Temporal and Heterogeneous Graph Neural Network for Financial Time Series Prediction

Sheng Xiang
AAII
University of Technology Sydney
Sydney, Australia
sheng.xiang@student.uts.edu.au

Dawei Cheng*
Department of Computer Science
Tongji University
Shanghai, China
dcheng@tongji.edu.cn

Chencheng Shang
HSBC Business School
Peking University
Beijing, China
pkuscc@stu.pku.edu.cn

Ying Zhang
AAII
University of Technology Sydney
Sydney, Australia
ying.zhang@uts.edu.au

Yuqi Liang
Seek Data Group
Emoney Inc.
Shanghai, China
roly.liang@seek-data.com

ABSTRACT

The price movement prediction of stock market has been a classical yet challenging problem, with the attention of both economists and computer scientists. In recent years, graph neural network has significantly improved the prediction performance by employing deep learning on company relations. However, existing relation graphs are usually constructed by handcraft human labeling or nature language processing, which are suffering from heavy resource requirement and low accuracy. Besides, they cannot effectively response to the dynamic changes in relation graphs. Therefore, in this paper, we propose a temporal and heterogeneous graph neural network-based (THGNN) approach to learn the dynamic relations among price movements in financial time series. In particular, we first generate the company relation graph for each trading day according to their historic price. Then we leverage a transformer encoder to encode the price movement information into temporal representations. Afterward, we propose a heterogeneous graph attention network to jointly optimize the embeddings of the financial time series data by transformer encoder and infer the probability of target movements. Finally, we conduct extensive experiments on the stock market in the United States and China. The results demonstrate the effectiveness and superior performance of our proposed methods compared with state-of-the-art baselines. Moreover, we also deploy the proposed THGNN in a real-world quantitative algorithm trading system, the accumulated portfolio return obtained by our method significantly outperforms other baselines.

CCS CONCEPTS

• **Information systems → Data mining**.

*Corresponding Author.

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557089

KEYWORDS

Graph Neural Network, Financial Time Series, Stock Movement Prediction, Heterogeneous Graph.

ACM Reference Format:
Sheng Xiang, Dawei Cheng, Chencheng Shang, Ying Zhang, and Yuqi Liang. 2022. Temporal and Heterogeneous Graph Neural Network for Financial Time Series Prediction. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557089

1 INTRODUCTION

Stock market is a financial ecosystem that involves transactions between companies and investors, with a global market capitalization of more than $83.5 trillion as of 2020 [33]. The Efficient Market Hypothesis [31] points out that the stock price represents all the available information, but the stock price is volatile in nature, resulting in difficulty on predicting its movement [1]. In recent years, deep learning has been widely used in predicting the price movement of stocks [24]. Researchers also explore to improve the prediction performance by incorporating more sources as model inputs, including more technical indicators [31], factors [14], financial status [2], online news [27], social media posts [40], etc.

Traditional learning methods treat the time series as independent and identically distributed to each other, which is not coincident to the real situation in the financial market [26]. For example, two stocks in the same sector may have a higher correlation than those in different fields [4]. Therefore, recent studies employ knowledge graphs to represent the internal relations among entities [7**?**] and leverage graph learning for the price movement prediction [28]. These works have shown the effectiveness by integrating stock's relations in the prediction models [10]. Thus, graph-based methods could benefit from learning meaningful representations of inputs, resulting in better prediction accuracy [9, 35].

However, generating relation graphs of entities is very challenging because it is fluctuate, noisy, amphibolous and dynamically changed [25]. For example, financial knowledge graphs mainly include the supply chain, primary business and investment relations of entities [15], which is labeled by domain experts or extracted from unstructured texts. But different experts have different knowledge

background which may lead to different relation graphs. Besides, the current nature language processing (NLP) techniques are still facing significant shortcomings in high accuracy relation extraction [12]. In other words, the relations may be misled by either unilateral text news or inaccurate extracting models. In addition, these relations may dynamically change in time series. For example, the main business of a company would change according to the market demands, and the supply chain graph would be upgraded because of the technique evolution [19]. Existing graph learning price prediction methods are inevitably suboptimal in learning these fluctuate and dynamical situations.

To address the above challenges, we propose a novel temporal and heterogeneous graph neural network-based method for financial time series prediction. Specifically, we directly model the relations of price time series of entities based on historical data and represent them in a temporal and heterogeneous graph, i.e., company relational graph. After obtaining the company relational graph, we leverage sequential transformers encode the historical prices and graph neural networks to encode internal relations of each company. Specifically, we update each company's representations by aggregating information from their neighbors in company relational graphs in two steps. The first step is a time-aware graph attention mechanism. The second is a heterogeneous graph attention mechanism. We thoroughly evaluate our approach on both the S&P 500[1] and CSI 300[2] dataset in the United States and China's stock markets. The experimental results show that our method significantly outperforms state-of-the-art baselines. In order to keep the sententious of our model, we conduct ablation studies to prove the effectiveness and essential of the each component of our method, including transformer encoder, time-aware graph attention, heterogeneous graph attention. Finally, we deploy our model in real-world quantitative algorithm trading platform, hosted in EMoney Inc.[3], a leading financial service provider in China. The cumulative returns of portfolios contributed by our approach is significantly better than existing models in financial industry. We will release the dataset as well as the source codes of the proposed techniques along with the paper. In conclusion, our principle contributions are summarized as follows:

- We propose a graph learning framework to effectively model the internal relations among entities for financial time series prediction, which fits the dynamical market status and is concordant with the ground-truth price movements.
- We design a temporal and heterogeneous graph neural network model to learn the dynamic relationships by two-stage attention mechanisms. The proposed model is concise and effective in joint and automatically learning from historical price sequence and internal relations.
- Our proposed THGNN is simple and can be easily implemented in the industry-level system. Extensive experiments on both the Unite States and China stock markets demonstrate the superior performance of our proposed methods. We also extensively evaluated its effectiveness by real-world trading platform.

