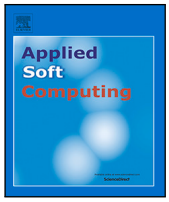
Applied Soft Computing Journal 154 (2024) 111329

Contents lists available at ScienceDirect

Applied Soft Computing

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



Stock trend prediction based on dynamic hypergraph spatio-temporal network

Sihao Liao [a], Liang Xie [a,*], Yuanchuang Du [a], Shengshuang Chen [a], Hongyang Wan [a], Haijiao Xu [b]

[a] School of Science, Department of Mathematics, Wuhan University of Technology, Wuhan 430070, China
[b] School of Computer Science, Guangdong University of Education, Guangzhou 510303, China

ARTICLE INFO

Keywords:
Stock trend prediction
Hypergraph neural network
Spatio-temporal learning
Deep neural network
Time series analysis

ABSTRACT

Predicting stock trends is conducive to optimize returns from stock investments, which gains great interest from investors and researchers. Relations between stocks can provide important information for stock trend prediction. However existing stock prediction approaches only consider pairwise linkages, and ignore complex higher-order relations among stocks. To address these limitations, this paper proposes a dynamic hypergraph spatio-temporal network (DHSTN). DHSTN utilizes GRU to learn the sequential embedding of stocks, and a dynamic hypergraph network is proposed to learn the spatio-temporal relations among stocks. In the dynamic hypergraph network, firstly, a novel dynamic hypergraph construction module based on graph attention network is designed to capture stock higher-order spatial relations which are dynamically changing over time. Secondly, an industry relations aggregator based on hypergraph is considered in hypergraph convolution. Finally, a multi-relation fusion module is designed to integrate static and dynamic stock relations. Experiments on CSI300 and NASDAQ100 datasets show that DHSTN outperforms representative stock prediction methods by at least 4.99% in terms of F1-score and at least 47.9% in terms of sharpe ratio.

1. Introduction

As a high-risk and high-reward way of investing in the financial markets, stock trading has attracted many investors. In recent years, stock trend prediction, which can maximize returns in stock trading, has become an important research focus. Stock trend prediction technology aims to predict the future trends of stock price, it has gained increasing attention in providing investors reliable information about the possible direction of stock price movement and helping investors make profitable transaction decisions in the stock market [1,2].

Most stock trend prediction methods treat predicting stock movements as a time series modeling problem and solve it by using statistical models. Examples of such models include Kalman filter [3] and autoregressive integrated moving average (ARIMA) [4]. Compared to statistical methods, machine learning can better capture non-linear relations in time series data. In recent years, machine learning methods such as support vector machine (SVM) [5], support vector regression (SVR) [6,7], extreme learning machine (ELM) [8], relevance vector machine (RVM) [9], random vector functional link (RVFL) [10], and adaptive neuro-Fuzzy inference system (ANFIS) [11] are widely used to deal with time series prediction problems, including stock trend prediction [12]. However, because of the inherent non-stationarity and complexity of stock price movements, traditional machine learning methods cannot accurately model the dynamic trend of chaotic stock data. With the development of deep learning [13,14], more and more researchers have started to use deep neural networks to analyze stock data. Indeed, these models have shown superior performance compared to traditional methods. Recurrent neural network (RNN) [15] and Transformer network [16] have shown promising results in stock trend prediction due to their strong ability to capture the underlying dynamic characteristics in chaotic time series. However, they have certain limitations in fully exploiting the temporal and relational features of the stock market. Deep neural network models typically treat each stock as independent. They overlook the rich relations among stocks, and thus neglect many valuable factors.

In financial markets, there are extensive links among firms, which means that the stock price movements of a target firm can be influenced by other related firms. On the one hand, stocks in the same industry tend to have similar long-term trend. On the other hand, when a new event impacts the market, it not only causes stock price fluctuation for the related firms but also affects their upstream and downstream firms. The rich relations among stocks can provide valuable information for predicting stock trends [17,18]. To incorporate stock relations into

* Corresponding author.
E-mail addresses: 319535@whut.edu.cn (S. Liao), whutxl@hotmail.com (L. Xie), 319527@whut.edu.cn (Y. Du), chenshsh@whut.edu.cn (S. Chen), 285676@whut.edu.cn (H. Wan), guesskkk@alumni.hust.edu.cn (H. Xu).

https://doi.org/10.1016/j.asoc.2024.111329
Received 26 August 2023; Received in revised form 18 January 2024; Accepted 26 January 2024
Available online 1 February 2024

stock trend prediction, an intuitive solution is to represent the stock relations as predefined graphs and then use graph-based techniques [19–21] to predict stock price movements. Traditional graph-based models cannot accurately model attribute-sensitive momentum spillovers [22] among relevant firms. To overcome this problem, Cheng and Li [23] proposed an attribute-sensitive approach, but they only considered momentum spillovers between pairwise stocks.

Existing research on using stock relations to predict stock movements now faces three major challenges:

(1) Traditional studies simplify the relations between stocks. They often assume pairwise associations between two stocks and fail to consider higher-order and multivariate relations among multiple stocks, such as multiple stocks belonging to the same industry.

(2) Predefined static relations of listed firms cannot capture dynamic changes. The relations among listed firms may dynamically change over time. For instance, two firms that were originally in the same industry may change over time to be in different industries. Additionally, the strength of relations between listed firms, and the strength of relations between industries and listed firms, may also vary over time.

(3) Focusing on the momentum spillovers between pairwise stocks has limitation. The momentum spillovers among stocks in the same industry are also significant. To capture the attribute-sensitive momentum spillovers among relevant stocks more accurately, it is necessary to consider the impact of stock industry attributes on stock trends.

To address these challenges, in this paper, we propose a novel framework called dynamic hypergraph spatio-temporal network (DHSTN). For the first challenge, we represent the higher-order relations between stocks by using hypergraphs, which extend the traditional graph theory by allowing a hyperedge to correlate multiple stocks simultaneously, which captures their higher-order relations. For the second challenge, we develop a dynamic hypergraph construction module based on graph attention network (GAT) to extract the dynamic hypergraph structures from stock time series data. For the third challenge, we design an attribute aggregator based on hypergraph structure, which considers the impact of stock industry attributes on stock trends.

The primary contributions of our study can be summarized as follows:

- A dynamic hypergraph network for learning spatio-temporal relations of stocks is designed in this paper. In hypergraph construction, dynamic hypergraphs are adaptively obtained by GAT to reflect the higher-order spatial relations among stocks which are dynamically changing over time.
- An industry relations aggregator in hypergraph convolution is designed in this paper, which can capture the impact of the industry on stock movements. And a multi-relation fusion module is proposed to integrate static stock relations and dynamic stock relations to obtain the final relational embedding of stocks.
- The temporal and spatial features of stocks are modeled in a unified way to improve the results of stock trend prediction. Experiments on CSI300 and NASDAQ100 datasets have demonstrated the effectiveness of DHSTN.

The remainder of this paper is organized as follows: Section 2 introduces the work related to our method. Section 3 introduces the preliminary knowledge about stock trend prediction and construction of stock relation hypergraphs. Section 4 presents our proposed DHSTN. Sections 5 and 6 describe the datasets and show the experimental results, respectively. Finally, the conclusions drawn from this study are presented in Section 7.

2. Related work

2.1. Deep learning-based time series modeling

With the rapid development of deep learning, many researchers have started to use deep neural networks to analyze time series data, including stock data. RNNs have shown satisfactory performance in predicting stock trends due to their ability to capture the underlying dynamics of chaotic time series. For instance, Akita et al. [24] utilized long short-term memory (LSTM) network combined with historical stock data and textual information for stock price forecasting. Rahman et al. [25] addressed the stock price prediction problem by using gated recurrent units (GRU). Ding et al. [26] proposed a Transformer model which is enhanced with a multiscale Gaussian prior for stock trend prediction tasks. Teng et al. [27] proposed a multi-scale local cues and hierarchical attention-based LSTM model to predict stock price movements. Park et al. [28] integrated LSTM and random forest to predict stock market returns. Zhang et al. [29] utilized the Transformer model and multiple attention mechanisms to achieve accurate stock movement prediction. Li et al. [30] proposed a clustering-enhanced deep learning framework to predict stock prices, they used RNN, LSTM and GRU to optimize the accuracy of stock price prediction. Zhao et al. [31] combined the emotion enhanced convolutional neural network (ECNN), the denoising autoencoder (DAE) models, and LSTM to predict stock market.

However, the above models treat each stock as independent, and they ignore the rich relations among stocks and fail to utilize these relations to improve stock prediction.

2.2. Graph-based market relations modeling

Graph convolutional network (GCN) [32] takes the graph structure and node features as inputs and aggregates neighbor information of nodes, and then performs nonlinear transformations across all feature dimensions to generate new features. GCN can capture the complex relations between nodes in the graph. In recent years, many researchers have utilized GCN to incorporate the rich relations among stocks into stock trend prediction. For instance, Chen et al. [33] modeled the relations between stocks based on the ownership relation graph and utilized GCN to predict stock movements. However, since traditional graph learning techniques use static graphs for convolution, they are unable to capture the temporal evolution characteristics of the stock market.

To address the limitations of traditional graph learning techniques mentioned above, Cheng and Li [23] combined RNN and graph networks to propose an attribute-driven graph attention network (ADGAT) that adjusts the momentum spillover effect of stock prices based on other attributes of two related stocks. Cheng et al. [34] proposed a multi-modality graph neural network, which learns multi-modal inputs including stock prices and stock relations for stock movement prediction. Shi et al. [35] utilized GCN to extract stock embeddings and applied LSTM to model temporal sequence data of stocks.

Although graph-based neural network models have made progress in stock prediction, they typically only consider pairwise connections between stocks, and ignore the higher-order complex relations among stocks. In real financial markets, stock entities exhibit intricate multivariate relations. For instance, industry sectors comprise multiple stocks. These relations extend beyond pairwise connections and require a more comprehensive modeling approach.

