Therefore, tensors are widely used in multiple stock prediction algorithms. The introduction of ranking algorithms allows the model to preserve the mutual influence between companies during the learning process. Song et al. [23] introduced investor sentiment data in stock prediction, combining neural networks with learning-to-rank algorithms (ListNet and RankNet) to generate a ranking model. The investment strategy of holding the top 25% of stocks long and the bottom 25% of stocks short is determined by using the model to predict stock rankings. Feng et al. [14] combined pointwise regression loss and pairwise ranking-aware loss to predict the ranking of stocks. FinGAT [15] extends the objective function proposed by Feng et al. by adding cross-entropy loss of stock trends, and employs multi-task learning of stock price prediction and trend prediction to predict the ranking of stocks. The above-mentioned algorithm uses a ranking algorithm to preserve the comparison between stocks during the model optimization process, but it adopts the same learning pattern for all stock sequences, without adjusting the optimal learning pattern according to the differences in stock features for different stocks.

3. Methodology

The proposed DR-GAT model consists of four main components: (1) stock temporal feature extraction, (2) corporate relation graph router, (3) corporate relational feature extraction, and (4) model learning, as shown in Fig. 1. The stock feature extraction module is used to obtain sequential embeddings from the stock price sequences. The relation graph router module routes the sequential embeddings to the effective relation graphs. The relation features extraction module learns relational features using the sequential embeddings as node information. Finally, the sequential embeddings and the relational features are fed into the model learning module, where the model generates rankings based on the returns and is optimized through a loss function. Following this process, we will now present the detailed algorithm description. First, the multi-view stock price vectors are processed by gated recurrent units (GRU) to extract features and obtain sequential embeddings. Positional information is crucial for time series and can be encoded in order to enable the data to carry its positional information. This enables the model to capture temporal dependency more effectively. Hence, we add a positional encoding dimension to the input stock price vectors to allow the GRU to better capture positional characteristics during feature extraction. The paper proposes a Relation Graph Router (RGR) based on stock price volatility for relation graph selection. The RGR offers two routing paths for sequential embedding: short-term relation graphs and long-term relation graph, based on the frequency components. In RGR, frequency components of each closing price series within the current time window are obtained using discrete wavelet transform (DWT), and they are then classified based on whether they oscillate or not, in order to route the current stock price series embedding to the optimal corporate relation graph based on the frequency components. Among them, the corporate relation graph is dynamically updated based on the similarity of stock price trends. Once the stock price sequential embedding is dynamically routed to an effective relation graph, the GAT is used to learn the relational features. Finally, we feed the stock price sequential embeddings extracted by GRU and dynamically aggregated relational features to the fully connected layer, generating a ranking of returns to complete the stock recommendation process.

3.1. Problem statement

The purpose of our study is to develop a rational investment strategy for maximizing profits. Let $S = \{s_1, s_2, ..., s_N\}$ denote the set of N stocks, where for each stock $s_p \in S$ on trading day t, its closing price is $p_t^{s_p}$ and its return rate $R_t^{s_p}$ is given by:

$$R_t^{s_p} = \frac{p_t^{s_p} - p_{t-1}^{s_p}}{p_{t-1}^{s_p}} \tag{1}$$

Let the multi-view stock price vector of stock s_p be denoted as $\boldsymbol{V}_t^{s_p} = \left[\boldsymbol{v}_{t-T+1}^{s_p}, \boldsymbol{v}_{t-T+1}^{s_p}, ..., \boldsymbol{v}_t^{s_p}\right] \in R^{T\times D}$, where $\boldsymbol{v}_t^{s_p}$ represents the multi-view stock price vector of stock s_p on the t-th trading day. D is the dimension of the stock feature vector, and T is the length of the input sequence. Then, for N stocks in the set S, their price vectors on the t-th trading day are represented as $\boldsymbol{X}_t = \left[\boldsymbol{V}_t^{s_1}, \boldsymbol{V}_t^{s_2}, ..., \boldsymbol{V}_t^{s_N}\right] \in R^{N\times T\times D}$, which serve as the input of the model. The output of the model is the return rate $\hat{\mathbf{r}}^{t+1} = \{\hat{r}_{t+1}^{s_1}, \hat{r}_{t+1}^{s_2}, \ldots, \hat{r}_{t+1}^{s_N}\}$ of N stocks on the $t+1$ trading day, where $\hat{r}_{t+1}^{s_1}$ represents the predicted return rate of stock s_1 on the $t+1$ trading day. The stock with the highest predicted return rate is selected for investment according to the predicted return rate. In the experiment, we partitioned the data into training set D^{train}, validation set D^{valid}, and test set D^{test}. The goal of the model learning is:

$$min \sum_{d\in D^{\text{valid}}} l\left(\hat{\boldsymbol{r}}^{\text{d}}, \boldsymbol{r}^{\text{d}}\right) \tag{2}$$

where l is the loss function, which will be introduced in Section 3.5.

3.2. Stock temporal feature extraction

For a stock $s_p, p = 1, 2, \cdots, N$, we have its initial feature vector $\mathbf{v}_i^{s_p}$ at day i:

$$\mathbf{v}_i^{s_p} = [p_i^{s_p}, MA@5, MA@10, MA@20, MA@30, PE_i] \tag{3}$$

where $p_i^{s_p}$ represents the closing price, and $MA@5$, $MA@10$, $MA@20$, and $MA@30$ represent the 5-day, 10-day, 20-day, and 30-day moving averages, respectively. As the order of time series data is essential information, we introduce a one-dimensional positional encoding PE_i to indicate the position of the data in the sequence, in order to better preserve the temporal dependencies between the data points. Absolute positional encoding is a common encoding method, but its large value fluctuations can interfere with the model's performance. Normalized positional encoding is more practical in series analysis to address this issue. Although it has a length-dependent problem, since the input series used in our experiments are fixed length, the impact of this problem on series analysis can be ignored. As a result, we normalize the absolute positional encoding and use it as the series position encoding, as follows:

$$PE_i = \frac{i}{T}, i \in [1, 2, 3, ..., T] \tag{4}$$

We obtain the stock price series data using a sliding window approach, with a series input length of T. The time dependencies of the series are learned by gated recurrent units (GRU). The hidden state on trading day i is given by the following equation:

$$h_i^{s_p} = GRU(v_i^{s_p}, h_{i-1}^{s_p}), t - T \le i \le t \tag{5}$$

where $h_{i-1}^{s_p}$ represents the hidden state vector of the day i. Due to the strong temporal dependencies present in time series data, the position of the data in the series is crucial. As a variant of RNN, GRU can capture long-term dependencies in series. It uses gate mechanisms to some extent to alleviate the problem of vanishing gradients. In particular, when combined with position encoding, it can capture series positional information more effectively. Therefore, we take the series within the current time window as the input to the GRU and use the last hidden state $h_t^{s_p}$ as the sequential embedding $e_t^{s_p}$ of the stock s_p. Finally, we obtain sequential embeddings of all stocks:

$$\boldsymbol{E}^t = GRU(\boldsymbol{X}_t) \tag{6}$$

where $\boldsymbol{E}^t = \left[\boldsymbol{e}_t^{s_1}, \boldsymbol{e}_t^{s_2}, ..., \boldsymbol{e}_t^{s_N}\right] \in R^{N\times T\times U}$ represents the sequential embeddings of all stocks, and U represents the dimension of the embeddings, i.e., the number of GRU hidden units.

3.3. Corporate relation graph router

The uncertainty in the stock market gives rise to dynamic changes in corporate relations, which means that the relation graph that is effective for learning relational features for a particular stock also changes dynamically over time. For example, the COVID-19 pandemic triggered a surge in stock prices in industries such as e-commerce and healthcare, and short-term relation concepts related to the pandemic dominated the capturing of correlation relationships of relevant stocks. Pre-defined corporate relationship data plays a relatively small role in stock price prediction at the moment. A dynamically capturing more effective corporate relationships in real-time is therefore one of the crucial factors to improving the $DR\text{-}GAT$ model's accuracy. We propose a Relation Graph Router (RGR) that automatically routes the sequential embeddings of stocks to an effective corporate relation graph (short-term or long-term), as shown in Fig. 1. With RGR, the level of volatility in stock prices is detected by frequency-domain analysis, and the sequential embeddings of the stocks are routed to an effective corporate relation graph based on the volatility level.

During the period of stable stock price trends, prior knowledge that has long existed can be invaluable for learning the relational features and predicting the next trading day's stock price. The long-term corporate relation graph based on prior knowledge can effectively capture the relational features that currently have a significant impact on the stock price now. However, during periods of stock price fluctuations triggered by sudden events, the predefined long-term corporate relation graph is not as effective at capturing relational features. The dynamically constructed short-term corporate relational graph will better capture latent relationships

between stocks at this time. The current stock price volatility can therefore be used to select the corporate relational graph. The construction process of the long-term and short-term corporate relational graphs will be discussed in Section 3.4. In order to obtain the level of stock price volatility, this paper proposes a method of oscillation detection using frequency domain analysis.

Frequency Domain Information Acquisition. The discrete wavelet transform (DWT) assists in the localized analysis of time series and describes the local characteristics of time-domain and frequency-domain signals. DWT can simultaneously decompose a sequence in both time and frequency domains [24], and the frequency components obtained from the decomposition can retain the time information of the original sequence. This characteristic makes DWT widely used in the processing of time-dependent stock price sequences. Lahmiri [25] integrated DWT and backpropagation neural network (BPNN) for stock prediction. Jothimani et al. [26] combined DWT with regression vector machine and artificial neural networks for stock price sequence prediction. Liu et al. [27] used DWT for stock behavior analysis at different scales and achieved good results in stock trend prediction. These studies demonstrate that DWT performs well in processing discrete stock price sequences. In DWT, the selection of different wavelet functions will result in different decomposition results. Since Haar wavelet function is more effective than other wavelet functions at capturing fluctuations between adjacent observations [28,25], it is widely used in time series analysis. This article uses the Haar wavelet function.

In the process of series decomposition, we adopt a sliding window to select the price series for DWT. For a given stock s_p and a price series $\mathbf{P}_t^{s_p} = \{p_{t-T}^{s_p}, p_{t-T+1}^{s_p}, ..., p_t^{s_p}\}$ with window size T, we obtain the low-frequency component $\mathbf{a}_t^{s_p}$ and high-frequency component $\mathbf{d}_t^{s_p}$ through Low-Pass Filter (LPF) and High-Pass Filter (HPF).