[1]https://www.spglobal.com/spdji/en/indices/equity/sp-500
[2]http://www.cffex.com.cn/en_new/CSI300IndexOptions.html
[3]http://www.emoney.cn/

2 RELATED WORKS

2.1 Financial Time Series Learning

It is widely known that price movements of stocks are affected by various aspects in financial market [14]. In previous studies, a common strategy is to manually construct various factors as feature inputs [8, 30]. For example, Michel et. al, [2] integrate market signals with stock fundamental and technical indicators to make decisions. Li et. al, [25] establish a link between news articles and the related entities for stock price movement forecasting. A large number of existing methods employ recurrent neural network and its variants, such as LSTM [17] and GRU [11], to learn the sequential latent features of historical information and employ them for downstream prediction task [21].

In these works, the market signals processing of each stock is carried out independently. However, this inevitably ignores the internal relationship among stocks and would lead to suboptimal performance. Some works [16] leverage the correlation information as model inputs, but cannot automatically capture the dynamic changes of relations. In this article, we model the relationship between stocks as dynamic company relation graphs and joint learn the graph relation and historical sequence feature automatically for future price movement prediction.

2.2 Graph Learning for Stock Prediction

Researchers have shown that the price movement of stocks is not only related to its own historical prices, but also connect to its linked stocks [10]. The link relation includes suppliers and customers, shareholders and investor, etc. Existing works normally employ knowledge graphs to store and represent these relations [8, 29]. Recently, graph neural network (GNN) [41] is proposed to effectively learn on graph-structured data, which has shown its superior performance in various domains, including fraud detection [5, 6], computer vision [37, 38], etc. Researchers also introduce the advanced GNN-based approaches in the stock price prediction task. For example, Chen et. al, [4] models the supply chain relationships of entities into knowledge graphs and uses graph convolution networks to predict stock movements. Ramit et. al, [36] leverage attentional graph neural network on the connections constructed by social media texts and company correlations. Cheng et.al, [8] leverage multi-modality graph neural network on the connections constructed by historical price series, media news, and associated events.

However, the graph constructed by these methods are limited by constant, predefined corporate relationships, which is powered by handcraft editing or nature language processing techniques, suffering from heavy resources labeling and low extracting accuracy [20]. But the actual corporate diagram evolves frequently over time. Besides, the company relation graph is also heterogeneous, which means there are multiple relation types among entities. Therefore, existing methods cannot exploit the full information from real-life company relation graphs. In this paper, we construct the relation graph dynamically and automatically based on their ground-truth historical price sequences and then propose a novel temporal and heterogeneous graph neural network methods to jointly learn their sequential and relational features for more accurate stock price

Table 1: The summary of symbols

Symbol	Definition
$\mathbf{X}$	the historical price of listed companies
$\hat{\mathbf{Y}}$	the probability of price movements
n	the total number of nodes
m	the total number of edges
r	the number of relationships in graph $\tilde{G}$
T	the number of trading days in $\tilde{G}$
$\tilde{G} = (\tilde{V}, \{\tilde{E}_{r_1}, \tilde{E}_{r_2}, ...\})$	the temporal and heterogeneous graph
$\tilde{V} = \{v_1^{t_{v_1}}, ..., v_n^{t_{v_n}}\}$	the set of temporal nodes
$\tilde{E}_r = \{e_1^{t_{e_1}}, ..., e_m^{t_{em}}\}_r$	the set of temporal edges of relation r
$\mathcal{N}(\cdot)$	the neighborhood function
d	the number of dimension

prediction. We demonstrate the effectiveness of our methods by extensive experiments and real-world trading platform.

3 THE PROPOSED METHOD

In this section, we introduce the framework of our proposed temporal and heterogeneous graph neural network and their each component in detail. Our model takes the historical price sequence as inputs and infer the probability of stock movements as outputs. We represent the relation of stocks in a dynamic heterogeneous graph with two types of edges. We then jointly encode the historical and relation features by transformers and heterogeneous graph attention network. We report the problem definition first and each module of our method in turn.

3.1 Problem Definition

Different from traditional graph-based methods that construct static and homogeneous graphs by handcraft labeling or nature language processing techniques to infer stock movements, our model represents the company relation graph as a collection of temporal and heterogeneous graphs, which are automatically generated by historical price sequences. In graphs, the node denotes each equity and edge represents their relations in sequential. Temporal graphs are composed of timestamped edges and timestamped nodes [34, 45]. Each node might be associated with multiple relationships and multiple timestamped edges on different trading days. There are multiple types of edge $E = \{E_1, ..., E_r\}$ in company relation graphs. And the occurences of node $V = \{V^{t_1}, ..., V^T\}$ and edge $E = \{E^{t_1}, ..., E^T\}$ are different on different trading days. Table 1 summarizes the symbols introduced in this paper.