2.3. Hypergraph representation and learning

Traditional graphs have limitations in modeling higher-order multivariate relations because they can only express relations between two nodes. To address this problem, the hypergraph neural network (HGNN) [36] has been proposed. Hypergraphs extend the concept of simple graphs in graph theory, where hyperedges can connect two or more vertices, forming relations among multiple entities [37].

Hypergraphs have gained increasing attention and applications in various fields. For instance, in computer vision, hypergraphs can describe relations between visual features, enabling tasks such as visual

Table 1
Mathematical notations.

Notation	Definition
$\boldsymbol{X}^t$	The historical feature sequence of N stock on trading day t.
$\boldsymbol{P}$	The aggregated hyperedge feature matrix.
$\boldsymbol{H}$	The predefined stock industry relations hypergraph.
$\boldsymbol{H}_D^t$	The dynamic hypergraph.
$\boldsymbol{Z}^t$	The stock temporal embedding.
$\boldsymbol{S}^t$	The stock relational embedding.
$\mathrm{HConv}(\cdot)$	The hypergraph convolutional layer.
$\mathrm{c}(\cdot)$	The industry relations aggregator.

classification [38] and image retrieval [39]. Each hyperedge in a hypergraph represents a set of vertices, making it suitable for modeling non-pairwise relations among stocks [40,41]. Sawhney et al. [42] applied hypergraph neural network to construct a relational graph for stocks and proposed a spatio-temporal hypergraph attention network model for stock selection. Li et al. [43] proposed a hypergraph-based reinforcement learning method for stock portfolio selection.

However, existing hypergraph neural network models often suffer from the limitation of using only the initial hypergraph structure and they ignore dynamic variations during the training process. Jiang et al. [44] introduced a dynamic hypergraph construction method that uses the k-nearest neighbors (k-NN) to dynamically generate hyperedges, but it ignores the temporal information of nodes at different time steps.

3. Preliminary

To more clearly describe the related definitions and formulas, bold capital letters (e.g., $\boldsymbol{X}$), bold lowercase letters (e.g., $\boldsymbol{x}$) and normal lowercase letters (e.g., x) are used to denote matrices, vectors and scalars respectively. If not otherwise specified, all vectors are in column form, and X_{ij} denotes the element at row i and column j of $\boldsymbol{X}$. The definitions of main mathematical notations in this paper are shown in Table 1.

3.1. Problem formulation

- **Stock Sequence**: We let $S = \{s_1, \ldots, s_N\}$ denote a set of N stocks, where each stock $s_i \in S$ has multiple technical indicator attributes on trading day t [45,46], such as the opening price, closing price, highest price, lowest price, and trading volume. Each stock s_i is associated with historical sequence data $\boldsymbol{x}_i^t \in R^{1\times T\times F}$ on trading day t, where T is the length of the sequence, F is the feature dimension of the original input data. $\boldsymbol{X}^t = [\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_N^t] \in R^{N\times T\times F}$ is the historical feature sequence of N stock on trading day t.
- **Hypergraph** : We denote a hypergraph as $G = (V, E, \boldsymbol{W})$ that represent the higher-order relations of stocks. V is the set of vertices and E is the set of hyperedges. Each hyperedge $\varepsilon \in E$ is assigned a positive weight, and all the weights are stored in the diagonal matrix $\boldsymbol{W} \in R^{|E|\times|E|}$. In this study, we set $\boldsymbol{W} = \boldsymbol{I}$, which means that all hyperedges have equal weights.

3.2. Stock trend prediction

Stock trend prediction is usually considered as a binary classification task [47,48]. If the closing price of stock s_i is higher than the opening price on day $t+1$, the stock s_i is labeled with "up" ($y_i^{t+1} = 1$), otherwise labeled with "down" ($y_i^{t+1} = 0$). The goal of stock trend prediction is to learn a function, which is defined as follows:

$$[\boldsymbol{X}^t] \xrightarrow{f} \hat{\boldsymbol{Y}}^{t+1}, \tag{1}$$

where $\boldsymbol{X}^t = [\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_N^t]$ is the input stock data and $\hat{\boldsymbol{Y}}^{t+1}$ is the predicted future trend of all stocks on day $t+1$:

$$\hat{\boldsymbol{Y}}^{t+1} = [\hat{\boldsymbol{y}}_1^{t+1}, \ldots, \hat{\boldsymbol{y}}_N^{t+1}], \tag{2}$$

where $\hat{\boldsymbol{y}}_i^{t+1}$ represents the predicted future trend of stock s_i on day $t+1$.

4. Method

The overall architecture of our DHSTN framework is shown in Fig. 1. Specifically, we extract the features of stocks in the temporal and spatial domains and combine them to achieve end-to-end stock trend prediction. In extracting the temporal features of stocks, GRU is used to learn stock temporal embedding. In extracting the spatial features of stocks, a dynamic hypergraph network is proposed to learn stock spatial embedding. In the dynamic hypergraph network, the dynamic hypergraph construction module based on GAT enables the hypergraph to change dynamically with time, reflecting the spatial relations of stocks at different times. The hypergraph convolutional layer is used to extract the relational embedding of stocks. The industry relations aggregator based on hypergraph structure is used to learn the impact of stock industry attributes on stock trends. And the multi-relation fusion module is used to combine predefined static stock relations and dynamic stock relations.

4.1. Temporal embedding

In this study, GRU [49] is selected to extract stock temporal embedding. GRU solves the problem faced by traditional RNNs in capturing long-distance information by introducing a gating structure with a relatively simple design, fewer parameters, and faster training capability. The GRU update process is as follows:

$$\begin{aligned} r^t &= \sigma(\boldsymbol{W}_r \cdot [\boldsymbol{z}^{t-1}, \boldsymbol{x}^t]), \\ u^t &= \sigma(\boldsymbol{W}_u \cdot [\boldsymbol{z}^{t-1}, \boldsymbol{x}^t]), \\ \widetilde{\boldsymbol{z}}^t &= \tanh(\boldsymbol{W}_{\widetilde{z}^t} \cdot [r^t \cdot \boldsymbol{z}^{t-1}, \boldsymbol{x}^t]), \\ \boldsymbol{z}^t &= (1-u^t)\cdot \boldsymbol{z}^{t-1} + u^t \cdot \widetilde{\boldsymbol{z}}^t, \end{aligned} \tag{3}$$

where r^t is reset gate, u^t is update gate, $\boldsymbol{W}_r, \boldsymbol{W}_u, \boldsymbol{W}_{\widetilde{z}^t} \in R^{U\times F}$ are the weight matrices of reset gate, update gate and candidate state respectively. U is the number of hidden units and F is the input dimension. $\boldsymbol{x}^t$ is the input at the current moment, $\boldsymbol{z}^{t-1}$ and $\boldsymbol{z}^t$ are the previous moment state and the final output state respectively.

Stock data contain rich time information, and the historical stock data is an important reference for predicting future trend. We use the historical stock data as the input data of GRU to extract the temporal embedding of stocks.

Specifically, we let $S = \{s_1, s_2, \ldots, s_N\}$ denote the set of N stocks, feed the historical feature sequence $\boldsymbol{x}_i^t$ of stock $s_i \in S$ with lookback time window length T into GRU, and extract the state $\boldsymbol{z}_i^t$ of the last hidden layer as the temporal embedding of stock:

$$\boldsymbol{Z}^t = GRU(\boldsymbol{X}^t), \tag{4}$$

where $\boldsymbol{X}^t = [\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_N^t] \in R^{N\times T\times F}$ is the historical feature sequence of N stocks on trading day t, T is the lookback time window length, F is the feature dimension of the original input data, $\boldsymbol{Z}^t = [\boldsymbol{z}_1^t, \ldots, \boldsymbol{z}_N^t] \in R^{N\times U}$ is the temporal embedding of N stocks, and U denotes the dimensions of the output embedding which is also the number of hidden units in GRU.

4.2. Dynamic hypergraph network for spatio-temporal learning

Traditional graph-based models rely on predefined relational graph structures, and the hypergraph structures in traditional hypergraph-based models are static. However, when dealing with time series data such as stocks or traffic flow, the spatial relations among nodes change dynamically over time. This paper employs hypergraphs to depict

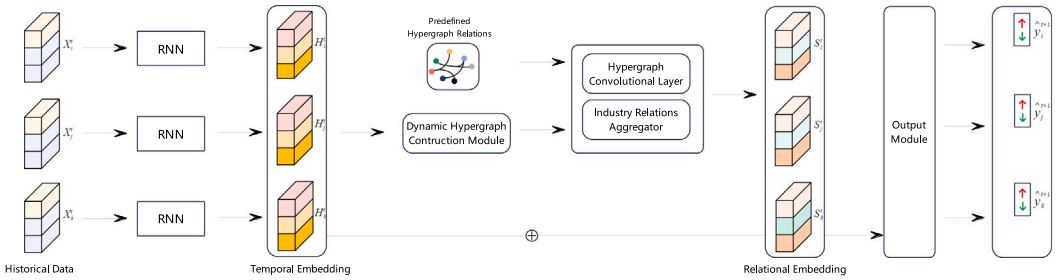


Fig. 1. The overall architecture of DHSTN framework.

the stock spatial relations which means the interconnections among stock nodes, and proposes a dynamic hypergraph network to learn the time-varying spatial relations among nodes. The dynamic hypergraph network consists of four modules, including dynamic hypergraph construction, hypergraph convolution, industry relations aggregator and multi-relation fusion. The details of each module are described in following subsections.

4.2.1. Dynamic hypergraph construction module

Time series data contain dynamic relational graph structures. In this study, we introduce a dynamic hypergraph construction module with the aim of learning the dynamic hypergraph structures in time series data to better capture the time-varying relations among nodes.

In hypergraph, the incidence matrix represents the first-order relation between stocks (nodes) and industries (hyperedges), and depicts the higher-order relation among stocks by hyperedges. GAT [50] has the capacity to learn the first-order relation dynamically, so we use it to generate hyperedges and construct the incidence matrix from stock time series data in this study. The drawback of existing dynamic hypergraph construction methods [44,51] is that they are detached from the overall network design and cannot be optimized in an unified process with the entire network. Compared with existing methods, GAT can be used as a part of the network and trained uniformly to better optimize the dynamic hypergraph relations.