Routing path selection. An analysis of low-frequency and high-frequency sequences shows that the high-frequency component of a stock price series with stable trends fluctuates near zero after decomposition. In contrast, when the stock price fluctuates more, the high-frequency component also fluctuates more, and its variance is significantly larger than that of the stable trend series. Thus, compared with the stable trend series, the difference between the variance of the high-frequency component and the low-frequency component is greater for the oscillating stock price series after decomposition. Based on the above analysis, for any stock s_p, we identify oscillations using the following formula:

$$VAR(\boldsymbol{d}_t^{s_p}) - VAR(\boldsymbol{a}_t^{s_p}) > \varepsilon \tag{7}$$

where $VAR()$ is the variance function and ε is the preset threshold, which is set to -0.0002 in the experiment. In the case of the price series of the target stock within a given window that satisfies the inequality above, the price of the target stock during this period is viewed as oscillating, and the short-term corporate relation graph is more effective in capturing the relational features of the target stock. Conversely, the long-term corporate relation graph is better at capturing its relational features if the above inequality is not satisfied. Therefore, the dominant corporate relation graph is selected for the target stock based on the degree of series volatility, and RGR routes the sequential embeddings of the stock to an effective corporate relationship graph, thereby achieving dynamic correlation learning for stock attributes at the spatial level.

3.4. Corporate relational feature extraction

Data of relevant stocks can help predict stock price changes [14,29]. This section discusses how to model this relationship and use it to improve the model's predictive ability. As stocks' correlations are dynamic, it is not sufficient to rely solely on known priors to establish a static relationship graph. Dynamic relevance degrees are necessary for learning relational features. We divide the corporate relation graph into two categories: the long-term relation graph that displays long-term existing relationship and the short-term relation graph created based on the short-term stock price status. Our goal is to dynamically model stock relationship from both long-term and short-term perspectives in order to obtain deeper latent relationships. To achieve this, we establish a long-term corporate relation graph and a short-term corporate relation graph respectively based on a window unit and capture the dynamic evolution of corporate relationships through window sliding. The long-term corporate relation graph describes the long-term existing corporate relationships, whereas the short-term corporate relation graph represents the relationship that are established under short-term shocks. The process of constructing the corporate relation graph is shown in Fig. 2. Following is a detailed introduction to how two types of corporate relation graphs can be built.

Short-term relation modeling. Short-term high volatility in stock prices is often attributed to factors such as national policies, sudden social events, and changes in the overall market environment. Due to the uncertainty of these factors, the advantage of prior knowledge of corporate relationships in modeling corporate relationships is further weakened. Dealing with the aggregation of relational features in the presence of uncertainty is a challenging task. According to the Efficient Market Hypothesis [30], all relevant information concerning stock prediction is reflected in prices, and the impact of uncertain factors on stocks is also manifested in price changes without exception. Wang et al. [31] also pointed out that analyzing stock price correlations is the primary method for studying stock relationships, and through empirical evidence, they demonstrated that stock price similarity can reflect the correlation between stocks. Therefore, in the task of learning relational features, we focus on stocks that exhibit similar changes in stock prices. In the periods of stock market volatility, the effectiveness of prior relationship data in stock relationship modeling may be diminished by the impact of unforeseen events. Based on the Efficient Market Hypothesis, we construct the corporate relation graph starting from stock price similarity. Some works [32] obtain related companies by calculating the cosine similarity of stock price feature vectors. While Cosine similarity is commonly used to compute vector distances, it has limitations in evaluating the similarity of time series data because it ignores their temporal characteristics. A better approach [33] results in better stock similarity by combining the adjusted Pearson correlation coefficient and the Euclidean distance of cumulative returns. However, the Pearson correlation coefficient is only suitable for linearly correlated variables and is greatly affected by outliers. In addition, measuring stock price

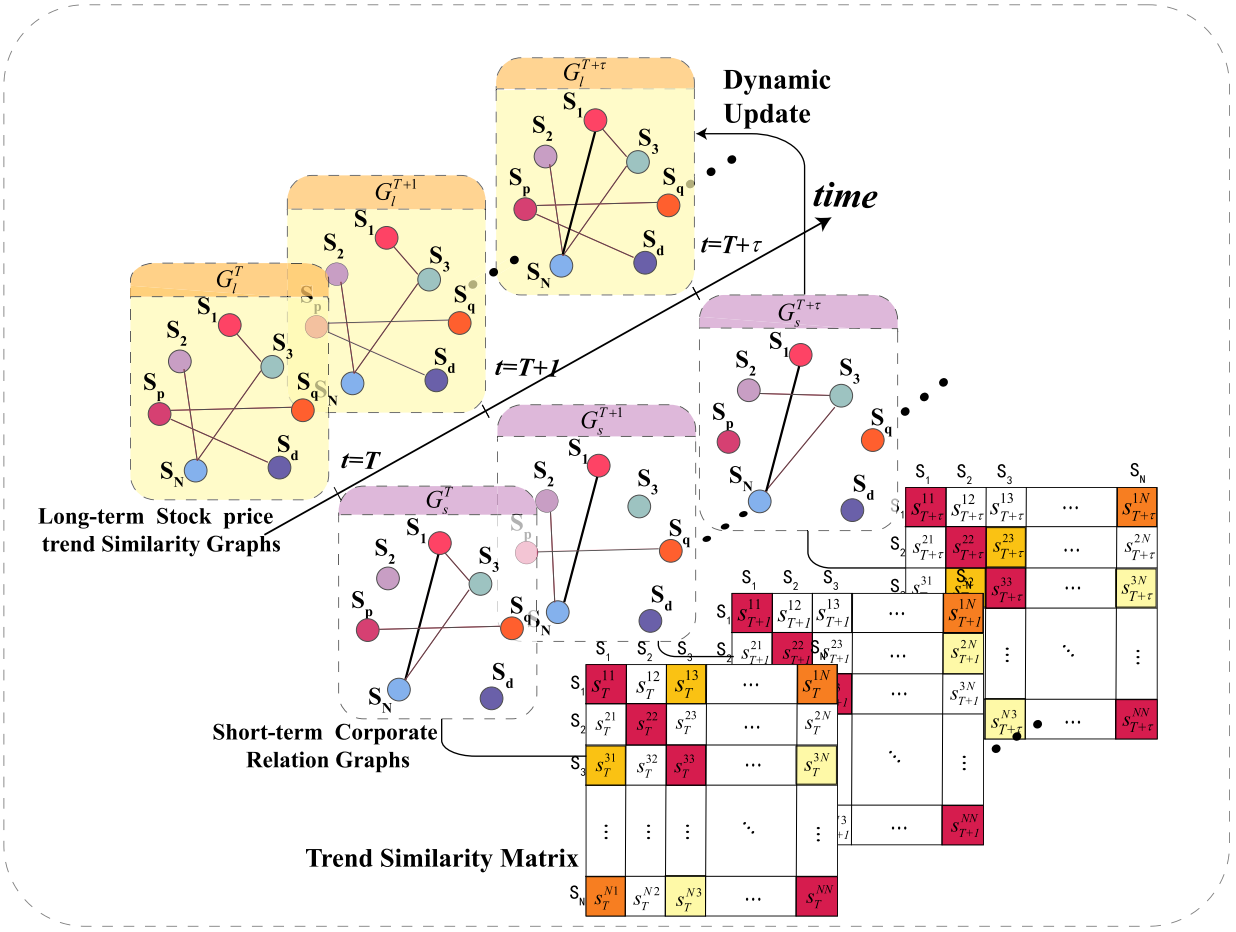


Fig. 2. Construction process of long-term stock price trend similarity relation graph and short-term corporate relation graph.

similarity using the Euclidean distance of cumulative returns is undoubtedly coarse-grained. To overcome the shortcomings of existing measures of stock price similarity, we propose a new measure of similarity, namely stock price trend similarity. During periods of stock market volatility, we construct a corporate relation graph using stock price trend similarity.

The short-term corporate relation graph is constructed based on the short-term stock price trend similarity. The construction process consists of two main steps: first, calculating the stock price trend similarity between stocks to obtain a similarity matrix; then, based on the similarity matrix, connecting stocks with high similarity through edges to establish associations in the graph. In order to calculate the similarity of stock price trends, the stocks are first classified according to their trends. Given a certain stock s_p, its price series within a window is encoded to obtain the encoding series $Y_t^{s_p} = \left\{ y_{t-T+1}^{s_p}, y_{t-T+2}^{s_p}, ..., y_t^{s_p} \right\}$, using the following encoding method:

$$y_{i+1}^{s_q} = \begin{cases} 1 & p_{i+1}^{s_q} - p_i^{s_q} > \delta \\ -1 & p_{i+1}^{s_q} - p_i^{s_q} < \delta \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

In this study, the threshold value δ, which determines whether the stock price moving trend is upward, downward, or stationary, is set to 0.0001. For any two stocks s_p and s_q in a set S, the similarity of their price trends is determined by calculating the proportion of days with the same price trend within a sliding window. Specifically, the encoding sequences $Y_t^{s_p} = \left\{ y_{t-T+1}^{s_p}, y_{t-T+2}^{s_p}, ..., y_t^{s_p} \right\}$ and $Y_t^{s_q}$ of s_p and s_q are compared bit by bit, and if they are the same, it means that the price moving trends of the two stocks are the same on that trading day. The final similarity score between s_p and s_q, denoted as S_t^{pq}, is calculated by dividing the number of days with the same price trend by the length of the encoding sequence. For example, when the window length is 4 and the encoding sequences of two stocks are 1,0,0,-1 and 1,0,1,0, the first two trading days have the same trend, resulting in a similarity score of 0.5.

Using a sliding window approach, we calculate the similarity between any two stocks within each window, resulting in a similarity matrix of stock price trends. We then select stocks with high trend similarity to construct the short-term corporate relation graph for the current window. Let $G_s^t = (S, E^t)$ denote the short-term corporate relation graph within a given window, and let S denote the set

of all stocks. For any stock $s_p \in S$, which has a high similarity in price trend with another stock s_q in G_s^t, s_p and s_q are connected by an edge $\langle s_p, s_q \rangle \in E^t$. Fig. 2 visualizes the similarity matrix of stock price trends, where the depth of color represents the magnitude of similarity values. With the passage of time, the stock price trend similarity matrix is dynamically constructed, and subsequently, the short-term corporate relation graph is dynamically built. For instance, in the visualized similarity matrix of stock price trends shown in Fig. 2, at time T, stocks s_1 and s_3 have a dark color at the corresponding position, indicating a high similarity in trend, so they are connected by an edge in the short-term enterprise relationship graph. However, at time $T+1$, the similarity between them decreases, and they are no longer adjacent nodes in the short-term corporate relation graph. The sequential embedding $e_t^{s_p}$ obtained by the temporal feature extraction module is used as the initial feature vector for s_p in G_s^t. Graph Attention Network (GAT) [20] is adopted as the graph neural network for learning the relational features. As the strength of the correlations among stocks varies, for each node in the graph, GAT assigns different weights based on the features of its neighbors, thereby selectively aggregating the relational features. Another advantage of GAT is that the computation of attention weights and aggregation of relational features are achieved by traversing nodes. As the operation process only considers the features of adjacent nodes, it is more flexible to adapt to changes in graph structure.