Definition 3.1. **Temporal and Relational Occurrence.** *In a temporal company relation graph, an edge e is associated with a series of temporal occurrences $e = \{e^{t_1}, e^{t_2}, ...\}$, which indicate the occurrences of edge e at trading days $\{t_1, t_2, ...\}$ in the company relation graph. Each type of relational occurrence is associated with a series of temporal occurrences with $E = \{E_{r_1}, E_{r_2}, ...\}$, which indicate the occurrences of edge e in different relationships $\{r_1, r_2, ...\}$. Same as temporal occurrences of edge e, a node v is associated with a set of temporal occurrences with $v = \{v^{t_1}, v^{t_2}, ...\}$.*

Definition 3.2. **Temporal and Heterogeneous Graph.** *A temporal and heterogeneous company relation graph $\tilde{G} = (\tilde{V}, \tilde{E})$ is formed by a set of temporal nodes $\tilde{V} = \{v_1^{t_{v_1}}, ..., v_n^{t_{vn}}\}$ and a series of sets of temporal edges $\tilde{E} = \{\tilde{E}_1, ..., \tilde{E}_r\}$, where $\tilde{E}_r = \{e_1^{t_{e_1}}, ..., e_m^{t_{em}}\}_r$ denotes the edges of relation r, and $e_i^{t_{e_i}} = (u_{e_i}, v_{e_i})^{t_{e_i}}$ denotes the temporal edge.*

In existing works [4, 10, 15], the graph neighborhood $\mathcal{N}(v)$ of node v is defined as static or homogeneous. Here, we generalize the definition of graph neighborhood to the temporal and heterogeneous graph, which is set as follows:

Definition 3.3. **Temporal and Heterogeneous Graph Neighborhood.** *Given a temporal node v, the neighborhood of v is defined as* $\mathcal{N}(v) = \{v_i | f_{sp}(v_i, v) \leq d_{\mathcal{N}}, |t_v - t_{v_i}| \leq t_{\mathcal{N}}\}$, where $f_{sp}(\cdot|\cdot)$ denotes the shortest path length between two nodes, $d_{\mathcal{N}}$ and $t_{\mathcal{N}}$ denote the hyper-parameters. As for heterogeneous graph, we define $\mathcal{N}_r(\cdot)$ as the neighborhood function of relation r.

Finally, we formally define the stock movement prediction problem as follows:

Input: *Historical price sequences of listed companies* $\mathbf{X} = \{x_1, x_2, \cdots, x_n\}$, *where each* $x_i = \{x_{i,1}, x_{i,2}, \cdots, x_{i,t}\}$ *denotes the historical price sequences. We then use the* $\mathbf{X}$ *to generate the temporal and heterogeneous company relation graph $\tilde{G}$, with multiple types of temporal edges $\{\tilde{E}_{r_1}, \tilde{E}_{r_2}, ...\}$, for downstream tasks .*

Output: *The probability $\hat{\mathbf{Y}}$ of price movements of each equity.*

3.2 Stock Correlation Graph Generation

In this subsection, we report the process of the temporal and heterogeneous graph construction. As mentioned in previous studies [4, 22], there may be multiple relationships between companies (such as suppliers and customers, shareholders and invested companies). Different from conventional knowledge graph-based relations that construct relation by human labeling or NLP techniques, generating relations directly based on market trend signals has proved to be effective [25, 42] in practical, which does not require additional ambiguity domain knowledge or text news sources, and is easy to be implemented.

Therefore, in this paper, we obtain the correlation matrix by calculating the correlation coefficients between ground-truth stock historical market signals directly. Then, the relationship between companies is determined according to the value of each element of the correlation matrix. The relationship between companies may be positive (correlation > threshold) or negative (correlation < -threshold). In order to reduce noise, we connect the edges whose absolute value is greater than the threshold, and the rest of the edges are considered not to be connected. So far, the edges E of company relation graph are generated. Therefore, we model the inter-company relation graph as a heterogeneous graph with two relationships, i.e., $G = (V, \{E_{r_1}, E_{r_2}, ...\})$, with $r \in \{pos, neg\}$.

As the relationships between companies tend to be dynamic, we generate the company relation graph in temporal format as our model's inputs. In particular, within T trading days, we generate temporal and heterogeneous company relation graphs with T timestamps, which is formulated as $\tilde{G} = (\tilde{V}, \{\tilde{E}_{pos}, \tilde{E}_{neg}\})$. Finally, the generated graphs and original sequence inputs are fed to downstream learning task simultaneously.