Specifically, we first feed the temporal embedding $\boldsymbol{Z}^t$ into GAT to extract the embedding $\boldsymbol{M}^t = [\boldsymbol{m}_1^t, \dots, \boldsymbol{m}_N^t] \in R^{N\times U}$ as follows:

$$\boldsymbol{m}_i^t = \mathrm{GAT}(\boldsymbol{z}_i^t) = \sigma\left(\sum_{j\in N(i)} \alpha_{ij} \boldsymbol{W} \boldsymbol{z}_i^t\right), \tag{5}$$

where σ is a sigmoid function, $N(i)$ is the set of neighbors of the node i, and α_{ij} represents the importance weight of the node i to the node j. The step for calculating α_{ij} is as follows:

$$\alpha_{ij} = \frac{\exp(\mathrm{LeakyReLU}(\boldsymbol{a}_{ij}{}^T[\boldsymbol{W}_i \boldsymbol{z}_i^t \oplus \boldsymbol{W}_j \boldsymbol{z}_j^t]))}{\sum_{k\in N(i)} \exp(\mathrm{LeakyReLU}(\boldsymbol{a}_{ik}^T[\boldsymbol{W}_i \boldsymbol{z}_i^t \oplus \boldsymbol{W}_k \boldsymbol{z}_k^t]))}, \tag{6}$$

where $\boldsymbol{W}_i, \boldsymbol{W}_j, \boldsymbol{W}_k \in R^{U\times U}$ and $\boldsymbol{a}_{ij}, \boldsymbol{a}_{ik} \in R^{2U}$ are learnable parameters.

Subsequently, we feed the embedding $\boldsymbol{M}^t$ into a fully connected layer (FC) with softmax function and get the incidence matrix $\boldsymbol{H}_D^t$ which denotes the relations between stock nodes and hyperedges from the stock time series data:

$$\boldsymbol{H}_D^t = \mathrm{FC}(\boldsymbol{M}^t) = \mathrm{softmax}(\tanh(\boldsymbol{W}_f \boldsymbol{M}^t + \boldsymbol{b}_f)), \tag{7}$$

where $\boldsymbol{H}_D^t \in R^{N\times\lambda}$ also represents the dynamic hypergraph embedded in stock time series data, and it can depict the higher-order relations among stocks. $\boldsymbol{W}_f \in R^{U\times\lambda}$ is the weight matrix, $\boldsymbol{b}_f \in R^{\lambda}$ is the bias vector, and λ is a hyperparameter.

4.2.2. Hypergraph convolutional layer

In this study, we use hypergraph convolution [36] to extract the relational embedding of stocks. The hypergraph is defined as $G = (V, E, \boldsymbol{W})$, and can also be represented by the incidence matrix $\boldsymbol{H} \in R^{|V|\times|E|}$:

$$\boldsymbol{H}(v,\varepsilon) = \begin{cases} 1, v \in \varepsilon \\ 0, v \notin \varepsilon \end{cases}, \tag{8}$$

where $|V|$ is the number of vertices and $|E|$ is the number of hyperedges.

Specifically, we define a hypergraph convolutional layer $\mathrm{HConv}(\cdot)$. The input to the hypergraph convolutional layer is a feature matrix $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_{|V|}] \in R^{|V|\times U}$ and an incidence matrix $\boldsymbol{H} \in R^{|V|\times|E|}$. U denotes the dimensions of the input features. The node update process of the hypergraph convolutional layer is as follows:

$$\begin{aligned} \boldsymbol{x}_i{}' &= \mathrm{HConv}\left(\boldsymbol{x}_i, \Theta, \boldsymbol{H}\right) \\ &= \sigma\left(\sum_{j=1}^{|V|}\sum_{m=1}^{|E|} \boldsymbol{H}(x_i, \varepsilon_m)\boldsymbol{H}(x_j, \varepsilon_m)\boldsymbol{x}_i\Theta\right), \end{aligned} \tag{9}$$

where $\boldsymbol{x}_i{}' \in R^{F'}$ is the output, σ is a sigmoid function, $\Theta \in R^{U\times F'}$ is the parameter to be learned during the training process and F' is the hidden size of the hypergraph convolutional layer.

The hypergraph convolutional layer is node-edge-node transformed to efficiently extract the relational embedding of stocks. The process is as follows:

$$\widetilde{\boldsymbol{X}} = \mathrm{ReLU}(\boldsymbol{X} \cdot \Theta), \tag{10}$$

$$\boldsymbol{P} = (\boldsymbol{H}^T \cdot \widetilde{\boldsymbol{X}}), \tag{11}$$

$$\boldsymbol{X}' = \sigma(\boldsymbol{H} \cdot \boldsymbol{P}), \tag{12}$$

where $\boldsymbol{X} \in R^{|V|\times U}$ is the input node feature matrix. $\widetilde{\boldsymbol{X}} \in R^{|V|\times F'}$ is the transformed node feature matrix. $\boldsymbol{P} \in R^{|E|\times F'}$ is the aggregated edge feature matrix. $\boldsymbol{X}' \in R^{|V|\times F'}$ is the output node feature matrix. ReLU is the activation function. Eq. (10), Eq. (11) and (12) represent node feature transformation, edge feature gathering and node feature aggregating respectively.

4.2.3. Industry relations aggregator

The existence of momentum spillover effects between interrelated stocks has not been precisely modeled in traditional graph convolutional models. For instance, in traditional GCN models [52–54], when the price of a stock fluctuates, the associated stocks are also affected correspondingly. However, in the real stock market, if the price movement of a stock is accompanied by a small volume of trades, its impact on associated stocks is not significant. To address this issue, Cheng and Li [23] proposed an attribute-driven graph network that adjusts the momentum spillover effect of an attribute (e.g., closing price) by aggregating the states of other attributes (e.g., trading volume) of two relevant stocks.

However, the proposed attribute aggregator is still based on the ordinary graph structure, which can only explain the momentum spillovers between pairwise stocks. We propose to aggregate all stocks belonging to the same industry to learn the impact of industry attributes on the future stock trends. Considering industry attributes enables

model to more comprehensively adjust the momentum spillover effects between stocks. We use a feed-forward neural network with tanh activation function to aggregate the current state $\boldsymbol{v}_i^t$ of a given stock i and the states $\boldsymbol{v}_j^t$ of other stock nodes j on its hyperedge ε_m to get the information gate $c(\boldsymbol{v}_i^t, \varepsilon_m)$ of stock i:

$$c(\boldsymbol{v}_i^t, \varepsilon_m) = \tanh(\boldsymbol{W}_c[\boldsymbol{v}_i^t \oplus \boldsymbol{v}_{\varepsilon_m}^t] + \boldsymbol{b}_c), \tag{13}$$

where $\boldsymbol{W}_c \in R^{|\varepsilon_m|U\times F'}$ is a learnable weight matrix, ε_m is the hyperedge to which stock node v_i belongs, $|\varepsilon_m|$ denotes the number of all nodes on the hyperedge ε_m, and $\boldsymbol{b}_c \in R^{F'}$ is the bias vector. $\boldsymbol{v}_{\varepsilon_m}^t$ is the aggregation of all node features on the hyperedge ε_m:

$$\boldsymbol{v}_{\varepsilon_m}^t = \boldsymbol{v}_1^t + \boldsymbol{v}_2^t + \boldsymbol{v}_3^t + \cdots + \boldsymbol{v}_{|\varepsilon_m|}^t. \tag{14}$$

4.2.4. Multi-relation embedding and fusion

By combining the predefined hypergraph and the dynamic hypergraphs learned from stock time series data, we are able to describe the relations among stock nodes more accurately.

Firstly, we get the predefined stock industry relations hypergraph $\boldsymbol{H} \in R^{N\times E}$ from stock industry relations data and the dynamic hypergraph $\boldsymbol{H}_D^t \in R^{N\times\lambda}$ from the dynamic hypergraph construction module.

Then, we feed the dynamic hypergraph $\boldsymbol{H}_D^t$ and the stock temporal embedding $\boldsymbol{Z}^t = [\boldsymbol{z}_1^t, \ldots, \boldsymbol{z}_N^t] \in R^{N\times U}$ into the hypergraph convolutional layer with gate mechanism to extract the stock relational embedding $\boldsymbol{S}_1^t = [s_{11}^t, \ldots, s_{1N}^t] \in R^{N\times F'}$. The node update process is as follows:

$$\begin{aligned} s_{1i}^t &= \mathrm{HConv}\left(\boldsymbol{z}_i^t, \Theta^{(0)}, \boldsymbol{H}_D^t, c^{(0)}(\boldsymbol{z}_i^t, \varepsilon_m)\right) \\ &= \sigma\left(\sum_{j=1}^{N}\sum_{m=1}^{\lambda} \boldsymbol{H}_D^t(z_i, \varepsilon_m)\boldsymbol{H}_D^t(z_j, \varepsilon_m)\boldsymbol{z}_i^t\Theta^{(0)} \otimes c^{(0)}(\boldsymbol{z}_i^t, \varepsilon_m)\right), \end{aligned} \tag{15}$$

where $\Theta^{(0)} \in R^{U\times F'}$ denotes the parameter matrix and F' denotes the hidden size of the hypergraph convolutional layer. The node feature transformation and edge feature gathering processes are shown in Eqs. (10) and (11) respectively. Different from Eq. (12), the new node feature aggregating process is as follows:

$$\boldsymbol{S}_1^t = \sigma(\boldsymbol{H}_D^t \cdot \boldsymbol{P} \otimes c^{(0)}(\boldsymbol{Z}^t, \boldsymbol{H}_D^t)), \tag{16}$$

where $\boldsymbol{P} \in R^{\lambda\times F'}$ is the aggregated edge feature matrix.