Given a relation graph G_s^t, the mathematical representation for aggregating the features of associated nodes to the target node s_p in G_s^t using GAT is as follows:

$$GAT(G_s^t; s_p) = ReLU\Big(\sum_{s_q \in \Gamma(s_p)} \beta_{pq} \boldsymbol{W}_1 e_t^{s_q}\Big) \tag{9}$$

where $\Gamma(s_p)$ denotes the set of neighboring nodes of node s_p in graph G, W_1 is a weight matrix that can be learned, and β_{pq} represents the attention weight between node s_p and node s_q in G:

$$\beta_{pq} = \frac{\exp\Big(Leaky\,\mathrm{ReLU}\Big(\boldsymbol{r}^\top \Big[\boldsymbol{W}_2 e_t^{s_p} \| \boldsymbol{W}_2 e_t^{s_q}\Big]\Big)\Big)}{\sum_{s_q \in \Gamma(s_p)} \exp\Big(Leaky\,\mathrm{ReLU}\Big(\boldsymbol{r}^\top \Big[\boldsymbol{W}_2 e_t^{s_p} \| \boldsymbol{W}_2 e_t^{s_q}\Big]\Big)\Big)} \tag{10}$$

where $.^T$ denote matrix transpose, $\|$ denote concatenation, W_2 be the learnable weight matrix, and $\boldsymbol{r}$ be the learnable vector that projects sequential embeddings to a scalar. By learning the short-term relation graph through GAT, we obtain the short-term relational embedding $\boldsymbol{g}_s^{s_p}(t) = GAT(G_s^t; s_p)$ of the current window sequence. $\boldsymbol{g}_s^{s_p}(t)$ is a short-term relational embedding obtained by attention-weighted means within a set of stocks with strong short-term correlations, reflecting how the stock s_p is influenced by adjacent companies in the short-term relation graph.

Long-term relation modeling. When the stock price fluctuates weakly, the long-term existing prior relationship knowledge is effective in capturing the relational features. However, due to limited ability to acquire prior knowledge and the difficulty representing implicit latent relationships with prior knowledge, it is inevitable that constructing a corporate relation graph solely based on prior knowledge will inevitably overlook some long-term potential corporate relations. Therefore, in the absence of severe short-term unexpected shocks to the stock, the stocks related to it can be explored from the following two perspectives. Firstly, there is mutual influence among stocks with interconnections in prior knowledge. In terms of industry data, such as the steel and aviation industries, stocks within the industry often exhibit cyclical changes following similar patterns. In addition, the inter-corporation relationship established by business logic is also a very valuable source of prior data, such as the shared "cloud computing" business concept between Amazon and Microsoft, and the shared "e-commerce" business concept with eBay. The fluctuations of their stock prices are often related due to the business association. Industry relationships and Wiki relationships [14] provide the aforementioned prior relationship information. Secondly, policies and social events that have sustained impact define potential long-term business relationships. For instance, the COVID-19 pandemic has led to the potential connections between industries, such as e-commerce and healthcare, which were both greatly impacted by the outbreak. These connections were initially reflected in high short-term stock price trend similarity. However, due to the prolonged duration of the pandemic, some stocks in the e-commerce and healthcare industries may continue to correlate in the future. Although the stock price volatility weakened during this period, the business association between them did not disappear but rather became a part of long-term relationships. Therefore, we construct the long-term corporate relation graph from two perspectives: first, by utilizing prior knowledge to build a relation graph, which includes connecting stocks belonging to the same industry to construct a fully connected graph, as well as connecting companies that have logical business relationships through Wiki data. Additionally, if the short-term correlation defined by the similarity of stock price trends persists over time, we consider this relationship as a long-term relationship that continuously affects stocks. In the same way, if the influence of this relationship weakens over time, we consider the stock to no longer be affected by this relationship. Thus, we construct the long-term corporate relation graph dynamically by adding or deleting edges based on the existence of this relationship.

According to the previous description, the long-term relation graph consists of three types of corporate relation graphs. Firstly, let $G_I = (S, E_I)$ denote the industry relation graph constructed based on industry information, where each connected component consists of all stocks belonging to the same industry, i.e., for any two stocks s_p and s_q in S, $\langle s_p, s_q \rangle \in E_I$ if and only if s_p and s_q belong to the same industry. Secondly, by utilizing the relation data from Wiki, we construct a business logic relation graph G_W for stocks, $G_W = (S, E_W)$, where for any two stocks s_p and s_q in S, $\langle s_p, s_q \rangle \in E_W$ if and only if the Wiki relation data defines the existence of business logic between them. Thirdly, due to the sustained impact of unexpected events on stocks, short-term relationships can evolve into long-term relationships. Similarly, the eventual attenuation of this impact can lead to the disappearance of this relationship. For example, as shown in Fig. 2, the relationship between s_1 and s_N that exists in the short-term relation graph persists in the long term, and thus this relationship is transformed into a part of the long-term stock price trends similarity relationship graph. We dynamically

construct a long-term stock price trend similarity relation graph based on the existence of short-term relationships. Initialization of the graph is an empty set, and as time passes, it is dynamically updated based on the existence of relationships within short-term relation graphs in multiple recent windows. Following is a mathematical representation of the adjacency matrix:

$$Adj(G_l^t) = Adj(G_s^{t-\tau+1}) \circ Adj(G_s^{t-\tau+2}) \circ \cdots \circ Adj(G_s^t) \tag{11}$$

where $Adj(G)$ represents the adjacency matrix of graph G, $\circ$ represents the Hadamard product, τ is a hyperparameter that defines the time limit for the transition from short-term relationships to long-term relationships, and is set to 45 days in the experiment. In other words, the long-term stock price trend similarity graph is constructed by looking back from the current window and retrospectively examining the short-term stock price trend similarity graph within each window, going back τ windows. When an edge persists over a long period of time, it is transformed into an edge in the long-term stock price trend similarity graph. The initial feature vector of s_p in G_l^t is its sequential embedding $\boldsymbol{e}_t^{s_p}$. When capturing potential relationships between stocks using long-term relationships, we once again use GAT. The long-term relational embedding generated by GAT is given by the following formula:

$$\boldsymbol{g}_I^{s_p}(t) = GAT(G_I^t; s_p) \tag{12}$$

$$\boldsymbol{g}_W^{s_p}(t) = GAT(G_W^t; s_p) \tag{13}$$

$$\boldsymbol{g}_l^{s_p}(t) = GAT(G_l^t; s_p) \tag{14}$$

$$\boldsymbol{g}_L^{s_p}(t) = \boldsymbol{W}_l \left[\boldsymbol{g}_I^{s_p}(t) \,||\, \boldsymbol{g}_W^{s_p}(t) \,||\, \boldsymbol{g}_l^{s_p}(t) \right] + b \tag{15}$$

The embeddings $\boldsymbol{g}_I^{s_p}(t)$, $\boldsymbol{g}_W^{s_p}(t)$, and $\boldsymbol{g}_l^{s_p}(t)$ are generated based on three different types of long-term corporate relationships. The parameters $\boldsymbol{W}_l$ and b are learnable parameters of the model. The resulting embedding $\boldsymbol{g}_L^{s_p}(t)$ encodes how stocks with long-term relationships interact with each other using different attention weights.

Algorithm 1: Algorithm for $DR-GAT$.

Input: All stock initial feature vector $\boldsymbol{V}_t^{s_p} = \left[\boldsymbol{v}_{t-T+1}^{s_p}, \boldsymbol{v}_{t-T+1}^{s_p}, ..., \boldsymbol{v}_t^{s_p} \right] \in R^{T \times D}$

Output: The vector of predicted return rate $\hat{\boldsymbol{r}}^{t+1} = [\hat{R}_{t+1}^{s_1}, \hat{R}_{t+1}^{s_2}, ..., \hat{R}_{t+1}^{s_N}]$

/* step 1:stock features extraction */

Obtain sequence embedding E_t by Equation (6); /* step 2:relation graph router */

for $i \leftarrow 1$ **to** N **do**

  /* discrete wavelet transform */

  $a^{s_i} \leftarrow HPF(\boldsymbol{P}^{s_i})$;

  $d^{s_i} \leftarrow LPF(\boldsymbol{P}^{s_i})$;

  /* Route target relation graph judgment. */

  if a^{s_i} *and* d^{s_i} *satisfy Equation (7)* **then**

    Use stock price trend similarity to construct the dynamic short-term relation graph G_s;

    Add the G_s to G as the target relation graph of s_i;

  else

    Use stock price trend similarity and prior knowledge to construct the dynamic short-term relation graph G_L;

    Add the G_L to G as the target relation graph of s_i;

  end

end

/* step 3:relation features extraction */

Obtain the relational embedding by Equation (9) and Equation (10);

/* step 4:model learning */

Equation (16) and Equation (17) are used for feature fusion of sequential embeddings and relational embeddings;

Obtain the predicted return rate vector $\hat{\boldsymbol{r}}^{t+1}$ by Equation (18);