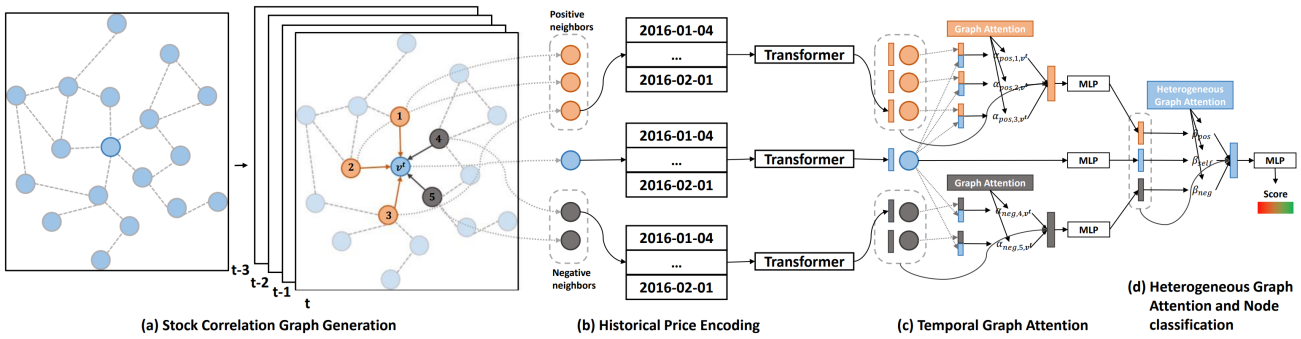


Figure 1: The proposed Temporal and Heterogeneous Graph Neural Networks architecture for stock movement predictions. The first part is the generation of a stock correlation graph, which builds dynamic relations for stocks in the market every trading day. The second part is the historical price encoding, which selects a temporal node v^t and its neighbor nodes to encode the historical price information. Transformer encoders share their parameters. The third part is the graph attention layer, which adaptively calculates the importance of the neighbors and aggregates the information according to the neighbors' importance. The fourth part is the heterogeneous graph attention layer, which adaptively calculates the importance and aggregates information from different types of neighbors. Then, we leverage a multi-layer perception to give the prediction of each stock's future movement.

3.3 Historical Price Encoding

The input stock movement feature of price sequences is defined as $\mathbf{X}^t \in \mathbb{R}^{n \times T \times d_{feat}}$ on trading day t, where n denotes the number of stocks, T means the number of trading days before t, and d_{feat} denotes the dimension of historical price features. We first leverage a linear transformation and positional encoding (PE) [39, 44] on trading features to obtain the input tensor $\mathbf{H}^t \in \mathbb{R}^{n \times T \times d_{in}}$, which is formulated as follows:

$$\begin{aligned} \hat{\mathbf{H}}^t &= \mathbf{W}_{in}\mathbf{X}^t + \mathbf{b}_{in} \\ \mathbf{H}^t &= \hat{\mathbf{H}}^t + \mathrm{PE} \\ \mathrm{PE}(p, 2i) &= sin(p/10000^{2i/d_{in}}) \\ \mathrm{PE}(p, 2i+1) &= cos(p/10000^{2i/d_{in}}) \end{aligned} \tag{1}$$

where $p \in \{1, 2, .., T\}$ is the trading day position, i is the dimension, d_{in} denotes the dimension of input features, and $\mathbf{W}_{in} \in \mathbb{R}^{d_{feat} \times d_{in}}$ and $\mathbf{b}_{in} \in \mathbb{R}^{d_{in}}$ denote the learnable parameters. After linear transformation, we proposed to leverage multi-head attentional transformer to encode the input feature for each stock in each day. Then, the proposed encoder outputs $\mathbf{H}^t_{enc} \in \mathbb{R}^{n \times T \times d_{enc}}$ for downstream tasks, where d_{enc} denotes the output dimension of the encoder. Mathematically, we formulate the historical feature encoder's output $\mathbf{H}^t_{enc}$ as follows:

$$\mathbf{H}^t_{enc} = \mathrm{Concat}(\mathrm{EncHead}^t_1, \ldots, \mathrm{EncHead}^t_{h_{enc}})\mathbf{W}_o \tag{2}$$

where $\mathbf{W}_o \in \mathbb{R}^{h_{enc} d_v \times d_{enc}}$ denotes the output projection matrix, h_{enc} denotes the number of heads in the encoder, d_v denotes the output dimension of each head, Concat denotes a concatenation of the output of heads, and $\mathrm{EncHead}^t_i \in \mathbb{R}^{n \times T \times d_v}$ denotes the output of encoder head with $\mathrm{EncHead}^t_i = \mathrm{Attention}(\mathbf{Q}^t_i, \mathbf{K}^t_i, \mathbf{V}^t_i)$, which is formulated as follows:

$$\begin{aligned} \mathbf{Q}^t_i = \mathbf{H}^t\mathbf{W}^Q_i, \mathbf{K}^t_i &= \mathbf{H}^t\mathbf{W}^K_i, \mathbf{V}^t_i = \mathbf{H}^t\mathbf{W}^V_i \\ \mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \mathrm{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{in}}})\mathbf{V} \end{aligned} \tag{3}$$

where $\mathbf{W}^Q_i \in \mathbb{R}^{d_{in} \times d_{hidden}}$, $\mathbf{W}^K_i \in \mathbb{R}^{d_{in} \times d_{hidden}}$, $\mathbf{W}^V_i \in \mathbb{R}^{d_{in} \times d_v}$ denote the projection matrices, and d_{hidden} denotes the dimension of hidden layer.