Similarly, we input the predefined hypergraph $\boldsymbol{H}$ and the stock temporal embedding $\boldsymbol{Z}^t$ to extract the stock relational embedding $\boldsymbol{S}_2^t$:

$$s_{2i}^t = \mathrm{HConv}\left(\boldsymbol{z}_i^t, \Theta^{(1)}, \boldsymbol{H}, c^{(1)}(\boldsymbol{z}_i^t, \varepsilon_m)\right) \tag{17}$$

Finally, we combine $\boldsymbol{S}_1^t$ and $\boldsymbol{S}_2^t$ through learnable weights to get the final relational embedding of stocks:

$$\begin{aligned} \boldsymbol{S}^t &= \beta_1^*\boldsymbol{S}_1^t + \beta_2^*\boldsymbol{S}_2^t, \\ \beta_1^*, \beta_2^* &= \mathrm{softmax}(\beta_1, \beta_2), \end{aligned} \tag{18}$$

where β_1 and β_2 are learnable weight parameters. The fused stock relational embedding obtained from predefined and dynamic hypergraphs represents a more comprehensive representation of the spatial relations among stocks.

To stabilize the learning process of the model, we add a multi-head mechanism with K heads to the model for training, and concatenate the results:

$$\boldsymbol{s}_i^t = \Big\|_{k=1}^{K} (\beta_1^{*(k)}\boldsymbol{s}_{1i}^{t(k)} + \beta_2^{*(k)}\boldsymbol{s}_{2i}^{t(k)}). \tag{19}$$

4.2.5. Discussion of dynamic hypergraph network

The illustration of dynamic hypergraph network is shown in Fig. 2. Specifically, we first input the temporal embedding into the dynamic hypergraph construction module to obtain dynamic hypergraph. Then, we input the temporal embedding and hypergraph into the hypergraph convolutional layer with the relations aggregator, and obtain two types of stock relational embeddings from predefined and dynamic stock relations respectively. Finally, the multi-relation fusion module integrates the two relational embeddings.

In the dynamic hypergraph network, the purpose of the dynamic hypergraph construction module based on GAT is to learn dynamic hypergraphs to reflect stock higher-order spatial relations which are dynamically changing over time. The industry relations aggregator based on hypergraph aims to capture momentum spillovers among stocks more comprehensively by considering the impact of stock industry attributes on stock trends. The goal of the multi-relation fusion module is to fuse the relational embeddings of stocks learned from the static and dynamic hypergraph. Integrating these modules into a unified network allows for a more comprehensive consideration of dynamic spatial relations among stocks.

4.3. Output module and loss function

In the output mapping module, we employ a single-layer feed-forward neural network with a softmax function to generate the probabilities of future stock price trends. The probabilities can be represented as follows:

$$\hat{\boldsymbol{Y}}^{t+1} = O([\boldsymbol{Z}^t \oplus \boldsymbol{S}^t]) = \mathrm{softmax}([\boldsymbol{Z}^t \oplus \boldsymbol{S}^t]\boldsymbol{E} + \boldsymbol{b}), \tag{20}$$

where $\hat{\boldsymbol{Y}}^{t+1} \in R^{N\times C}$ is the probability matrix of stock node labels on day $t+1$, N is the total number of stocks, C is the number of classes, $\boldsymbol{E} \in R^{(U+KF')\times C}$ is the weight matrix and $\boldsymbol{b} \in R^C$ is the bias vector.

The parameters are optimized by minimizing the cross-entropy loss function:

$$\mathrm{L} = \frac{1}{N}\sum_{i=1}^{N}\mathrm{L_i} = -\frac{1}{N}\sum_{i=1}^{N}[\boldsymbol{y}_i^{t+1}\log(\hat{\boldsymbol{y}}_i^{t+1}) + (1-\boldsymbol{y}_i^{t+1})\log(1-\hat{\boldsymbol{y}}_i^{t+1})], \tag{21}$$

where $\boldsymbol{y}_i^{t+1}$ and $\hat{\boldsymbol{y}}_i^{t+1}$ denote the ground truth and predicted trend probability of stock i on day $t+1$, respectively. The cross-entropy loss of $\hat{\boldsymbol{Y}}^{t+1}$ and $\boldsymbol{Y}^{t+1}$ is back-propagated to learn the parameters of the model.

The comprehensive training process of the proposed DHSTN framework is shown in Algorithm 1.

Algorithm 1: The training process of DHSTN.

Input: stock data $\boldsymbol{x}_i^t$, ground truth trend labels of stock data $\boldsymbol{y}_i^{t+1}$, training epochs E, model parameters θ.
Output: θ^*.

```
Initialize model parameters θ.
for e = 1, 2...., E do
    for i = 1, 2...., N do
        Learn the temporal embedding of stock i z_i^t according to Eq. (4).
        for j = 1, 2...., N do
            Extract the dynamic higher-order relations for stock i H_D^t according to Eq. (7).
            Learn the information gate of stock i c(z_i^t, ε_m) according to Eq. (13).
            Learn the relational embedding of stock i s_i^t according to Eq. (19).
        end
        Output the predicted trend of stock i ŷ_i^{t+1} according to Eq. (20).
    end
    Calculate the loss function L and update parameters θ.
end
return θ*.
```

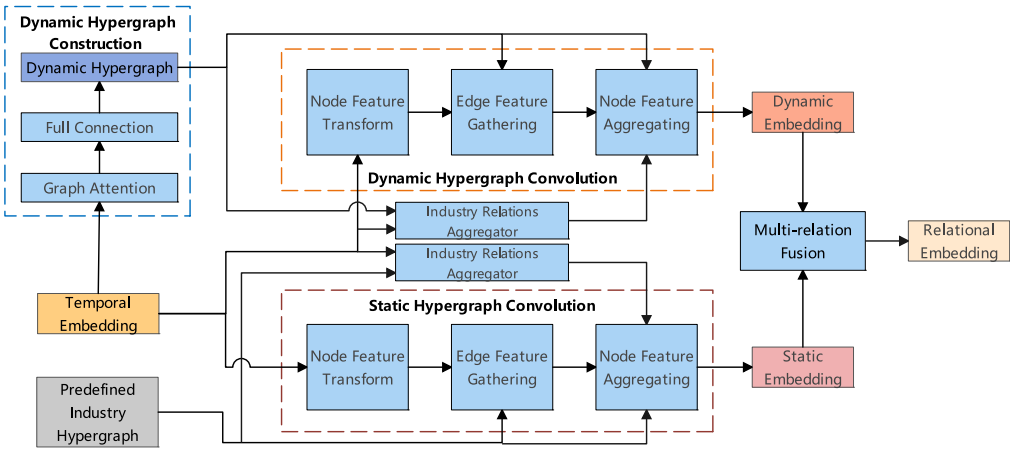


Fig. 2. The illustration of dynamic hypergraph network.

5. Experiments

5.1. Datasets

To validate the effectiveness of our proposed DHSTN framework, we evaluated it on two real-world datasets, CSI300 and NASDAQ100, both of which are publicly available. We collected historical stock price data from Yahoo Finance (https://finance.yahoo.com) and stock industry relations data from CSMAR (https://www.gtarsc.com). We collected the stock data from Chinese market (2017–07 to 2023–04) and American market (2017–09 to 2023–04) for a total of 1400 days and removed stocks with missing data during the 1400 days. As a consequence, we selected 230 stocks in CSI300 and 83 stocks in NASDAQ100. The descriptions of data are shown in Table 2. Stock trends are influenced by different types of indicators. Therefore, we selected the following three types of indicators to predict stock trends:

- **Stock Historical Data** : Historical price data of stocks depict the past price movements and contain information about future price trends. We extract five price attributes per day, including daily opening prices, closing prices, highest prices, lowest prices, and trading volume. Additionally, we calculated four trading indicators for each stock from the daily price data, which are the 5, 10, 20 and 30 day closing moving averages of the stock. They represent the daily, weekly and monthly price trends of the stock.
- **Alpha Technical Indicators** [55]: Alpha technical indicators are obtained by applying mathematical formulas to historical stock data, and now they have been widely used to predict the price movements of stocks. In this study, we selected 10 Alpha technical indicators of stocks as input data for the model. The details of the Alpha technical indicators are shown in Table 3.
- **Industry Relations Data**: In the real market, stocks in the same industry tend to have similar price movements. To capture this market signal, we collected static industry relations data for all stocks in CSI300 and NASDAQ100.

5.2. Rolling training, validation and testing

To effectively capture the sequential information presented in stock data, we adopt an approach with rolling training, validation, and testing in the learning and predicting process. The procedures of rolling training, validation and testing are illustrated in Fig. 3. We divide the total dataset into four parts and each part contains 800 trading days, of which the first 500 days are used for training, the next 100 days for validation, and the last 200 days for testing.

Table 2
Details of the dataset.

Dataset	Num of stocks	Date period	Trading days
CSI300	230	2017-07–2023-04	1400
NASDAQ100	83	2017-09–2023-04	1400

5.3. Evaluation metrics

We adopt Accuracy (ACC), the area under the precision–recall curve (AUC), Precision (PRE), Recall (REC) and F1-score (F1) to evaluate classification performance in our experiments. The ACC, PRE, REC and F1 are defined as follows:

- **Accuracy** :

$$\mathrm{ACC} = (\mathrm{TP}+\mathrm{TN})/(\mathrm{TP}+\mathrm{FP}+\mathrm{TN}+\mathrm{FN}), \tag{22}$$

where TP is the true positive, TN is the true negative, FP is the false positive and FN is the false negative.

- **Precision** :

$$\mathrm{PRE} = \mathrm{TP}/(\mathrm{TP}+\mathrm{FP}). \tag{23}$$

- **Recall** :

$$\mathrm{REC} = \mathrm{TP}/(\mathrm{TP}+\mathrm{FN}). \tag{24}$$

- **F1-score** :

$$\mathrm{F1} = 2(\mathrm{PRE}\times\mathrm{REC})/(\mathrm{PRE}+\mathrm{REC}). \tag{25}$$

To evaluate the investment performance of the methods in the real stock market, we adopt the following four commonly used performance indicators, including the final accumulated portfolio value (fAPV), the annualized rate of return (AR), sharpe ratio (SR) and maximum drawdown (MDD):

- **fAPV**: The final accumulated portfolio value measures the cumulative returns compared to the initial capital.

$$\mathrm{fAPV} = \frac{p_T}{p_0}, \tag{26}$$

where p_0 and p_T denote initial capital and final accumulated capital, respectively.