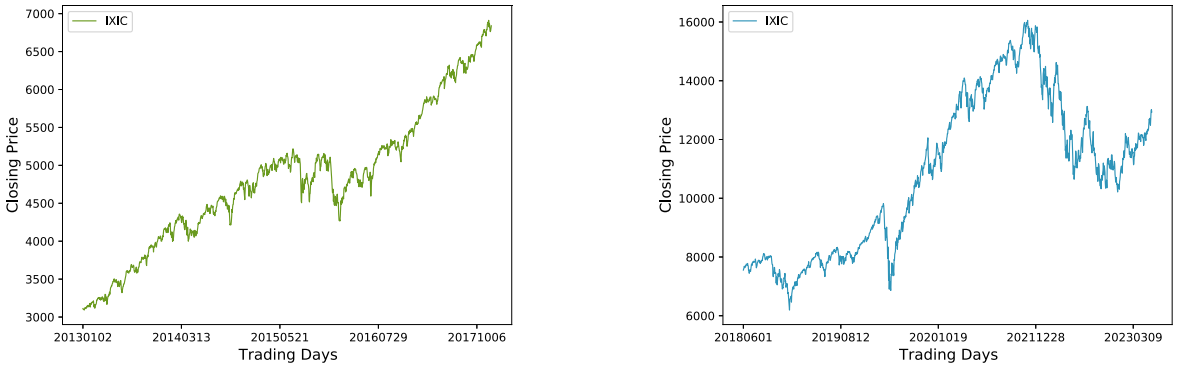
3.5. Model learning

Embedding Fusion. We predict the return rate of stocks by using the initial stock price features and corporate relational features. The derived feature vectors include the initial stock price sequential embedding $\boldsymbol{e}_t^{s_p}$ and the corporate short-term relational embedding $\boldsymbol{g}_s^{s_p}(t)$, as well as the corporate long-term relational embedding $\boldsymbol{g}_L^{s_p}(t)$. Based on the results of dynamic routing, the final embedding vector $\boldsymbol{f}_t^{s_p}$ is obtained by fusing the three embeddings using the following formula. When the stock is routed to the long-term relation graph attention network, it satisfies:

$$\boldsymbol{f}_t^{s_p} = LeakyReLU\left(\left[\boldsymbol{e}_t^{s_p} \,||\, \boldsymbol{g}_L^{s_p}(t) \right] \boldsymbol{W}_f \right) \tag{16}$$

When the stock is routed to the short-term relation graph attention network, it satisfies:

$$\boldsymbol{f}_t^{s_p} = LeakyReLU\left(\left[\boldsymbol{e}_t^{s_p} \,||\, \boldsymbol{g}_s^{s_p}(t) \right] \boldsymbol{W}_f \right) \tag{17}$$



(a) 2013-2017 (b) 2018-2023

Fig. 3. NASDAQ index trend.

where $\boldsymbol{W}_f$ is a learnable weight matrix and LeakyReLU is the activation function. Given a window of data containing the trading day $t-T+1$ to the trading day t, a final embedding vector $\boldsymbol{f}_t^{s_p}$ is generated to predict the return rate $\hat{\boldsymbol{r}}^{t+1}$ of the trading day $t+1$.

Network Optimization. Let $\boldsymbol{F}^t = [\boldsymbol{f}_t^{s_1}, \boldsymbol{f}_t^{s_2}, \cdots, \boldsymbol{f}_t^{s_N}] \in R^{N\times U}$ and U represent the size of the embedding (i.e., U is the number of hidden units in GRU). Then, the return rate $\hat{\boldsymbol{r}}^{t+1}$ on the trading day $t+1$ predicted for all stocks in S is given by the following formula:

$$\hat{\boldsymbol{r}}^{t+1} = \boldsymbol{F}^t \boldsymbol{W}_U + \boldsymbol{b} \tag{18}$$

where $\boldsymbol{W}_U \in R^U$ denotes a trainable weight matrix, while $\boldsymbol{b} \in R^N$ represents a bias vector. In model learning, the mutual comparison between stocks cannot be ignored to achieve profitable stock recommendation. We utilized the combination of pointwise regression loss and pairwise ranking-aware loss to form the loss function [14] of DR-GAT. Pointwise regression can minimize the difference between predicted and actual returns during the training process. The pairwise ranking-aware loss ensures that stocks with higher expected returns are ranked higher. The combined loss function is given as follows:

$$\begin{aligned} l\left(\hat{\boldsymbol{r}}^{t+1}, \mathrm{r}^{t+1}\right) &= \left\|\hat{\mathrm{r}}^{t+1} - \mathrm{r}^{t+1}\right\|^2 + \\ \alpha \sum_{i=0}^{N}\sum_{j=0}^{N} \max\left(0, -\left(\hat{r}_i^{t+1} - \hat{r}_j^{t+1}\right)\left(r_i^{t+1} - r_j^{t+1}\right)\right) &+ \lambda\|\Theta\|^2 \end{aligned} \tag{19}$$

where $\boldsymbol{r}^{t+1} = [R_{t+1}^{s_1}, R_{t+1}^{s_2}, ..., R_{t+1}^{s_N}]$ denote the actual return rate of all stocks on the trading day $t+1$, and let α denote the weighting parameter. Θ depicts all learnable weights, and λ is the regularization hyperparameter to prevent overfitting. The algorithm flow of DR-GAT model is shown in Algorithm 1.

4. Experiments

We conduct a series of experiments to answer the following four evaluation questions.

- ***EQ*1**:How does the proposed DR-GAT perform in stock recommendation compared to state-of-the-art stock recommendation methods, and how can it be used to guide investment?
- ***EQ*2**: Does each component of DR-GAT improve the effectiveness of stock recommendation?
- ***EQ*3**:How do hyperparameters affect the recommendation performance of DR-GAT?
- ***EQ*4**:What information is captured by the attention weights when DR-GAT aggregates stock relational features?

4.1. Evaluation settings

Datasets. We employed two datasets from real financial markets: NASDAQ and NYSE. To comprehensively evaluate the robustness of our model under different market conditions, we conducted experiments using datasets from two distinct time periods. As shown in Fig. 3, the period from 2013 to 2017 demonstrates an overall upward trend in the NASDAQ index, indicating a bullish market for US stocks. In contrast, the period from June 2018 to June 2023 exhibits both prolonged bullish and bearish intervals in the NASDAQ index, encompassing both bull and bear market states. Therefore, by testing the performance of our model during these two time periods, we can effectively demonstrate its adaptability to real-world scenarios. For the first time period, we collected historical price data, industry relation data, and Wiki relation data [14] for 1,026 NASDAQ stocks and 1,737 NYSE stocks, spanning from February 1st, 2013 to August 12th, 2017, as well as dividing the stock price series data into three time periods for training (2013-2015), validation (2016), and evaluation (2017) in order to conduct stock prediction work. For the second time period, after removing delisted and long-suspended stocks, we obtained a total of 724 NASDAQ stocks and 1,227 NYSE stocks. The time span

Table 1
Statistics of the sequential data.

Market	Stocks#	Training Days#	Validation Days#	Testing Days#
NASDAQ	1026	756 01/02/2013 12/31/2015	252 01/04/2016 12/30/2016	237 01/03/2017 12/08/2017
NYSE	1737	756 01/02/2013 12/31/2015	252 01/04/2016 12/30/2016	237 01/03/2017 12/08/2017
NASDAQ	724	754 06/01/2018 05/28/2021	253 06/01/2021 05/31/2022	250 06/01/2022 05/31/2023
NYSE	1227	754 06/01/2018 05/28/2021	253 06/01/2021 05/31/2022	250 06/01/2022 05/31/2023

of the dataset covers from June 1, 2018, to May 31, 2023. Similarly, we used the first three years of data as the training set, and the subsequent two years were divided into the validation and test sets. During the stock prediction process, we organized the data through a sliding window of 15 days, using the 16th day as the prediction target. We summarize the collected data in Table 1.

Evaluation Metrics. The Mean Squared Error (MSE) metric is widely used to evaluate regression tasks, thus we compute the MSE of the stock returns within each window. The Mean Reciprocal Rank (MRR) is a common metric used in recommendation systems, and we use it to evaluate ranking performance. The Investment Return Ratio (IRR) is obtained by summing up the returns of the selected stocks for each testing day, which directly reflects the effectiveness of stock investment. It is the most direct and effective metric for measuring the profitability of the model, thus IRR is our most important metric. Smaller MSE (≥ 0), larger MRR ([0,1]), and IRR values indicate better model performance.

Let $\mathbf{r}^t = [R_t^{s_1}, R_t^{s_2}, ..., R_t^{s_N}]$ represent the true return on the trading day t for all stocks in S, where $R_t^{s_p}$ is the true return on stock s_p. The predicted return on stock s_p is represented as $\hat{R}_t^{s_p}$. Let $L@K(\hat{R}_t^S)$ represent the top-K ranked stocks in the prediction. We aim to formulate the optimal investment strategy that has the best return when investing in the top-1 predicted stock. Therefore, we evaluate the performance of the model by computing the MRR and IRR of the top-1 stock predicted, as well as the MSE.

- ***Mean Reciprocal Rank*(*MRR*)**:

$$MRR = \sum_{s_p \in L@1(\hat{R}_t^S)} \frac{1}{\text{rank}\left(R_t^{s_p}\right)} \tag{20}$$

  where $rank(R_t^{s_p})$ represents the true return rate ranking of stock s_p on the trading day t.
- ***Investment Return Ratio*(*IRR*)**:

$$\text{IRR}^t = \sum_{s_p \in L@1(\hat{R}_i^S)} \sum_{i=0}^{t-1} R_i^{s_p} \tag{21}$$

- ***Mean Squared Error*(*MSE*)**:

$$\text{MSE}^t = \frac{1}{N} \sum_{s_p \in S} \left(\hat{R}_t^{s_p} - R_t^{s_p}\right)^2 \tag{22}$$

We evaluate the performance of the model on the test set and take the average of MRR and MSE for each day on the test set as the final MRR and MSE of the model.

Baselines. We will compare our proposed DR-GAT method with nine baseline methods, including current state-of-the-art serieslearning methods and graph-based relation learning methods, as listed below:

- ***LSTM*** [6]:The vanilla LSTM was employed here, which utilizes historical sequential data to obtain sequential embeddings, and then employs fully connected layers for predicting returns. The model optimization is achieved through regression loss.
- ***SFM*** [34]:SFM combines the Discrete Fourier Transform (DFT) with LSTM to obtain frequency domain information from historical price sequences via DFT, learn frequency-aware sequential embeddings through $LSTM$, and then predict returns through fully connected layers.
- ***Rank-LSTM*** [14]:Rank-LSTM is based on $LSTM$ for predicting stock returns, and its model optimization is achieved through pointwise regression loss and pairwise ranking-aware loss.
- ***Rank-GRU***:Rank-GRU is based on GRU for predicting stock returns, and its model optimization is achieved through pointwise regression loss and pairwise ranking-aware loss.

Table 2
Main experimental results on the 2013-2017 dataset.