3.4 Temporal Graph Attention Mechanism

Given the historical sequence encoder output $\mathbf{H}^t_{enc}$ and temporal relation graph $\tilde{G}$, we propose to employ graph attention mechanism on the sequential and heterogeneous inputs. In particular, we flatten the embeddings of all nodes to $\mathbf{H}^t_{enc} \in \mathbb{R}^{n \times T d_{enc}}$ and leverage two-stage temporal attention mechanism to aggregate messages from graph structures and temporal sequences, which is illustrative reported in Figure 1 (c). The two-stage temporal graph attention layers could aggregate messages from both the postive and negative neighbors simultaneously. For each relationship $r \in \{pos, neg\}$, the message aggregating is formulated as follows:

$$\mathbf{H}^t_r = \mathrm{Concat}(\mathrm{TgaHead}_1, ..., \mathrm{TgaHead}_{h_{tga}})\mathbf{W}_{o,r} \tag{4}$$

where $\mathbf{H}^t_r \in \mathbb{R}^{n \times d_{att}}$ denotes the output of the temporal graph attention layer on trading day t, and $\mathbf{W}_{o,r} \in \mathbb{R}^{h_{tga} T d_{enc} \times d_{att}}$ denotes the output projection matrix, h_{tga} denotes the number of heads, and each head of temporal graph attention layer $\mathrm{TgaHead}_i \in \mathbb{R}^{n \times T d_{enc}}$ is formulated as follows:

$$\mathrm{TgaHead}_i = \sum_{v^t \in \tilde{V}} \sigma\Big(\sum_{u^t \in \mathcal{N}_r(v^t)} \alpha^i_{u^t, v^t} \mathbf{h}_{u^t} \Big) \tag{5}$$

where σ denotes the activation function, $\mathbf{h}_{u^t} \in \mathbb{R}^{T d_{enc}}$ denotes the u^t-th row of historical price embedding $\mathbf{H}^t_{enc}$, and $\alpha^i_{u^t, v^t}$ denotes the importance of temporal edge (u^t, v^t) in i-th head, which is formulated as follows:

$$\alpha^i_{u^t, v^t} = \frac{\exp(\mathrm{LeakyReLU}(\mathbf{a}^T_{r,i}[\mathbf{h}_{u^t} || \mathbf{h}_{v^t}]))}{\sum_{k^t \in \mathcal{N}_r(v^t)} \exp(\mathrm{LeakyReLU}(\mathbf{a}^T_{r,i}[\mathbf{h}_{k^t} || \mathbf{h}_{v^t}]))} \tag{6}$$

where $\mathbf{a}_{r,i} \in \mathbb{R}^{2 T d_{enc}}$ denotes the weight vector of relation r and i-th head.

3.5 Heterogeneous Graph Attention Mechanism

As shown in Figure 1 (d), we already have messages from different types of neighbors through the two-stage attention mechanism. Then, we propose the heterogeneous graph attention network to learn from different relationships in relation graphs.

We define message sources as three types of embeddings, namely, messages from ourselves $\mathbf{H}^t_{self}$, positive neighbors $\mathbf{H}^t_{pos}$, and negative neighbors $\mathbf{H}^t_{neg}$, respectively. $\mathbf{H}^t_{self} \in \mathbb{R}^{n \times d_{att}}$ is derived from $\mathbf{H}^t_{enc}$ through a linear transformation with $\mathbf{H}^t_{self} = \mathbf{W}_{self}\mathbf{H}^t_{enc} + \mathbf{b}_{self}$, where $\mathbf{W}_{self} \in \mathbb{R}^{Td_{enc} \times d_{att}}$ and $\mathbf{b}_{self} \in \mathbb{R}^{d_{att}}$ denote the learnable parameters. $\mathbf{H}^t_{pos}$ and $\mathbf{H}^t_{neg}$ are derived from the graph attention mechanism in section 3.4. Taking three groups of node embeddings as input, we can adaptively generate the importance of different relationships through attention mechanism. The weights of three relationships $(\beta_{self}, \beta_{pos}, \beta_{neg})$ can be shown as follows:

$$(\beta_{self}, \beta_{pos}, \beta_{neg}) = \mathrm{HGA}(\mathbf{H}^t_{self}, \mathbf{H}^t_{pos}, \mathbf{H}^t_{neg}) \tag{7}$$

We first use three Multi-Layer Perceptrons (MLP) to transform these three embeddings. Then we measure the importance of each embedding using a heterogeneous attention vector $\mathbf{q}$. Furthermore, we average the importance of all node embeddings, which can be explained as the importance of each company relation. The importance of each company relation, denoted as $r \in \{self, pos, neg\}$, is shown as follows:

$$w_r = \frac{1}{|\tilde{V}|} \sum_{v^t \in \tilde{V}} \mathbf{q}^T \tanh(\mathbf{W}\mathbf{h}_{v^t,r} + \mathbf{b}) \tag{8}$$

where $\mathbf{W} \in \mathbb{R}^{d_{att} \times d_q}$ and $\mathbf{b} \in \mathbb{R}^{d_q}$ are the parameters of MLP, $\mathbf{q} \in \mathbb{R}^{d_q}$ denotes the attention vector, and $\mathbf{h}_{v^t,r}$ denotes the v_t-th row of $\mathbf{H}^t_r$. Note that all above parameters are shared for all relationship of node embeddings. After obtaining the importance of each relationship, we calculate the contribution of each relationship and obtain the final embedding $\mathbf{Z}^t \in \mathbb{R}^{n \times d_{att}}$ as follows:

$$\begin{aligned} \beta_r &= \frac{\exp(w_r)}{\sum_{r \in \{self,pos,neg\}} \exp(w_r)} \\ \mathbf{Z}^t &= \sum_{r \in \{self,pos,neg\}} \beta_r \cdot \mathbf{H}^t_r \end{aligned} \tag{9}$$

For a better understanding of the aggregating process of heterogeneous graph attention layer, we give a brief explanation in Figure 1 (d). Then we apply the final embedding to a semi-supervised node classification task.