- **AR**: The annualized rate of return measures the annualized profitability of the investment method. In this study, the number of trading days in one year is assigned as 252.

$$\mathrm{AR}=\{(\frac{p_T}{p_0})^{252/T}-1\}\times 100\%. \tag{27}$$

Table 3
Details of the Alpha technical indicators.

Technical indicators	Specific meaning
RESI5	Residual of the linear regression of returns over the last 5 days
RSQR5	Sum of squared returns for the last 5 days
KLEN	Difference between the highest and lowest price of the day
KLOW	Difference between the opening and lowest price of the day
RET20	Return ratio for the last 20 days
ROC60	Price rate of change for the last 60 days
VSTD5	Standard deviation of trading volume over the last 5 days
BIAS20	Percent deviation between the closing price of the day and the average closing price of the past 20 days
STD5	Standard deviation of returns over the last 5 days
WVMA5	Moving average closing price over the last 5 days weighted by trading volume

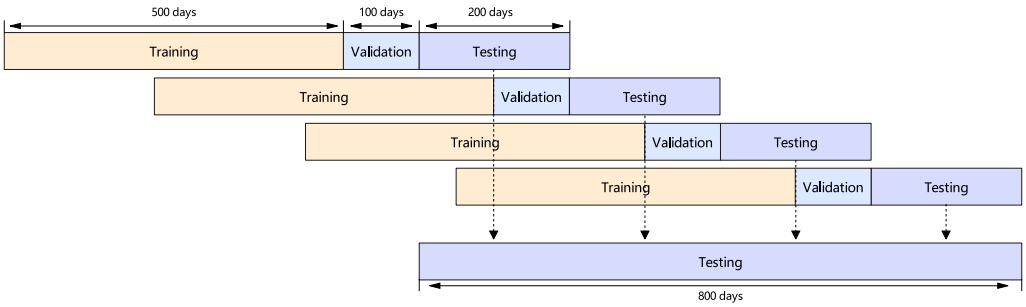


Fig. 3. Rolling training, validation and testing process.

- **SR**: Sharpe ratio measures the profitability of the investment method and take into account risk.

$$\mathrm{SR} = \frac{E_t[\rho_t - \rho_F]}{\sqrt{\mathrm{var}_t(\rho_t - \rho_F)}}, \tag{28}$$

where $\rho_t = \frac{p_t}{p_{t-1}} - 1$ is the return on the portfolio and p_F is the return on the risk-free asset which is always 0.

- **MDD**: Maximum drawdown measures the maximum decline in cumulative portfolio value from the peak.

$$\mathrm{MDD} = \max_{\tau\in(0,T)}(\max_{t\in(0,\tau)} \frac{p_t - p_\tau}{p_t}). \tag{29}$$

where p_t and p_τ represent capital at moment t and moment τ, respectively.

These metrics measure different aspects of stock investment performance, with higher fAPV and ARR indicating better profitability, higher SR indicating higher profits considering risk, and lower MDD indicating lower risk.

5.4. Baselines

To evaluate the performance of the proposed DHSTN framework, we compare it with the following representative stock prediction methods:

- **LSTM** [47]: This method learns long-term dependent information from sequential data and is used to predict stock trends based on historical price data.
- **Transformer** [26]: This method discards the sequential structure of traditional RNNs and introduces a self-attention mechanism to present global information in the data. Transformer has achieved success in the field of natural language processing and has full potential in the field of stock prediction.
- **CNN-Transformer** [56]: This method harnesses convolutional neural networks (CNNs) and Transformers to model both short-term and long-term dependencies within stock data series, and predict stock trends.
- **VMD-SVM-GWO** [57]: This method utilizes variational mode decomposition (VMD) to decompose the original stock data and then inputs them into SVM with grey wolf optimizer (GWO) to predict stock trends. This paper trains a VMD-SVM-GWO model individually for each stock.
- **GRU-GAT** [50]: This method is based on GRU and GAT, which firstly extracts temporal features from stock data by GRU; secondly, it dynamically assigns different weights to the relations of different nodes in the graph structure by GAT to model the complex relations in the stock market and extract relational features; finally, it combines temporal and relational features to predict stock trends.
- **GCN-LSTM** [35]: This method utilizes GCN to extract stock embeddings in multiple time-periods and then inputs the obtained temporal stock embeddings into LSTM to discriminate stock trends.
- **AD-GAT** [23]: This method uses GAT to update stock relations and designs a stock relations aggregator to adjust the transfer of relations between stocks in an attribute-sensitive manner.

5.5. Backtesting settings

To compare the profitability of all methods, and to consider the real situation of stock trading in different markets, we consider different trading rates in the Chinese and American markets and design two different simple trading strategies for CSI300 dataset with "T+1" trading system and NASDAQ100 dataset with "T+0" trading system. Our backtesting experiments are based on two assumptions: 1. It can be possible to complete the full transactions in the stock market to avoid trading failures. 2. Transaction slippage is considered in the transaction fees of the stock market. The assumption 1 ensures the efficacy and integrity of stock transactions in backtesting, and assumption 2 ensures a more accurate simulation of real stock trading environment in backtesting. More details of backtesting settings are as follows:

Table 4
Comparisons of predictive performance on CSI300 dataset and NASDAQ100 dataset ($\times 10^{-2}$).

	CSI300					NASDAQ100				
	AUC	ACC	F1	PRE	Rec	AUC	ACC	F1	PRE	Rec
LSTM	50.86	50.64	52.38	51.66	53.11	50.07	50.10	54.29	50.87	58.20
Transformer	50.91	50.61	50.60	51.75	49.49	50.16	50.22	52.94	51.04	55.00
CNN-Transformer	50.97	50.71	51.71	51.78	51.63	50.26	50.25	53.38	51.05	55.95
VMD-SVM-GWO	50.61	50.01	51.41	51.73	51.08	50.12	50.20	51.19	51.09	51.28
GRU-GAT	50.88	50.77	52.48	51.79	53.20	50.29	50.35	54.22	51.11	57.75
GCN-LSTM	51.06	50.75	51.96	51.80	52.11	50.18	50.42	54.38	51.16	58.03
AD-GAT	50.99	50.80	52.96	51.78	54.21	50.54	50.59	54.51	51.31	58.12
DHSTN (ours)	**51.51**	**51.34**	**58.30**	**51.87**	**66.57**	**51.26**	**51.38**	**57.23**	**51.83**	**63.89**

- **Trading Period**: From 2019–12 to 2023–04 on CSI300 dataset and from 2020–01 to 2023–04 on NASDAQ100 dataset.
- **Select Stock**: For CSI300 dataset, we select the top 50 stocks which are predicted to rise on trading day t, based on the probability of price rising. Similarly, for NASDAQ100 dataset, we select the top 5 stocks on trading day t.
- **Stop Loss Mechanism**: If the total number of stocks predicted to rise on a trading day is less than half of our chosen number, trading will stop.
- **Trading Strategy (CSI300)**: We buy the selected stocks with equal positions at the opening price on trading day t. After holding for 3 days, we sell them at the closing price on trading day $t+2$, following a cycle of three days for continuous trading.
- **Trading Strategy (NASDAQ100)**: We buy the selected stocks with equal positions at the opening price on trading day t and sell them at the closing price on trading day t, following a cycle of one day for continuous trading.

5.6. Hyper parameters and training settings

The hyperparameters of DHSTN include the window size T for the input time series, the hidden layer size U of the GRU, the hidden size F' of the hypergraph convolutional layer, the number of heads K in the multi-head mechanism, and the number of hyperedges λ in the dynamic hypergraph construction module. We perform a grid search method to select the optimal parameters. The range of values for T is {15, 20, 25, …, 45}, K is {2, 4, 6, 8, 10}, λ is {3, 5, 10, 15, 20}, U is {30, 60, 120, 240, 360} and F' is {30, 60, 120, 240, 360}. In the training process, the Adam optimizer [58] is utilized with an initial learning rate of 0.0005. Finally, through the grid search, the optimal window size for the time series is set to $T = 30$. The hidden size of the hypergraph convolutional layer is set to $F' = 60$. The hidden layer size of the GRU is set to $U = 360$. The number of heads in the multi-head mechanism is set to $K = 6$, and the number of hyperedges in the dynamic hypergraph construction module is set to $\lambda = 10$.

6. Results and analysis

6.1. Predictive performance

The results of different methods are shown in Table 4. As presented in the table, the comparisons on two datasets demonstrate that the proposed DHSTN can outperform other methods in all metrics.

VMD-SVM-GWO exhibits inferior performance in terms of ACC, AUC and F1 compared to other models, which may demonstrate that it cannot effectively capture temporal dependencies of stock time series data. LSTM, Transformer and CNN-Transformer only consider the sequential dependence between market signals, while GRU-GAT and GCN-LSTM take into account the sequential dependence between market signals and the structural information of the stock market. GRU-GAT and GCN-LSTM basically outperform LSTM, Transformer and CNN-Transformer in terms of ACC, AUC and F1, which means that considering the graph relations of stocks can lead to better results. Among all methods, AD-GAT has the second best performance in all metrics. The reason is that AD-GAT takes into account temporal information, structural information of markets, and attribute-sensitive momentum spillovers among stocks.

Table 5
Results of Wilcoxon signed-rank test on all datasets.

Compared models	Wilcoxon signed-rank test $\alpha = 0.05$ p-value	
Dataset	CSI300	NASDAQ100
DHSTN vs LSTM	0.0000**	0.0065
DHSTN vs Transformer	0.0000**	0.0000**
DHSTN vs CNN-Transformer	0.0000**	0.0002
DHSTN vs VMD-SVM-GWO	0.0000**	0.0000**
DHSTN vs GRU-GAT	0.0000**	0.0089
DHSTN vs GCN-LSTM	0.0000**	0.0163
DHSTN vs ADGAT	0.0000**	0.0000**

DHSTN implements dynamic hypergraph network to capture higher-order relations which are dynamically changed over time among stocks and considers the effect of overall stock industry attributes on stock movements. DHSTN outperforms the lower-order graph relations methods GRU-GAT and GCN-LSTM in all metrics, demonstrating the effectiveness of DHSTN in utilizing hypergraphs to represent higher-order relations. Compared with AD-GAT, DHSTN outperforms it with the improvements of at least 4.99% regarding to F1, 1.06% regarding to ACC and 0.88% regarding to AUC. The comparisons of AUC, ACC, and F1 are shown in Fig. 4.