Model	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
LSTM	$3.81e-4\pm2.20e-6$	$3.64e-2\pm1.04e-2$	0.13 ± 0.62	$3.81e-4\pm9.30e-5$	$4.82e-2\pm4.95e-3$	0.49 ± 0.47
SFM	$5.20e-4\pm5.77e-5$	$2.33e-2\pm1.07e-2$	-0.25 ± 0.52	$3.81e-4\pm9.30e-5$	$4.82e-2\pm4.95e-3$	0.49 ± 0.47
Lstm_RANK	$3.79e-4\pm1.11e-6$	$4.17e-2\pm7.50e-3$	0.68 ± 0.60	$2.28e-4\pm1.16e-6$	$3.79e-2\pm8.82e-3$	0.56 ± 0.68
GRU_RANK	$3.79e-4\pm1.45e-6$	$4.59e-2\pm2.71e-3$	0.90 ± 0.29	$2.83e-4\pm3.67e-5$	$4.61e-2\pm2.02e-4$	0.74 ± 0.48
RSR_E	$3.80e-4\pm7.20e-7$	$3.94e-2\pm8.15e-3$	0.81 ± 0.85	$2.29e-4\pm2.77e-6$	$4.28e-2\pm6.18e-3$	1.00 ± 0.58
RSR_I	$3.79e-4\pm6.60e-7$	$4.09e-2\pm5.18e-3$	1.19 ± 0.55	$2.26e-4\pm5.30e-7$	$4.51e-2\pm2.41e-3$	1.06 ± 0.27
FinGAT	$1.70e-3\pm6.34e-7$	$4.09e-2\pm2.40e-5$	0.67 ± 0.15	$3.76e-4\pm7.56e-10$	$5.19e-2\pm3.44e-6$	0.66 ± 0.02
TRAN	$\mathbf{3.79e-4\pm3.90e-7}$	$3.81e-2\pm4.37e-3$	0.92 ± 0.25	$2.26e-4\pm2.30e-7$	$4.91e-2\pm4.82e-3$	1.38 ± 0.35
STHAN	$1.45e-3\pm1.35e-4$	$4.83e-2\pm4.78e-3$	1.10 ± 0.19	$2.93e-4\pm2.13e-6$	$4.35e-2\pm7.46e-5$	0.61 ± 0.23
DR-GAT	$1.21e-3\pm1.13e-8$	$\mathbf{5.13e-2\pm9.29e-6}$	$\mathbf{1.52\pm0.01}$	$\mathbf{2.26e-4\pm7.04e-15}$	$\mathbf{6.32e-2\pm6.98e-5}$	$\mathbf{1.62\pm0.01}$

- ***RSR_E*** [14]:RSR_E is a stock ranking prediction model based on stock correlation relationships, which incorporates relational embeddings on top of Rank-LSTM by explicitly modeling the relational features.
- ***RSR_I*** [14]:RSR_I is a variant of RSR_E, with its relational embeddings implemented through implicit modeling.
- ***FinGAT*** [15]:FinGAT is a stock ranking prediction model based on the correlation relationships among stocks, which models the hierarchical relationships of stocks using prior knowledge. It employs graph attention network (GAT) to obtain the relational embeddings for stock ranking prediction, and is optimized with pointwise regression loss and pairwise ranking-aware loss.
- ***TRAN*** [35]:TRAN is a stock ranking prediction model based on stock correlation relationships. It utilizes historical series features and stock document features to obtain relational features through interactions, and thus predicts and ranks stock returns.
- ***STHAN − SR*** [29]:STHAN-SR is a stock ranking prediction model based on stock correlation relationships. It models stocks of different types and degrees into hypergraphs and uses a hypergraph attention network to learn related features, thereby predicting stock rankings.

Settings of DR-GAT. We conduct all experiments on an Nvidia GeForce RTX 3090 GPU. The hidden layer dimensions of GRU and GAT are set to 32. Learning is done through dynamically adjusted learning rates, and the weight α of the ranking loss is adjusted within [0.1, 1, 10, 100, 500]. The final strategy selects the top 1 predicted stock for investment. The model is optimized using the Adam optimizer. We report the average test performance when the model achieves the best performance on the validation set in 10 trials.

4.2. Experimental results

Main Results. Table 2 presents the main experimental results obtained on the dataset from 2013 to 2017. DR-GAT achieves the best results on the NASDAQ and NYSE markets compared with the current state-of-the-art models. Ranking-based models, such as Rank-LSTM and Rank-GRU, outperform regression-based models, such as SFM and LSTM. This result suggests that optimizing the model using pairwise ranking-aware loss results in better performance. Additionally, graph-based relationship learning models, such as TRAN and STHAN, are more profitable than sequence-based learning methods, indicating that the relational features are effective in predicting stock returns. As the most important indicator in stock recommendation, our model's IRR value is significantly better than all baselines. Compared to the current state-of-the-art method TRAN, our model achieved a 65.22% improvement in IRR on the NASDAQ dataset and a 17.39% increase in IRR on the NYSE dataset. The MRR indicator used to evaluate ranking effectiveness is also significantly better than all baselines on both datasets. For the MSE indicator, our model achieved the best results on the NYSE dataset. Our model outperforms current graph-based relationship learning models, indicating that both the dynamic routing approach we have adopted in learning the relational features of stocks and the non-reliance solely on prior knowledge are highly effective.

Table 3 presents the main experimental results obtained on the dataset from 2018 to 2023. Under unstable market conditions, our model outperformed all the baselines. Compared to the best-performing baseline, FinGAT, our model achieved a 65.57% increase in IRR on the NASDAQ dataset and a 15.8% increase in IRR on the NYSE dataset. For the MRR metric, our model improved by 13.37% on the NASDAQ dataset and 5.99% on the NYSE dataset. Backtesting the return rates of recommended stocks to simulate the investment return effect of the model. As shown in Fig. 4, the IRR curves of all models exhibited fluctuations, which can be attributed to the high risk involved in selecting top1 investments. However, DR-GAT consistently delivered higher returns compared to other models.

In testing on two real-market datasets across two different time periods, DR-GAT consistently demonstrated outstanding performance. This further validates the stability and robustness of our model under diverse market conditions.

Time Analysis DR-GAT selects Graph Attention Network (GAT) for learning relational features in graph neural networks, which not only adapts to changes in the graph structure but also makes computations more efficient. A GAT calculates attention weights between nodes only considering neighbors rather than global nodes. Because of this locality, GAT can reduce computational costs and

Table 3
Main experimental results on the 2018-2023 dataset.

Model	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
LSTM	$1.88e-3\pm3.74e-7$	$6.86e-2\pm5.37e-5$	2.18 ± 0.30	$3.20e-3\pm1.40e-5$	$7.03e-2\pm1.47e-4$	7.31 ± 10.05
SFM	$8.82e-2\pm1.01e-3$	$6.02e-2\pm8.08e-5$	2.04 ± 0.07	$8.71e-3\pm1.52e-5$	$6.86e-2\pm8.22e-6$	7.97 ± 0.03
Lstm_RANK	$1.98e-3\pm2.04e-6$	$6.95e-2\pm1.34e-4$	2.49 ± 0.93	$2.63e-3\pm2.30e-6$	$6.92e-2\pm7.48e-5$	7.36 ± 9.71
GRU_RANK	$\mathbf{1.53e-3\pm3.85e-8}$	$7.58e-2\pm2.20e-4$	2.60 ± 0.91	$3.43e-3\pm6.66e-6$	$7.54e-2\pm5.63e-5$	7.64 ± 11.69
RSR_E	$1.53e-1\pm3.10e-3$	$6.10e-2\pm3.04e-6$	2.02 ± 0.08	$3.34e-2\pm2.20e-4$	$6.96e-2\pm3.87e-5$	7.02 ± 10.41
RSR_I	$1.50e-1\pm2.33e-3$	$5.49e-2\pm3.18e-5$	1.79 ± 0.18	$5.85e-2\pm1.79e-3$	$7.31e-2\pm1.47e-4$	7.32 ± 12.49
FinGAT	$4.43e-3\pm3.62e-6$	$7.63e-2\pm3.73e-5$	2.44 ± 0.54	$2.11e-2\pm3.66e-4$	$7.18e-2\pm1.51e-5$	8.67 ± 0.32
STHAN	$4.18e-1\pm1.35e-2$	$3.53e-2\pm8.63e-5$	1.47 ± 0.02	$7.38e-2\pm3.07e-5$	$3.63e-2\pm1.47e-4$	7.91 ± 8.10
DR-GAT	$3.52e-3\pm1.27e-7$	$\mathbf{8.65e-2\pm2.96e-5}$	$\mathbf{4.04\pm0.09}$	$\mathbf{2.08e-3\pm2.83e-7}$	$\mathbf{7.61e-2\pm2.93e-5}$	$\mathbf{10.04\pm0.05}$

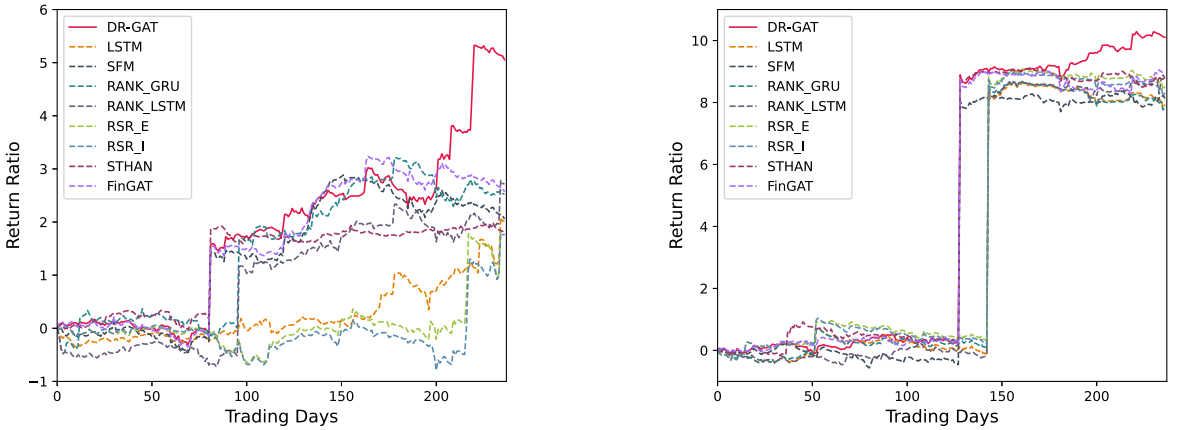


Fig. 4. Performance comparison of baseline and DR-GAT regarding IRR on the 2018-2023 dataset.

Table 4
The running time comparison of different models on two markets.