3.6 Optimization Objectives

Here we give the implementation of objective functions. We model the stock movement prediction task as a semi-supervised node classification problem. Specifically, we selected 200 stocks of which future movements are ranked in top-100 or bottom-100, and labeled the corresponding nodes as 1 and 0, respectively. Then, we use one layer of MLP as the classifier to get the classification results of labeled nodes. Furthermore, we use binary cross-entropy to calculate the objective function L, which is formulated as follows:

$$\begin{aligned} \hat{\mathbf{Y}} &= \sigma(\mathbf{W}\mathbf{Z}^t_l + \mathbf{b}) \\ \mathcal{L} &= -\sum_{l \in \mathcal{Y}_t} [\mathbf{Y}^t_l \log(\hat{\mathbf{Y}}_l) + (1 - \mathbf{Y}^t_l) \log(1 - \hat{\mathbf{Y}}_l)] \end{aligned} \tag{10}$$

where $\mathcal{Y}_t$ denotes the set of labeled nodes, $\mathbf{Y}^t_l$ and $\mathbf{Z}^t_l$ denote the label and embedding of the labeled node l, respectively, σ denotes the Sigmoid activation function, and $\mathbf{W}$ and $\mathbf{b}$ are the parameters of MLP. With the guide of labeled data, we use Adam [23] optimizer to update the parameters of our proposed method. Please note that we use this objective function to jointly optimize the parameters of historical price encoder, temporal and heterogeneous graph neural network and node classifier.

4 EXPERIMENTS

In this section, we first introduce the datasets and experimental settings. Then we detail report the experimental results in real-world dataset and applications.

4.1 Experimental Setttings

Datasets. Extensive experiments are conducted in both the Unite States and Chinna's stock markets by choosing the constituted entities in S & P 500 and CSI 300 index. The historical price data from 2016 to 2021 are chosen as our datasets. In addition to historical price data, our input data also includes company relation graphs. The graphs are generated by the stock price correlation matrix, which is introduced in Section 3.2. The stock price correlation matrix of each day is determined by the historical price movement of past 20 trading days. Specifically, we compare the opening price, closing price, trading volume, and trading volume of each pair of two stocks, calculate the correlation coefficient between them and take the mean value as the element of the correlation matrix.

Parameter Settings. Our temporal graph $\tilde{G}$ contains company relationships for 20 trading days. The $d_{\mathcal{N}}$ and $t_{\mathcal{N}}$ of neighborhood function $\mathcal{N}(\cdot)$ are both set as 1. During the graph generation process, the threshold for generating one edge is set as 0.6. The historical price data of the previous 20 trading days are used as input features. The feature dimension d_{feat} of the encoding layer is 6, the input dimension d_{in} and encoding dimension d_{enc} of the encoding layer are both 128. The hidden dimension d_{hidden} is 512, the dimension of value d_v is 128, and the number of heads h_{enc} is 8. In temporal graph attention layer, d_{att} is 256, and the number of heads h_{tga} is 4. The dimension of the attention vector in the heterogeneous graph attention layer, d_q, is 256.

Trading Protocols. On the basis of [13], we use the daily buy-hold-sell trading strategy to evaluate the performance of stock movement prediction methods in terms of returns. During each trading day during the test period (from January 1, 2020 to December 31, 2020), we use simulated stock traders to predict transactions:

(1) When the trading day t closes, traders use this method to get the prediction score, that is, the ranking of the predicted rate of return of each stock.
(2) When the trading day $t + 1$ opens: the trader sells the stock bought on the trading day t. Meanwhile, traders buy stocks with high expected returns, i.e., the stocks with top-k scores.

Table 2: Performance evaluation of compared models for financial time series prediction in S&P 500 and CSI 300 datasets. ACC and ARR measure the prediction performance and portfolio return rate of each prediction model, respectively, where the higher is better. AV and MDD measure the investment risk of each prediction model where the lower absolute value is better. ASR, CR, and IR measure the profit under a unit of risk, where the higher is better.

	S&P 500							CSI 300						
	ACC	ARR	AV	MDD	ASR	CR	IR	ACC	ARR	AV	MDD	ASR	CR	IR
LSTM	0.532	0.377	0.449	-0.382	0.842	0.989	0.954	0.515	0.291	**0.318**	-0.240	0.915	1.213	0.877
GRU	0.530	0.362	**0.445**	-0.379	0.813	0.955	0.934	0.517	0.312	0.320	-0.243	0.975	1.284	0.932
Transformer	0.533	0.385	0.454	-0.384	0.848	1.005	0.960	0.518	0.327	0.322	-0.245	1.016	1.335	0.969
eLSTM	0.534	0.434	0.454	-0.373	0.955	1.163	1.041	0.520	0.330	0.323	-0.239	1.022	1.381	0.991
LSTM+GCN	0.538	0.470	0.442	-0.354	1.062	1.326	1.103	0.523	0.351	0.320	**-0.217**	1.097	1.618	1.119
LSTM+RGCN	0.565	0.558	0.463	-0.366	1.205	1.522	1.203	0.536	0.509	0.326	-0.235	1.561	2.166	1.537
TGC	0.552	0.528	0.455	**-0.344**	1.163	1.535	1.180	0.531	0.453	0.323	-0.224	1.402	2.022	1.412
MAN-SF	0.551	0.527	0.467	-0.357	1.130	1.478	1.157	0.527	0.418	0.334	-0.225	1.251	1.858	1.282
HATS	0.541	0.494	0.466	-0.387	1.060	1.277	1.110	0.525	0.385	0.332	-0.249	1.160	1.546	1.116
REST	0.549	0.502	0.466	-0.359	1.079	1.398	1.117	0.528	0.425	0.331	-0.228	1.284	1.864	1.298
AD-GAT	0.564	0.535	0.457	-0.371	1.170	1.444	1.187	0.539	0.537	0.329	-0.240	1.632	2.238	1.596
THGNN	**0.579**	**0.665**	0.468	-0.369	**1.421**	**1.804**	**1.340**	**0.551**	**0.632**	0.336	-0.237	**1.881**	**2.667**	**1.875**

(3) Please note that if a stock is continuously rated with the highest expected return, the trader holds the stock.