Additionally, we calculate F1 of all stocks on each day and test model's predictive performance with Wilcoxon signed-rank test. As shown in Table 5, DHSTN significantly outperforms the other models with the significance level of $\alpha = 0.05$ on CSI300 and NASDAQ100 datasets.

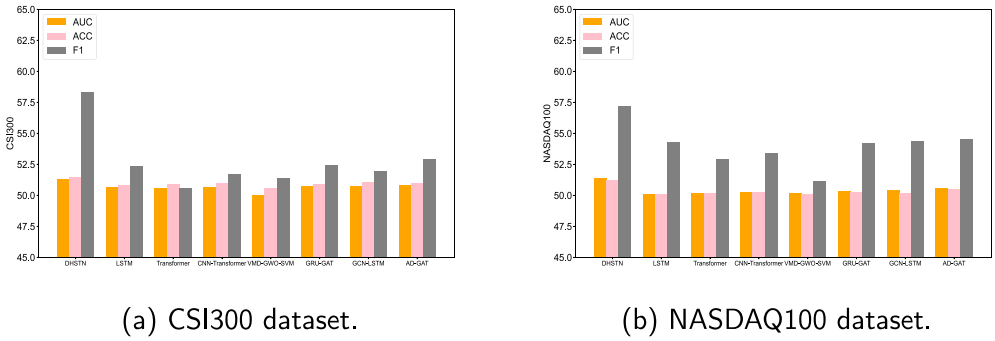
6.2. Backtesting performance

The returns of DHSTN and other methods over the trading time are shown in Fig. 5. The same trading strategy and initial capital are adopted for all methods over the trading time period. DHSTN has largely outperformed the other methods in terms of money returns in Chinese and American market. It shows that DHSTN selects stocks with more profitably than other methods.

The comparisons between DHSTN and other methods in the profitability metrics are shown in Table 6. After comparing the annualized rate of return, maximum drawdown and sharpe ratio of all methods, it shows that DHSTN outperforms other methods in all the profitability metrics for both CSI300 and NASDAQ100 datasets. DHSTN outperforms other methods by at least 47.9% in terms of SR and exhibits a reduction of at least 13.6% in terms of MDD.

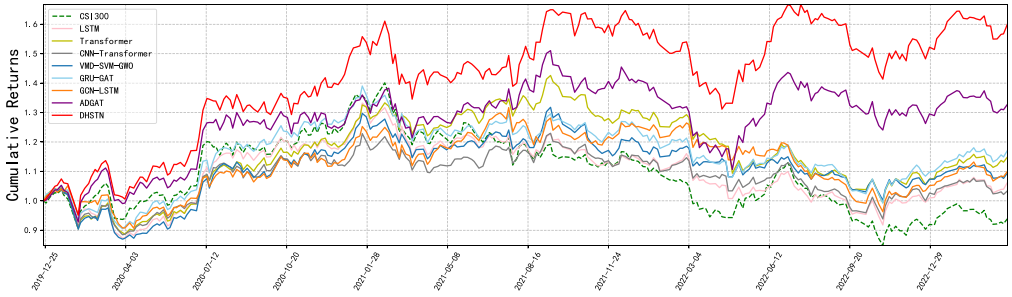
6.3. Ablation study

To analyze the effectiveness of the components in our proposed framework, we perform ablation experiments and compare them with the following experimental models:

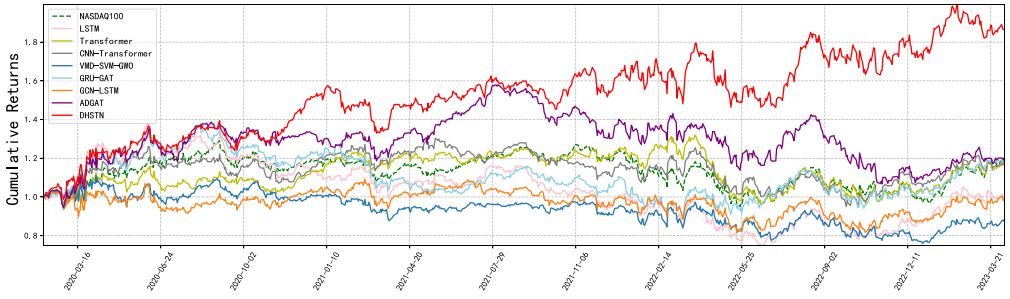


(a) CSI300 dataset. (b) NASDAQ100 dataset.

Fig. 4. Comparisons of AUC,ACC and F1.



(a) CSI300 Dataset



(b) NASDAQ100 Dataset

Fig. 5. Backtesting performance illustration of all methods.

Table 6
Comparisons of backtesting performance on CSI300 dataset and NASDAQ100 dataset ($\times 10^{-2}$).

	CSI300				NASDAQ100			
	MDD	SR	fAPV	AR	MDD	SR	fAPV	AR
LSTM	31.09	38.44	7.730	2.370	45.72	11.73	−0.900	−0.300
Transformer	30.31	62.05	15.79	4.740	27.33	34.06	17.38	5.170
CNN-Transformer	23.68	28.89	4.457	1.387	27.04	33.52	17.17	5.113
VMD-SVM-GWO	23.99	45.65	9.937	3.037	31.73	−6.637	−12.65	−4.169
GRU-GAT	28.60	69.50	18.48	5.500	34.92	35.42	19.57	5.780
GCN-LSTM	26.86	50.75	11.87	3.606	25.44	10.32	−1.281	−0.404
AD-GAT	26.74	97.48	34.57	9.830	32.41	35.84	20.08	5.920
DHSTN (ours)	**20.47**	**144.2**	**62.91**	**16.66**	**18.51**	**90.14**	**87.73**	**21.91**

- **DHSTN-DH**: The dynamic hypergraph construction module in DHSTN is removed and all other components are retained.
- **DHSTN-IRA**: The industry relations aggregator in DHSTN is removed and all other components are retained.
- **DHSTN-DH-IRA**: The dynamic hypergraph construction module and the industry relations aggregator in DHSTN are removed and all other components are retained.

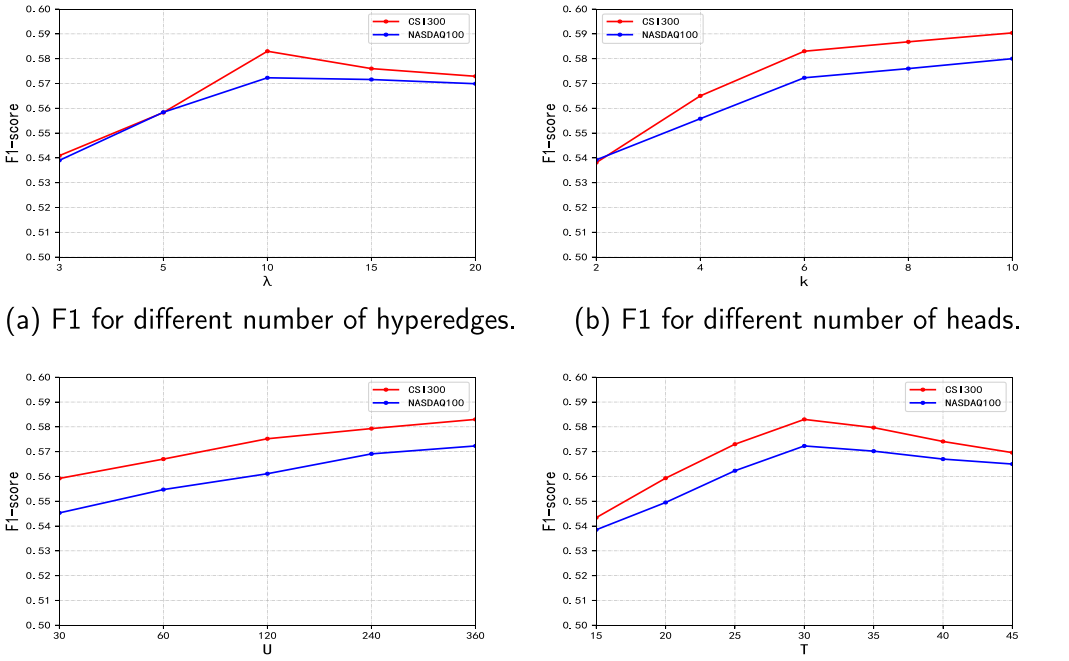
The performance comparisons of ablation experiments are presented in Table 7.

6.3.1. *Effectiveness of the dynamic hypergraph construction module*

The purpose of introducing the dynamic hypergraph construction module is to extract the dynamic higher-order relations among stocks. To evaluate the impact of the dynamic hypergraph construction module on the predictive performance of the model, we conduct experiments by removing this module. According to the results of the predictive metrics in Table 7, it is evident that the inclusion of the dynamic hypergraph construction module leads to an improvement in the model's predictive performance.

Table 7
Comparisons of ablation experiment results on CSI300 dataset and NASDAQ100 dataset ($\times 10^{-2}$).

	CSI300					NASDAQ100				
	AUC	ACC	F1	PRE	Rec	AUC	ACC	F1	PRE	Rec
DHSTN-DH-IRA	50.94	50.74	52.61	51.74	53.49	50.30	50.37	53.05	51.18	55.06
DHSTN-IRA	51.23	50.99	55.34	51.79	59.43	50.44	50.60	54.57	51.31	58.27
DHSTN-DH	50.95	50.91	54.94	51.75	58.55	50.55	50.48	54.27	51.22	57.70
DHSTN (ours)	**51.51**	**51.34**	**58.30**	**51.87**	**66.57**	**51.26**	**51.38**	**57.23**	**51.83**	**63.89**



(a) F1 for different number of hyperedges. (b) F1 for different number of heads.