Market	*LSTM*	*SFM*	*RANK_LSTM*	*RANK_GRU*	*RSR_E*	*RSR_I*	*STHAN*	*FinGAT*	*DR-GAT*
NASDAQ	14.5003	344.7969	15.1279	15.3872	18.7032	18.7208	115.0083	1188.5228	32.4768
NYSE	14.5003	348.3784	15.2943	16.1703	21.0626	20.8346	152.5803	5023.0747	34.7095

efficiently handle large-scale graphs. To visually compare the computational efficiency of the models, we set the same batch size for all methods and ran them on the same GPU device. We recorded the time taken by each method for one epoch on the NASDAQ and NYSE datasets (in seconds). As shown in Table 4, our method incurred a slight sacrifice in terms of time compared to methods that do not involve graph learning or only utilize a single graph for relation learning (i.e., RSR_I, RSR_E). However, compared to other multi-graph learning methods, including FinGAT and STHAN, our model exhibited significantly improved computational efficiency. The additional time spent on multiple graphs and dynamic graphs is worthwhile considering the prominent returns achieved.

Research on investment strategies. DR-GAT combines the sequential features of stocks and the inter-company relational features to predict the stock returns, which is a direct indicator of the effectiveness of investment decisions. By backtesting the predicted IRR of the model, the profitability of trading according to the model's predictions can be simulated in real market conditions. Therefore, the results of backtesting the IRR can be used to measure the effectiveness of the model in guiding investments.

In the actual investment process, investors often select multiple stocks to minimize risk. We examine the performance of DR-GAT under three different investment strategies (Top1, Top5, Top10), which select stocks ranked in the top 1, 5, and 10 in terms of predicted returns, respectively. The IRR is calculated by adding up the average returns of the selected stocks for each strategy. The IRR of DR-GAT tested on the dataset from 2013 to 2017 under three different strategies is shown in Fig. 5. As can be seen from the figure, the Top1 strategy had the highest return rate, but its volatility was higher than those of the Top5 and Top10. Therefore, investing in the top-ranked stock is a high-risk strategy with high returns. This is also consistent with the portfolio theory [36] in real-market investments. According to portfolio theory, diversifying investments can reduce the risk of investment, but at the same time, it also diversifies profits. As shown in the Fig. 5, the IRR curves for the top 5 and top 10 strategies are relatively stable, which

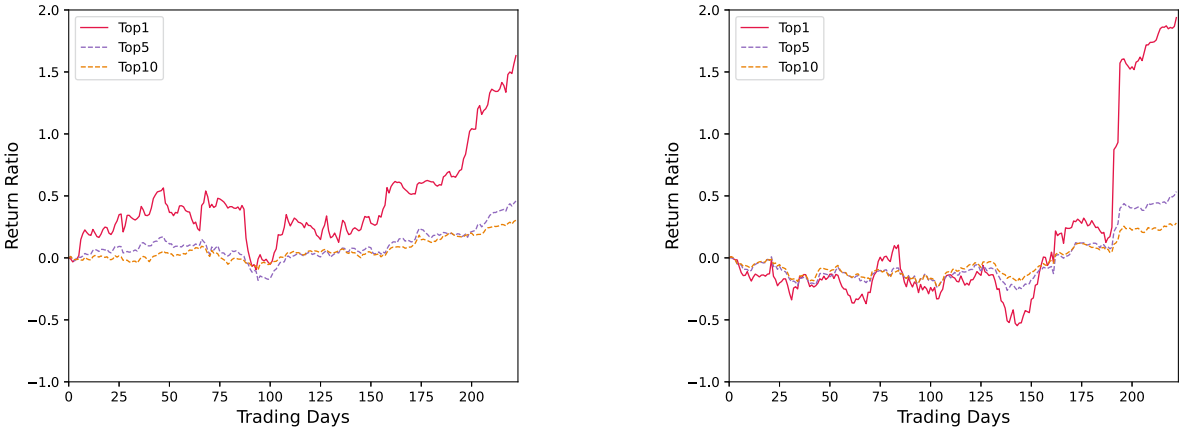


Fig. 5. Comparison of market simulation strategies (Top1, Top5, and Top10) based on DR-GAT's predicted IRR on the 2013-2017 dataset.

Table 5
Results on ablation study of the proposed DR-GAT.

Model	NASDAQ			NYSE		
	MSE	MRR	IRR	MSE	MRR	IRR
Full model	$\mathbf{1.21e-3 \pm 1.13e-8}$	$\mathbf{5.13e-2 \pm 9.29e-6}$	$\mathbf{1.52 \pm 0.01}$	$\mathbf{2.26e-4 \pm 7.04e-15}$	$\mathbf{6.32e-2 \pm 6.98e-5}$	$\mathbf{1.62 \pm 0.01}$
w/o long	$1.58e-3 \pm 1.12e-7$	$4.37e-2 \pm 9.50e-6$	0.72 ± 0.05	$2.74e-4 \pm 1.03e-8$	$4.76e-2 \pm 1.89e-5$	0.96 ± 0.12
w/o short	$1.30e-3 \pm 1.05e-7$	$4.14e-2 \pm 1.05e-5$	0.49 ± 0.06	$2.94e-4 \pm 8.51e-9$	$5.24e-2 \pm 7.53e-6$	0.43 ± 0.01
w/o router	$1.26e-3 \pm 1.53e-7$	$4.25e-2 \pm 8.69e-6$	0.36 ± 0.05	$4.00e-4 \pm 4.51e-8$	$4.92e-2 \pm 1.05e-4$	0.57 ± 0.10
w/o PE	$1.44e-3 \pm 6.33e-8$	$4.22e-2 \pm 1.29e-5$	0.42 ± 0.13	$2.61e-4 \pm 8.37e-9$	$4.49e-2 \pm 8.29e-5$	0.46 ± 0.26
w/o dynamic	$1.29e-3 \pm 1.90e-7$	$3.75e-2 \pm 8.95e-7$	0.43 ± 0.03	$4.17e-4 \pm 5.29e-8$	$5.26e-2 \pm 2.72e-5$	0.65 ± 0.05

demonstrates that investors bear less risk when allocating funds to stocks ranked in the top 5 or top 10 for investment. Meanwhile, although the IRR curve for the top 1 strategy exhibits high volatility, it yields higher returns, which indicates that the strategy of diversifying funds among the top 5 and top 10 stocks not only diversifies risk but also diversifies profits. Our backtesting results are in line with the investment principles observed in real markets, and therefore, they can effectively guide investment decisions. Furthermore, the performance of Top1, Top5, and Top10 on most testing days followed the order of Top1 > Top5 > Top10, further demonstrating that our model's predictions can rank stocks fairly accurately. With an equal budget, investing in stocks with high expected returns can result in higher cumulative returns.

Ablation Study. We evaluate the efficacy of each component of DR-GAT on the NASDAQ and NYSE datasets from 2013 to 2017. We compare the results of DR-GAT with the following five sub-models.

1) ***DR-GAT*** (***Full Model***):using all components of the proposed DR-GAT.
2) ***w/o long-term graph attention*** (***w/o long***):
   DR-GAT without using embeddings $\boldsymbol{g}_L^{s_p}(t)$ derived from long-term graph attention network.
3) ***w/o short-term graph attention*** (***w/o short***):
   DR-GAT without using embeddings $\boldsymbol{g}_S^{s_p}(t)$ derived from short-term graph attention network.
4) ***w/o relation mode router*** (***w/o router***):DR-GAT without using the Relation Graph Router. Instead, it performs relational feature learning for all stock series features simultaneously through a long-term graph attention network and a short-term graph attention network. Finally, it obtains the final relational features through fully connected layers.
5) ***w/o positional encoding*** (***w/o PE***):When
   DR-GAT performs series feature learning, it removes the position encoding from the multi-view stock price vectors.
6) ***w/o dynamic relation*** (***w/o dynamic***):When
   DR-GAT is performing relational feature learning, it removes all the relational embeddings obtained from the dynamic graph attention networks, including the dynamic long-term relational embedding $\boldsymbol{g}_l^{s_p}(t)$ and the dynamic short-term relational embedding $\boldsymbol{g}_s^{s_p}(t)$.

The results are shown in Table 5. We can observe that the complete DR-GAT achieves the best performance on all metrics, indicating the effectiveness of each component in our model. Removing the relation graph routing significantly reduces the model's performance, highlighting the necessity of dynamically selecting relation graphs based on the characteristics of the stock sequences. Ablation experiments demonstrate the remarkable effectiveness of improvements made in relation graphs construction and selection.

Table 6
The average IRR values of DR-GAT for the prediction of different learning rate in two markets.

Market	$lr = 0.0005$	$lr = 0.001$	$lr = 0.005$	dynamically adjusted learning rate
NASDAQ	1.07	1.06	0.71	1.52
NYSE	0.77	1.15	1.30	1.64

The combination of dynamic and static relation graph construction as well as relation graph routing based on oscillation detection significantly enhances the model's ability to aggregate stock relational features. In addition, positional encoding facilitates the capture of temporal dependencies and obtains more efficient series embeddings.

4.3. Hyperparameter analysis

We aim to investigate the effect of hyperparameters on the performance of DR-GAT, including the weight of pairwise ranking-aware loss α, the embedding dimension used in GRU and GAT, the length of the input series T, as well as the weight decay λ and learning rate. We fix other hyperparameters and modify any of these hyperparameters to conduct model training and prediction on the dataset from 2013 to 2017, comparing the changes in IRR values.

Balance Weight α. To examine the effects of different values of α on balancing the pointwise regression loss and pairwise ranking-aware loss, we conducted experiments on six α values (0, 0.1, 1, 10, 500, 800) separately on the NASDAQ and NYSE datasets. Fig. 6(a) and 6(b) show the IRR values obtained by the DR-GAT model with different α values on the NASDAQ and NYSE stock markets. By observation, the following results can be obtained. In the NASDAQ market, the IRR value for $\alpha = 0$ is 1.11, the IRR value for $\alpha = 0.1$ is 1.12, and as α increases to 1, IRR increases to 1.52. As α increases to 10, IRR decreases to 0.99. When α further increases to 500 and 800, IRR increases to 1.02 and 1.23, respectively. However, it can be observed that this increase is due to sacrificing regression loss for ranking loss, which leads to a decrease in performance on MSE while IRR increases. In the NYSE stock market, the IRR value increases from 0.71 to 0.77 and then to 1.23 as α increases from 0 to 0.1 and then to 1, respectively. Similarly to the NASDAQ market, as α increases from 1 to 10, the IRR value decreases to 1.06. When $\alpha = 500$, the peak IRR value of 1.63 is achieved in the tests, and as α increases again to 800, IRR falls to 1.35. Through multiple experiments, we found that the DR-GAT model with $\alpha = 1$ on the NASDAQ market and $\alpha = 500$ on the NYSE market can achieve higher IRR without sacrificing the accuracy of return rate prediction.