In calculating the cumulative return on investment, we follow several simple assumptions:

(1) Traders spend the same amount on each trading day (for example, \$50,000). We made this assumption to eliminate the time dependence of the testing process in order to make a fair comparison.
(2) There is always enough liquidity in the market to satisfy the opening price of the order on the $t + 1$ day, and the selling price is also the opening price on the $t + 1$ day.
(3) Transaction costs are ignored because the cost of trading US stocks through brokers is relatively cheap, whether on a transaction-by-transaction basis or on a stock-by-stock basis.Fidelity Investments (Fidelity Investments) and Interactive Brokerage (Interactive Broker), for example, charge \$4.95 and \$0.005 per transaction, respectively.

Compared Baselines. We compared our proposed method with state-of-the-art sequential-based models as well as the graph-based approaches. They are 1) Non-graph-based Methods, includes LSTM [17], GRU [11], Transformer [39], eLSTM [24]. 2) Graph-based Methods: LSTM-GCN [4], LSTM-RGCN [25], TGC [15], MAN-SF [35], HATS [22], REST [43] and AD-GAT [10].

Evaluating Metrics. Since the goal is to accurately select the stocks with highest returns appropriately, we use seven metrics: prediction accuracy (ACC), annual return rate (ARR), annual volatility (AV), maximum draw down (MDD), annual sharpe ratio (ASR), calmar ratio (CR), and information ratio (IR) to report the performance of baselines and our proposed model. Prediction accuracy is widely used to evaluate classification tasks, such as stock movement prediction [3, 24], so we calculate the prediction accuracy of all stocks for each trading day during the test period. Because ARR directly reflects the effect of stock investment strategy, ARR is our main measure, which is calculated by adding up the returns of selected stocks on each test day in a year. AV directly reflects the average risk of investment strategy per unit time. MDD reflects the maximal draw down of investment strategy during the whole test period. ASR reflects the benefits of taking on a unit of volatility with $ASR = \frac{ARR}{AV}$. CR reflects the benefits of taking on a unit of draw down with $CR = \frac{ARR}{\text{abs}(MDD)}$. IR reflects the excess return under additional risk. The smaller the absolute value of AV and MDD is, the higher the value of ACC, ARR, ASR, CR, and IR is, the better the performance is. For each method, we repeated the test process five times and reported average performance to eliminate fluctuations caused by different initialization.

4.2 Financial Prediction

In this section, we evaluate the performance of financial time series prediction and portfolio, which is the main task of this paper. Table 2 reports the performance through evaluation metrics such as ACC and ARR for each method in two datasets. The first four rows of Table 2 show the performance of models that do not use graph-based technology. It it clear that, none of these four methods is satisfactory, and their performance is all lower than that of other baselines. This proves that models without using company relationship data cannot achieve optimal performance.

Lines 5 to 11 of Table 2 show the performance of the baseline models using graph-based technology. According to line 6, LSTM+RGCN performs best. This proves the effectiveness of using heterogeneous graphs of inter-company relationships. Note that according to line 7, TGC's performance is also competitive and its investment strategy is less volatile. This proves the effectiveness of using the dynamic relationship between companies.

According to the previous observation, the financial prediction model can be improved by using the heterogeneous or dynamic relationships of the company relation graph. Therefore, it is necessary to design an innovative model to improve the prediction performance of financial series from both dynamic and heterogeneous graph structure. According to the last row of Table 2, our proposed THGNN outperforms all baselines and proves the superiority of temporal and heterogeneous graph neural network in financial time series prediction.

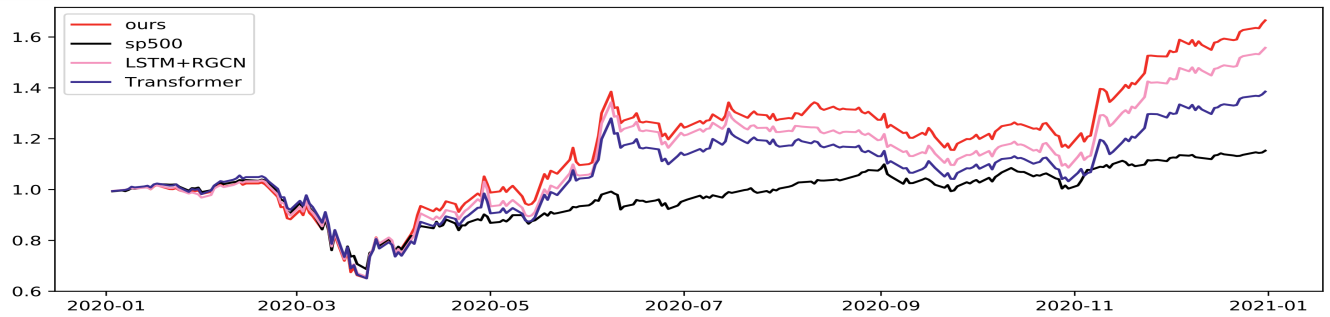


Figure 2: The accumulated returns gained in the test set (2020) by our proposed THGNN and selected baselines. For better illustration, we select one baseline from non-graph-based model and graph-based model, respectively.