(c) F1 for different number of hidden sizes. (d) F1 for different number of window sizes.

Fig. 6. DHSTN parameter analysis.

6.3.2. Effectiveness of the industry relations aggregator

Traditional linear aggregators based on GCNs fail to consider the attribute spillover effects of other stocks in the same industry when modeling relations among stocks. Consequently, they struggle to accurately capture the complex attribute spillover effects in the stock market. According to the results of the predictive metrics of DHSTN, DHSTN-DH and DHSTN-IRA in Table 7, it is evident that the predictive performance of DHSTN is improved by incorporating the industry relations aggregator.

6.4. Parameter analysis

- **Number of hyperedges** λ: Fig. 6(a) shows the F1-score of DHSTN on CSI300 and NASDAQ100 datasets for different number of hyperedges in the dynamic hypergraph construction module. The comparison shows that DHSTN exhibits better predictive performance when the number of hyperedges is set to 10. When the number of hyperedges is further increased, the predictive performance of DHSTN tends to be stable. Therefore, the number of hyperedges is set to 10 in this study.
- **Number of heads** K: Fig. 6(b) shows the predictive performance of DHSTN for different number of heads in the multi-head mechanism. Compared to a smaller number of heads, DHSTN has a significant improvement in prediction accuracy when the number of heads is set to 6. When the number of heads is further increased, the improvement of predictive performance is not significant, but the training time consumption increases significantly. Therefore, we set 6 heads in this study.
- **Number of hidden sizes** U: Fig. 6(c) shows the predictive performance of DHSTN for different number of hidden sizes of the GRU. The comparison shows that as the number of hidden sizes increases, the predictive performance of DHSTN becomes better. Therefore, the number of hidden sizes of the GRU is set to 360.
- **Number of window sizes** T: Fig. 6(d) shows the predictive performance of DHSTN for different number of window sizes for the input time series. When the number of window sizes is set to 30, the predictive performance of DHSTN improves significantly. When the number of window sizes increases further, the predictive performance of DHSTN decreases. Therefore, the number of window sizes for the input time series is set to 30.

6.5. Time complexity analysis

In this paper, the time to run one training epoch is utilized to represent the time complexity of the model. Given the individual training of a VMD-SVM-GWO model for each stock in this paper, the time for one such model is computed. As shown in Table 8, although the training time of our model is extended, considering the improved predictive performance, this increase is acceptable.

7. Conclusion

This paper proposes a dynamic hypergraph spatio-temporal network (DHSTN) for stock trend prediction, which combines temporal and spatial features of stocks. The proposed DHSTN utilizes the hypergraph neural network to capture relations among stocks, and addresses the limitation of traditional graph models in expressing higher-order

Table 8
Running time of models.

Model	Running time
LSTM	48 s for one training epoch.
Transformer	67 s for one training epoch.
CNN-Transformer	95 s for one training epoch.
VMD-SVM-GWO	1.5 s for one training epoch (one model).
GRU-GAT	74 s for one training epoch.
GCN-LSTM	79 s for one training epoch.
AD-GAT	107 s for one training epoch.
DHSTN(ours)	164 s for one training epoch.

multivariate relations among stocks. Furthermore, DHSTN employs a dynamic hypergraph construction module which enables model to extract dynamic hypergraphs from ever-changing stock time series data. Additionally, DHSTN utilizes an industry relations aggregator based on hypergraph structure which considers the impact of industry attributes on stock movement to realistically capture the relations among stocks. And a multi-relation fusion module is designed to integrate static and dynamic stock relations. Extensive experiments on two real world stock datasets validate the effectiveness of DHSTN.

The proposed model in this study still has two limitations. One is the lack of consideration for the impact of breaking news and investor sentiment on stock price trends. The other is the relatively high complexity of the dynamic hypergraph construction.

One direction for future research is to consider other sources of information, such as financial news and stock reviews, to enhance the performance of stock trend prediction model. And it is also valuable to explore reducing the complexity of the dynamic hypergraph construction to optimize running time of the model and using decomposition methods to capture stock data noises.

CRediT authorship contribution statement

Sihao Liao: Writing – review & editing, Writing – original draft, Software, Investigation. **Liang Xie:** Writing – review & editing, Methodology, Conceptualization. **Yuanchuang Du:** Writing – review & editing, Validation, Formal analysis. **Shengshuang Chen:** Writing – review & editing, Supervision. **Hongyang Wan:** Writing – review & editing, Visualization. **Haijiao Xu:** Funding acquisition, Data curation.

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

Data will be made available on request.

Acknowledgments

This work is supported in part by Natural Science Foundation of Guangdong Province, China [Grant No. 2020A1515011208], Science and Technology Program of Guangzhou, China [Grant No. 202102080353], and Characteristic Innovation Project of Guangdong Province, China [Grant No. 2019KTSCX117]. We thank all the anonymous reviewers who generously contributed their time and efforts. Their professional recommendations have greatly enhanced the quality of the manuscript.

References

[1] A.M. Rather, V. Sastry, A. Agarwal, Stock market prediction and portfolio selection models: a survey, Opsearch 54 (2017) 558–579, http://dx.doi.org/10.1007/s12597-016-0289-y.
[2] O. Bustos, A. Pomares-Quimbaya, Stock market movement forecast: A systematic review, Expert Syst. Appl. 156 (2020) 113464, http://dx.doi.org/10.1016/j.eswa.2020.113464.
[3] X. Yan, Z. Guosheng, Application of kalman filter in the prediction of stock price, in: Proceedings of the 5th International Symposium on Knowledge Acquisition and Modeling, Atlantis Press, 2015, pp. 197–198, http://dx.doi.org/10.2991/kam-15.2015.53.
[4] A.A. Adebiyi, A.O. Adewumi, C.K. Ayo, et al., Comparison of arima and artificial neural networks models for stock price prediction, J. Appl. Math. 2014 (2014) 614342, http://dx.doi.org/10.1155/2014/614342.
[5] R.M. Adnan, H.-L. Dai, R.R. Mostafa, K.S. Parmar, S. Heddam, O. Kisi, Modeling multistep ahead dissolved oxygen concentration using improved support vector machines by a hybrid metaheuristic algorithm, Sustainability 14 (6) (2022) 3470, http://dx.doi.org/10.3390/su14063470.
[6] Z. Zhang, W.-C. Hong, Application of variational mode decomposition and chaotic grey wolf optimizer with support vector regression for forecasting electric loads, Knowl.-Based Syst. 228 (2021) 107297, http://dx.doi.org/10.1016/j.knosys.2021.107297.
[7] Z. Zhang, W.-C. Hong, Electric load forecasting by complete ensemble empirical mode decomposition adaptive noise and support vector regression with quantum-based dragonfly algorithm, Nonlinear Dynam. 98 (2019) 1107–1136, http://dx.doi.org/10.1007/s11071-019-05252-7.
[8] R.M. Adnan, H.-L. Dai, R.R. Mostafa, A.R.M.T. Islam, O. Kisi, S. Heddam, M. Zounemat-Kermani, Modelling groundwater level fluctuations by elm merged advanced metaheuristic algorithms using hydroclimatic data, Geocarto Int. 38 (1) (2023) 2158951, http://dx.doi.org/10.1080/10106049.2022.2158951.
[9] R.M. Adnan, R.R. Mostafa, H.-L. Dai, S. Heddam, A. Kuriqi, O. Kisi, Pan evaporation estimation by relevance vector machine tuned with new metaheuristic algorithms using limited climatic data, Eng. Appl. Comput. Fluid Mech. 17 (1) (2023) 2192258, http://dx.doi.org/10.1080/19942060.2023.2192258.
[10] R.R. Mostafa, O. Kisi, R.M. Adnan, T. Sadeghifar, A. Kuriqi, Modeling potential evapotranspiration by improved machine learning methods using limited climatic data, Water 15 (3) (2023) 486, http://dx.doi.org/10.3390/w15030486.
[11] R.M. Adnan, R.R. Mostafa, A.R.M.T. Islam, O. Kisi, A. Kuriqi, S. Heddam, Estimating reference evapotranspiration using hybrid adaptive fuzzy inferencing coupled with heuristic algorithms, Comput. Electron. Agric. 191 (2021) 106541, http://dx.doi.org/10.1016/j.compag.2021.106541.
[12] M.M. Kumbure, C. Lohrmann, P. Luukka, J. Porras, Machine learning techniques and data for stock market forecasting: A literature review, Expert Syst. Appl. 197 (2022) 116659, http://dx.doi.org/10.1016/j.eswa.2022.116659.
[13] R.M.A. Ikram, R.R. Mostafa, Z. Chen, K.S. Parmar, O. Kisi, M. Zounemat-Kermani, Water temperature prediction using improved deep learning methods through reptile search algorithm and weighted mean of vectors optimizer, J. Mar. Sci. Eng. 11 (2) (2023) 259, http://dx.doi.org/10.3390/jmse11020259.
[14] X. Yuan, C. Chen, X. Lei, Y. Yuan, R.M. Adnan, Monthly runoff forecasting based on lstm-alo model, Stoch. Environ. Res. Risk Assess. 32 (2018) 2199–2212, http://dx.doi.org/10.1007/s00477-018-1560-y.
[15] S.-Y. Shih, F.-K. Sun, H.-y. Lee, Temporal pattern attention for multivariate time series forecasting, Mach. Learn. 108 (2019) 1421–1441, http://dx.doi.org/10.1007/s10994-019-05815-0.
[16] D. Daiya, C. Lin, Stock movement prediction and portfolio management via multimodal learning with transformer, in: Proceedings of the ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2021, pp. 3305–3309, http://dx.doi.org/10.1109/ICASSP39728.2021.9414893.
[17] K. Hou, Industry information diffusion and the lead–lag effect in stock returns, Rev. Financ. Stud. 20 (4) (2007) 1113–1138, http://dx.doi.org/10.1093/revfin/hhm003.
[18] Y. Ma, R. Mao, Q. Lin, P. Wu, E. Cambria, Multi-source aggregated classification for stock price movement prediction, Inf. Fusion 91 (2023) 515–528, http://dx.doi.org/10.1016/j.inffus.2022.10.025.
[19] J. Ye, J. Zhao, K. Ye, C. Xu, Multi-graph convolutional network for relationship-driven stock movement prediction, in: Proceedings of the 2020 25th International Conference on Pattern Recognition, ICPR, IEEE, 2021, pp. 6702–6709, http://dx.doi.org/10.1109/ICPR48806.2021.9412695.
[20] Z. Zhou, L. Zhang, R. Zha, Q. Hao, T. Xu, D. Wu, E. Chen, Multi-relational graph convolution network for stock movement prediction, in: Proceedings of the 2022 International Joint Conference on Neural Networks, IJCNN, IEEE, 2022, pp. 1–8, http://dx.doi.org/10.1109/IJCNN55064.2022.9892482.
[21] T. Wang, J. Guo, Y. Shan, Y. Zhang, B. Peng, Z. Wu, A knowledge graph-gcn-community detection integrated model for large-scale stock price prediction, Appl. Soft Comput. 145 (2023) 110595, http://dx.doi.org/10.1016/j.asoc.2023.110595.
[22] U. Ali, D. Hirshleifer, Shared analyst coverage: Unifying momentum spillover effects, J. Financ. Econ. 136 (3) (2020) 649–675, http://dx.doi.org/10.1016/j.jfineco.2019.10.007.