Embedding Dimension. The results shown in Fig. 6(c) and 6(d) indicate that the model performs best when the embedding size is 32. Embedding sizes that are too small (16) or too large (64) result in decreased model performance. Therefore, an embedding size of 32 was chosen for DR-GAT.

Sequential Length Input T. In order to investigate the impact of input series length T on model performance, we set T to 4, 8, 15, and 20. Fig. 6(e) and 6(f) illustrate the IRR under different input series lengths. It can be observed from the figures that the IRR performance of our model is affected by the input series length T. When T is less than or equal to 15, the IRR of DR-GAT increases with the increase of T in both markets. Specifically, when T is 15, our model achieves the best IRR results of 1.52 and 1.63 in the NASDAQ and NYSE markets, respectively. However, when T increases further, the performance of IRR deteriorates. When T increases to 20, the IRR values for the NASDAQ and NYSE markets decrease to 0.52 and 0.48, respectively. This is because sequences of different lengths contain different information. Firstly, when the series length is short, the information it contains is insufficient, and as the series length increases, the effective information it contains also increases. The series length, however, cannot grow indefinitely. When the series length reaches a large value, the proportion of redundant information increases, interfering with the model's prediction. Due to this, the IRR value increases first and then decreases as series length increases.

Weight Decay λ The parameter λ represents the coefficient of the regularization term in the loss function, which helps prevent overfitting of the model. The results in Figs. 6(g) and 6(h) indicate that the best performance is achieved when λ is set to 5e-4. If the weight decay is too small, it fails to effectively constrain the model parameters. On the other hand, if the weight decay is too large, it may interfere with the model's learning process.

Learning Rate lr As shown in Table 6, the choice of learning rate has a significant impact on the model's performance. Adaptive learning rates, compared to static learning rates, are more advantageous for model learning. Dynamic decay of the learning rate allows the model to converge as quickly as possible while also enabling learning in the vicinity of the optimal point.

4.4. Attention weight analysis

The DR-GAT model aims to dynamically adapt the relation graph and learn the relationship features. As discussed in Section 3.3, different companies require different relation graphs for more effective relationship learning. Based on the duration of relationship effectiveness, the relationship graph is divided into long-term and short-term relation graphs. Two types of relationship data are employed: static prior relationship data (i.e., industry relationship data and Wiki relationship data) and dynamic stock price trend similarity data. During a specific period, a company's status may be affected by market, policy, and other factors, and long-term relationships may remain effective or be broken. In such cases, the short-term relation graph established through stock price similarity will then take its place. By using graph attention networks, DR-GAT simulates how relevant companies interact with each other in the prior relation graph and the stock price trend similarity relation graph. The interaction between companies is represented as attention weights between adjacent nodes in the graph attention network, thus exploring long-term and short-term corporate

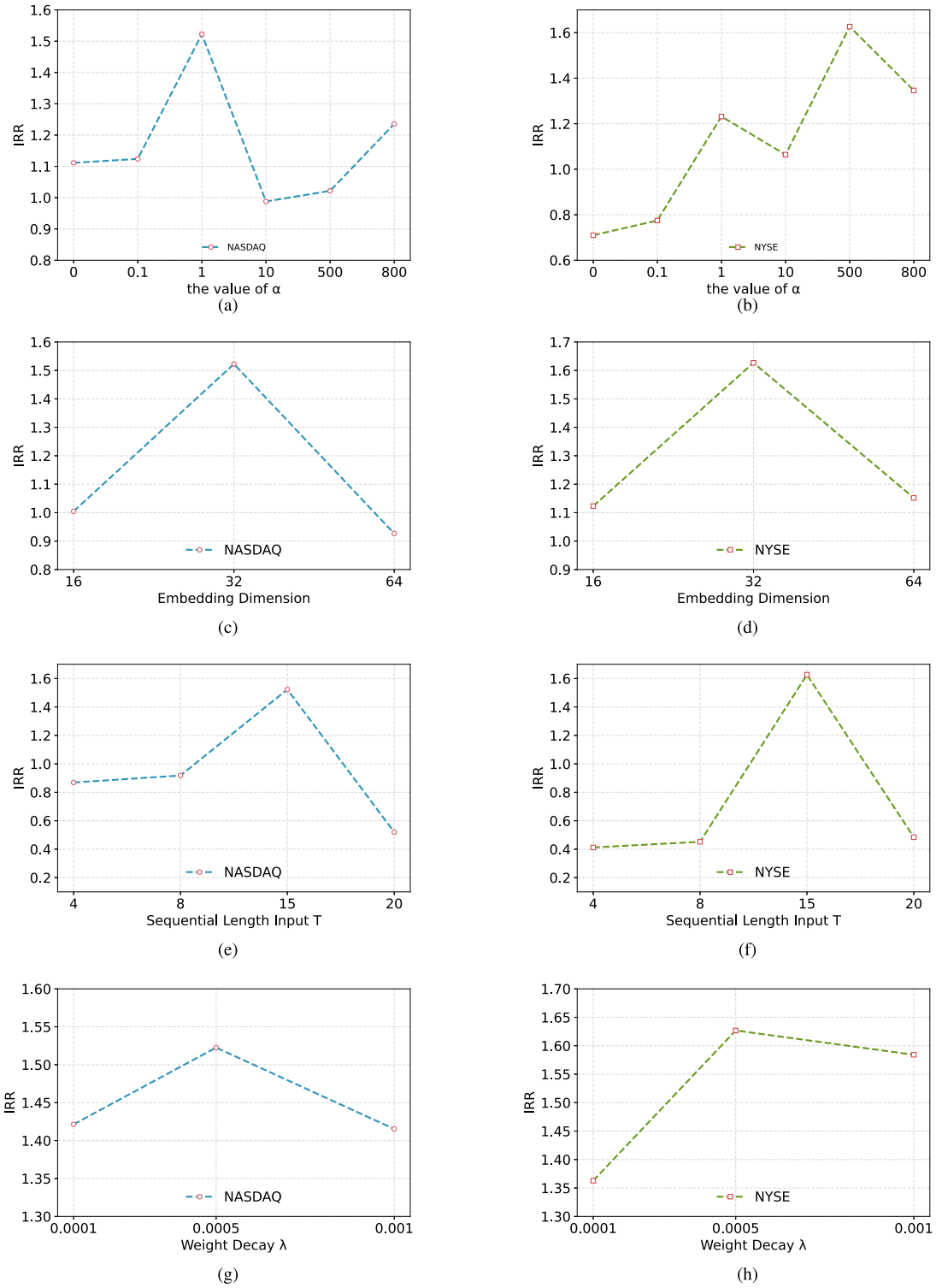
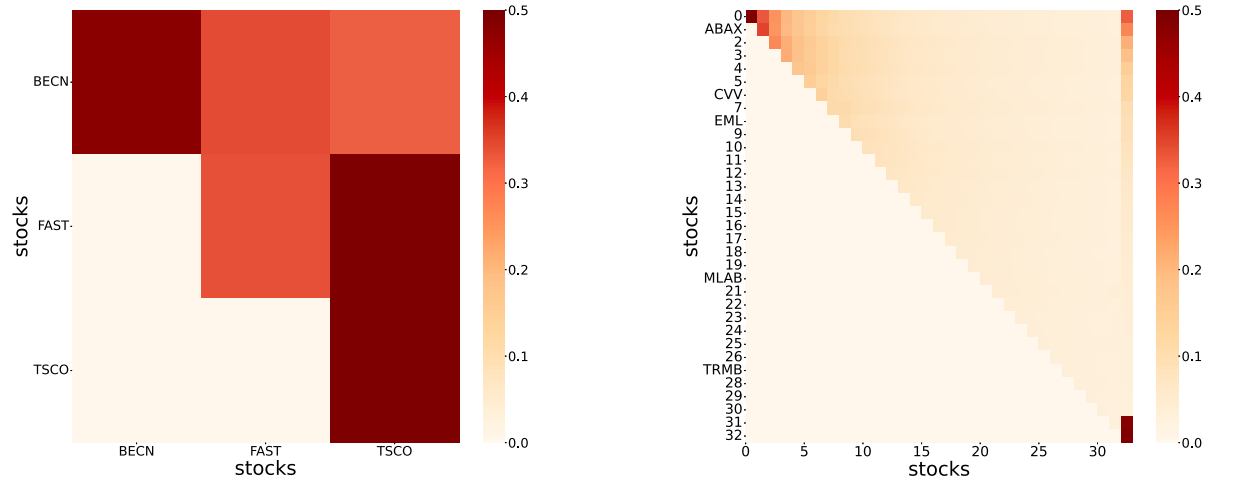


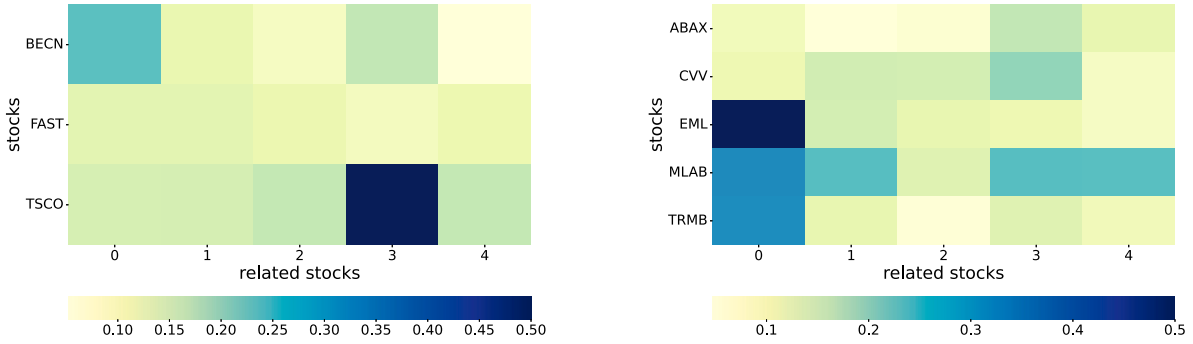
Fig. 6. Performance by varing (a)&(b): the balance weight, (c)&(d): the embedding size, (e)&(f): the input series length T in DR-GAT, and (g)&(h): the weight decay.

relationships through attention weights is meaningful. We randomly selected several test instances to visualize their graph attention weights. For long-term relation graphs, we studied the attention weights within industries using industry relationships as an example, and for some companies within an industry, we visualized the attention weights of their short-term relation graph attention network.



(a) "Building Materials" Industry (b) "Industrial Machinery/Components" Industry

Fig. 7. Visualizing attention weights within the industry.



(a) "Building Materials" Industry (b) some companies in the "Industrial Machinery/Components" Industry

Fig. 8. Visualizing short-term relationship attention weights.

We analyzed the latent relationships between companies captured by the model by comparing the attention weights in the long-term relation graph attention network and the short-term relation graph attention network.

Long-term Relation Attention Weight. The long-term relationship consists of three parts: industry prior relationship, Wiki prior relationship, and long-term dynamic stock price similarity relationship. As the long-term dynamic stock price similarity relationship evolves from short-term dynamic stock price similarity relationship, we choose a static prior graph attention network for analysis to better compare with the short-term relationship graph attention network. The visualization of attention weights for the "Building Materials" sector and the "Industrial Machinery/Components" sector in the NASDAQ market are shown in Fig. 7(a) and Fig. 7(b), respectively. It can be observed that there is a high correlation among stocks within the "Building Materials" industry, which is due to the fact that building materials are often used in combination and it is difficult for a single building material to enter the market. As for the "Industrial Machinery/Components" industry in Fig. 7(b), the attention weights among stocks within the industry are relatively low, which may be due to the fierce competition within the industry, leading to a tendency of uniform distribution of attention weights.

Short-term Relation Attention Weight. When long-standing relationships are at risk of failure due to market or policy shocks, it is necessary to establish new effective relationships to complete the learning of relational features. All impacts on stocks will be reflected in stock prices, and we establish short-term relationships through the similarity of stock price trends. Taking the "Building Materials" industry and the "Industrial Machinery/Components" industry used in the visualization of long-term relation weights as an example, we randomly select some stocks for visualization of short-term relation attention weights. Fig. 8(a) and 8(b) respectively visualize the short-term relation attention weights of some stocks in the "Building Materials" industry and the "Industrial Machinery/Components" industry. Each target stock establishes a new relation graph with other stocks through the medium of stock price trends, and the attention weights describe the strength of the correlation between the target stock and the related stock. As can be seen from the figures, the short-term relation attention weight of the FAST stock in the "Building Materials" industry is lower, while

that of TSCO is higher with some short-term related stocks. For some stocks in the “Industrial Machinery/Components” industry such as MLAB, their short-term relation attention weights are higher, while for ABAX, their short-term relation attention weights are lower.

Through the comparison of the weights of long-term and short-term relationships, we find that some companies have higher weights on long-term relation attention during the same period, while others have higher weights on short-term relation attention. Therefore, in order to capture more association information, it is necessary to adapt the optimal corporate relation graph for different companies. The dynamic relationship graph router proposed in this study selects the more effective relationships based on the company unit, which can achieve more precise capture of potential relationships between companies. In addition, stocks in the “Building Materials” industry have higher industry attention weights, which conforms to market rules and also proves the usefulness of learning relationships through graph attention networks.

5. Conclusion

Our work aims to model stock prediction tasks based on stock price information and prior relationship data. We propose a novel deep learning-based model, DR-GAT, for predicting stock returns and recommending profitable stocks based on those returns. Firstly, DR-GAT no longer solely relies on prior knowledge for learning the correlations between stocks, but rather uses a dynamic enterprise relationship graph that combines prior knowledge with the similarity of stock price trends. Secondly, DR-GAT is a dynamic model, which is reflected in two aspects: dynamic graph construction and relation graph routers. The dynamic graph based on the similarity of stock price trends compensates for the inadequacy of static prior relationship data in capturing potential relationships. Relation graph routing takes into account the impact of market uncertainty on stocks and selects the optimal relationship graph in both temporal and spatial dimensions. We conducted experiments using the NASDAQ and NYSE datasets. A comparison between DR-GAT and the current state-of-the-art competitive methods reveals that the former performs significantly better. Additionally, we discuss how DR-GAT can guide actual trading operations, and validate its effectiveness through ablation experiments.

Three directions are possible for the future extension of DR-GAT. Firstly, while graph attention networks have been successful in modeling the relationship between stocks, we believe that there are more powerful graph structures that can capture the relation information more accurately. Therefore, we will try to improve the graph structure. As a second point, although the proposed RGR, which adapts the relationship graph through oscillation detection, has achieved good results in stock price prediction, it is still worth exploring more detailed routing criteria. Although the proposed RGR, which adapts the relationship graph through oscillation detection, has achieved good results in stock price prediction, it is still worth exploring more granular routing evaluation criteria. As a third point, news data has guiding significance for predicting stock behavior, and it is a valuable data source. Our goal is to combine news data to further improve the effectiveness of DR-GAT.

CRediT authorship contribution statement

Zengyu Lei: Investigation, Methodology, Software, Writing – original draft. **Caiming Zhang:** Conceptualization, Methodology, Writing – review & editing. **Yunyang Xu:** Software, Validation, Visualization. **Xuemei Li:** Conceptualization, Writing – review & 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

The data that has been used is confidential.

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. U22A2033, No. 62072281).

References

[1] O.B. Sezer, M.U. Gudelek, A.M. Ozbayoglu, Financial time series forecasting with deep learning: a systematic literature review: 2005–2019, Appl. Soft Comput. 90 (2020) 106181.
[2] R.F. Engle, Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation, Econometrica (1982) 987–1007.
[3] A.A. Ariyo, A.O. Adewumi, C.K. Ayo, Stock price prediction using the arima model, in: 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 2014, pp. 106–112.
[4] A.A. Adebiyi, A.O. Adewumi, C.K. Ayo, Comparison of arima and artificial neural networks models for stock price prediction, J. Appl. Math. 2014 (2014).
[5] C. Xiao, W. Xia, J. Jiang, Stock price forecast based on combined model of ari-ma-ls-svm, Neural Comput. Appl. 32 (2020) 5379–5388.
[6] 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.

[7] K. Chen, Y. Zhou, F. Dai, A lstm-based method for stock returns prediction: a case study of China stock market, in: 2015 IEEE International Conference on Big Data (Big Data), IEEE, 2015, pp. 2823–2824.
[8] 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.
[9] J.B. Pollack, Recursive distributed representations, Artif. Intell. 46 (1990) 77–105.
[10] R. Socher, C.C. Lin, C. Manning, A.Y. Ng, Parsing natural scenes and natural language with recursive neural networks, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 129–136.
[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.
[12] X. Ying, C. Xu, J. Gao, J. Wang, Z. Li, Time-aware graph relational attention network for stock recommendation, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 2281–2284.
[13] 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.
[14] 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.
[15] Y.-L. Hsu, Y.-C. Tsai, C.-T. Li, Fingat: financial graph attention networks for recommending top-k profitable stocks, arXiv preprint arXiv:2106.10159, 2021.
[16] W. Xu, W. Liu, L. Wang, Y. Xia, J. Bian, J. Yin, T.-Y. Liu, Hist: a graph-based framework for stock trend forecasting via mining concept-oriented shared information, arXiv preprint arXiv:2110.13716, 2021.
[17] C. Xu, H. Huang, X. Ying, J. Gao, Z. Li, P. Zhang, J. Xiao, J. Zhang, J. Luo Hgnn, Hierarchical graph neural network for predicting the classification of price-limit-hitting stocks, Inf. Sci. 607 (2022) 783–798.
[18] J. Wu, K. Xu, X. Chen, S. Li, J. Zhao, Price graphs: utilizing the structural information of financial time series for stock prediction, Inf. Sci. 588 (2022) 405–424.
[19] R. Kim, C.H. So, M. Jeong, S. Lee, J. Kim, J. Kang, Hats: a hierarchical graph attention network for stock movement prediction, arXiv preprint arXiv:1908.07999, 2019.
[20] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint arXiv:1710.10903, 2017.
[21] T. Fischer, C. Krauss, Deep learning with long short-term memory networks for financial market predictions, Eur. J. Oper. Res. 270 (2018) 654–669.
[22] J. Wang, Y. Hu, T.-X. Jiang, J. Tan, Q. Li, Essential tensor learning for multimodal information-driven stock movement prediction, Knowl.-Based Syst. (2023) 110262.
[23] Q. Song, A. Liu, S.Y. Yang, Stock portfolio selection using learning-to-rank algorithms with news sentiment, Neurocomputing 264 (2017) 20–28.
[24] P. Chaovalit, A. Gangopadhyay, G. Karabatis, Z. Chen, Discrete wavelet transform-based time series analysis and mining, ACM Comput. Surv. 43 (2011) 1–37.
[25] S. Lahmiri, Wavelet low- and high-frequency components as features for predicting stock prices with backpropagation neural networks, J. King Saud Univ, Comput. Inf. Sci. 26 (2014) 218–227.
[26] D. Jothimani, A. Başar, Stock index forecasting using time series decomposition-based and machine learning models, in: International Conference on Innovative Techniques and Applications of Artificial Intelligence, Springer, 2019, pp. 283–292.
[27] G. Liu, Y. Mao, Q. Sun, H. Huang, W. Gao, X. Li, J. Shen, R. Li, X. Wang, Multi-scale two-way deep neural network for stock trend prediction, in: IJCAI, 2020, pp. 4555–4561.
[28] L. Ortega, K. Khashanah, A neuro-wavelet model for the short-term forecasting of high-frequency time series of stock returns, J. Forecast. 33 (2014) 134–146.
[29] 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.
[30] E.F. Fama, The behavior of stock-market prices, J. Bus. 38 (1965) 34–105.
[31] Y. Wang, H. Li, J. Guan, N. Liu, Similarities between stock price correlation networks and co-main product networks: threshold scenarios, Phys. A, Stat. Mech. Appl. 516 (2018).
[32] 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.
[33] R. Dolphin, B. Smyth, Y. Xu, R. Dong, Measuring financial time series similarity with a view to identifying profitable stock market opportunities, in: Case-Based Reasoning Research and Development: 29th International Conference, ICCBR 2021, Salamanca, Spain, September 13–16, 2021, Proceedings 29, Springer, 2021, pp. 64–78.
[34] L. Zhang, C. Aggarwal, G.-J. Qi, Stock price prediction via discovering multi-frequency trading patterns, 2017, pp. 2141–2149, https://doi.org/10.1145/3097983.3098117.
[35] 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.
[36] H. Markowitz, Portfolio selection, J. Finance 7 (1952) 77–91, https://doi.org/10.1111/j.1540-6261.1952.tb01525.x, https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1952.tb01525.x.