[23] R. Cheng, Q. Li, Modeling the momentum spillover effect for stock prediction via attribute-driven graph attention networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 55–62, http://dx.doi.org/10.1609/aaai.v35i1.16077.

[24] R. Akita, A. Yoshihara, T. Matsubara, K. Uehara, Deep learning for stock prediction using numerical and textual information, in: Proceedings of the 2016 IEEE/ACIS 15th International Conference on Computer and Information Science, ICIS, IEEE, 2016, pp. 1–6, http://dx.doi.org/10.1109/ICIS.2016.7550882.

[25] M.O. Rahman, M.S. Hossain, T.-S. Junaid, M.S.A. Forhad, M.K. Hossen, Predicting prices of stock market using gated recurrent units (grus) neural networks, Int. J. Comput. Sci. Netw. Secur. 19 (1) (2019) 213–222, http://paper.ijcsns.org/07_book/201901/20190126.pdf.

[26] Q. Ding, S. Wu, H. Sun, J. Guo, J. Guo, Hierarchical multi-scale gaussian transformer for stock movement prediction, in: Proceedings of the 29th International Joint Conference on Artificial Intelligence, 2021, pp. 4640–4646, https://dl.acm.org/doi/abs/10.5555/3491440.3492080.

[27] X. Teng, X. Zhang, Z. Luo, Multi-scale local cues and hierarchical attention-based lstm for stock price trend prediction, Neurocomputing 505 (2022) 92–100, http://dx.doi.org/10.1016/j.neucom.2022.07.016.

[28] H.J. Park, Y. Kim, H.Y. Kim, Stock market forecasting using a multi-task approach integrating long short-term memory and the random forest framework, Appl. Soft Comput. 114 (2022) 108106, http://dx.doi.org/10.1016/j.asoc.2021.108106.

[29] Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, P. Liu, Transformer-based attention network for stock movement prediction, Expert Syst. Appl. 202 (2022) 117239, http://dx.doi.org/10.1016/j.eswa.2022.117239.

[30] M. Li, Y. Zhu, Y. Shen, M. Angelova, Clustering-enhanced stock price prediction using deep learning, World Wide Web 26 (1) (2023) 207–232, http://dx.doi.org/10.1007/s11280-021-01003-0.

[31] Y. Zhao, G. Yang, Deep learning-based integrated framework for stock price movement prediction, Appl. Soft Comput. 133 (2023) 109921, http://dx.doi.org/10.1016/j.asoc.2022.109921.

[32] F. Scarselli, M. Gori, A.C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Trans. Neural Netw. Learn. Syst. 20 (1) (2008) 61–80, http://dx.doi.org/10.1109/TNN.2008.2005605.

[33] 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, http://dx.doi.org/10.1145/3269206.3269269.

[34] D. Cheng, F. Yang, S. Xiang, J. Liu, Financial time series forecasting with multi-modality graph neural network, Pattern Recognit. 121 (2022) 108218, http://dx.doi.org/10.1016/j.patcog.2021.108218.

[35] Y. Shi, Y. Wang, Y. Qu, Z. Chen, Integrated gcn-lstm stock prices movement prediction based on knowledge-incorporated graphs construction, Int. J. Mach. Learn. Cybern. 15 (2023) 161–176, http://dx.doi.org/10.1007/s13042-023-01817-6.

[36] Y. Feng, H. You, Z. Zhang, R. Ji, Y. Gao, Ypergraph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 3558–3565, http://dx.doi.org/10.1609/aaai.v33i01.33013558.

[37] C. Chen, Z. Cheng, Z. Li, M. Wang, Hypergraph attention networks, in: Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom, IEEE, 2020, pp. 1560–1565, http://dx.doi.org/10.1109/TrustCom50675.2020.00215.

[38] H. Shi, Y. Zhang, Z. Zhang, N. Ma, X. Zhao, Y. Gao, J. Sun, Hypergraph-induced convolutional networks for visual classification, IEEE Trans. Neural Netw. Learn. Syst. 30 (10) (2018) 2963–2972, http://dx.doi.org/10.1109/TNNLS.2018.2869747.

[39] Y. Huang, Q. Liu, D. Metaxas, Video object segmentation by hypergraph cut, in: Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 1738–1745, http://dx.doi.org/10.1109/CVPR.2009.5206795.

[40] Y. Luo, J. Hu, X. Wei, D. Fang, H. Shao, Stock trends prediction based on hypergraph modeling clustering algorithm, in: Proceedings of the 2014 IEEE International Conference on Progress in Informatics and Computing, IEEE, 2014, pp. 27–31, http://dx.doi.org/10.1109/PIC.2014.6972289.

[41] R. Sawhney, S. Agarwal, A. Wadhwa, R.R. Shah, Spatiotemporal hypergraph convolution network for stock movement forecasting, in: Proceedings of the 2020 IEEE International Conference on Data Mining, ICDM, IEEE, 2020, pp. 482–491, http://dx.doi.org/10.1109/ICDM50108.2020.00057.

[42] 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, http://dx.doi.org/10.1609/aaai.v35i1.16127.

[43] X. Li, C. Cui, D. Cao, J. Du, C. Zhang, Hypergraph-based reinforcement learning for stock portfolio selection, in: Proceedings of the ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2022, pp. 4028–4032, http://dx.doi.org/10.1109/ICASSP43922.2022.9747138.

[44] J. Jiang, Y. Wei, Y. Feng, J. Cao, Y. Gao, Dynamic hypergraph neural networks, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019, pp. 2635–2641, https://dl.acm.org/doi/abs/10.5555/3367243.3367406.

[45] P. Mondal, L. Shit, S. Goswami, Study of effectiveness of time series modeling (arima) in forecasting stock prices, Int. J. Comput. Sci. Eng. Appl. 4 (2) (2014) 13, http://dx.doi.org/10.5121/ijcsea.2014.4202.

[46] A. Moghar, M. Hamiche, Stock market prediction using lstm recurrent neural network, Procedia Comput. Sci. 170 (2020) 1168–1173, http://dx.doi.org/10.1016/j.procs.2020.03.049.

[47] D.M. Nelson, A.C. Pereira, R.A. De Oliveira, Stock market's price movement prediction with lstm neural networks, in: Proceedings of the 2017 International Joint Conference on Neural Networks, IJCNN, IEEE, 2017, pp. 1419–1426, http://dx.doi.org/10.1109/IJCNN.2017.7966019.

[48] S. Li, J. Wu, X. Jiang, K. Xu, Chart gcn: Learning chart information with a graph convolutional network for stock movement prediction, Knowl.-Based Syst. 248 (2022) 108842, http://dx.doi.org/10.1016/j.knosys.2022.108842.

[49] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder–decoder for statistical machine translation, 2014, http://dx.doi.org/10.48550/arXiv.1406.1078.

[50] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, et al., Graph attention networks, 2017, http://dx.doi.org/10.48550/arXiv.1710.10903.

[51] Z. Zhang, H. Lin, Y. Gao, K. BNRist, Dynamic hypergraph structure learning, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3162–3169, https://dl.acm.org/doi/abs/10.5555/3304889.3305100.

[52] S. Shi, J. Li, G. Li, P. Pan, Q. Chen, Q. Sun, Gpm: A graph convolutional network based reinforcement learning framework for portfolio management, Neurocomputing 498 (2022) 14–27, http://dx.doi.org/10.1016/j.neucom.2022.04.105.

[53] W. Chen, M. Jiang, W.-G. Zhang, Z. Chen, A novel graph convolutional feature based convolutional neural network for stock trend prediction, Inform. Sci. 556 (2021) 67–94, http://dx.doi.org/10.1016/j.ins.2020.12.068.

[54] X. Yin, D. Yan, A. Almudaifer, S. Yan, Y. Zhou, Forecasting stock prices using stock correlation graph: A graph convolutional network approach, in: Proceedings of the 2021 International Joint Conference on Neural Networks, IJCNN, IEEE, 2021, pp. 1–8, http://dx.doi.org/10.1109/IJCNN52387.2021.9533510.

[55] Z. Kakushadze, 101 Formulaic alphas, Wilmott 2016 (84) (2016) 72–81, http://dx.doi.org/10.1002/wilm.10525.

[56] Z. Zeng, R. Kaur, S. Siddagangappa, S. Rahimi, T. Balch, M. Veloso, Financial time series forecasting using cnn and transformer, 2023, http://dx.doi.org/10.48550/arXiv.2304.04912.

[57] R. Liu, G. Li, L. Wei, Y. Xu, X. Gou, S. Luo, X. Yang, Spatial prediction of groundwater potentiality using machine learning methods with grey wolf and sparrow search algorithms, J. Hydrol. 610 (2022) 127977, http://dx.doi.org/10.1016/j.jhydrol.2022.127977.

[58] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2014, http://dx.doi.org/10.48550/arXiv.1412.6980.