Table 3: Performance evaluation of ablated models for financial time series prediction in S&P 500 dataset. ACC, ARR, ASR, CR, and IR measure the prediction performance and portfolio return rate of each prediction model, where the higher is better.

	ACC	ARR	ASR	CR	IR
THGNN-noenc	0.548	0.571	1.201	1.524	1.082
THGNN-notemp	0.539	0.486	0.964	1.292	0.946
THGNN-nohete	0.553	0.600	1.279	1.618	1.198
THGNN	**0.579**	**0.665**	**1.421**	**1.804**	**1.340**

4.3 Ablation Study

In this section, we conduct the ablation experiments, that is, evaluating the performance of our methods that one part is dropped.

According to the first row of Table 3, THGNN-noenc can not achieve the best performance after dropping the historical price encoding layer. This is because the encoder is responsible for extracting the time correlation in the historical price information. According to the second row, THGNN-notemp achieves unsatisfactory performance after dropping the temporal graph attention layer. This is because the temporal graph attention layer is responsible for dynamically adjusting the relationship between companies. Moreover, the relationship between companies changes dynamically over time, especially over a long period of time. According to the third row, THGNN-nohete cannot achieve the best performance after dropping the heterogeneous attention mechanism. This is because the heterogeneous graph attention layer is responsible for weighing the importance of messages from different relationships.

4.4 Performance of the Portfolio

In the performance evaluation of the portfolio strategy, we reported 6 widely-used evaluating metrics for portfolio, e.g., the annualized rate of return (ARR), annual sharp ratio (ASR), calmar ratio (CR), and information ratio (IR). Then, we show the accumulative return curve to compare the investment portfolio of our model and baselines during the test period. According to the trading protocols mentioned by section 4.1, we use the output of the prediction model to adjust our position day by day.

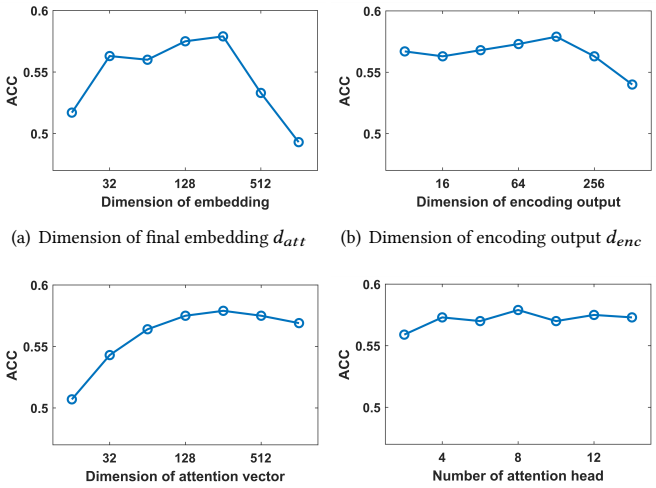
Table 2 reports the performance of our model's and baselines' portfolio strategies. It is clear that our method has achieved the best performance in four of the six investment evaluating metrics. Specifically, our method performs best in terms of ARR, ASR, CR and IR. TGC and LSTM+GCN perform better than our model in terms of AV and MDD. This shows that our proposed THGNN takes the initiative to take more risks to pursue higher risk-return ratio than other baselines'. According to Table 3, THGNN outperforms its sub-models in terms of portfolio strategy return evaluating metrics (e.g., ARR, ASR, CR, and IR). Therefore, our model shows strong effectiveness in building profitable portfolio strategies.

In order to further establish and evaluate the returns of our model during the test time, we calculate the cumulative returns for each trading day during the test period. We report the cumulative return curve of our model and other models on each trading day in Figure 2. Due to space constraints, we select some representative baselines to compare with our model. We can observe that all baselines outperform the S&P500 index. None of the models beat the others in the first four months. After starting in May, our model began to stay ahead of other models. In the following time, our model gradually widened the gap with other models. THGNN remained in the lead in the last month of 2020 and eventually achieved a profit on investment of more than 60 per cent. Experimental results on other baselines can draw similar conclusion.

4.5 Parameter Sensitivity

In this section, we report the experimental results of parameter sensitivity on the financial prediction task on S&P500 dataset with various parameters in Figure 3.

According to Figure 3 (a), we can observe that the performance of the model increases with the increase of embedded dimensions. The performance of the model peaked at 256 and then degraded rapidly. This is because embedded information needs a suitable dimension to reduce information loss and noise. According to Figure 3 (b), the performance of the model slowly improves as the coding output dimension increases. The performance of the model begins to deteriorate after the dimension reaches 128. This is because the input information dimension is low, so the low-dimensional output can also achieve better performance. According to Figure 3 (c), the performance of the model increases with the increase of the attention vector dimension. When the dimension reaches 256, the performance of the model reaches its peak. Continuing to increase the dimension will lead to overfitting, which will degrade the model. According to Figure 3 (d), we can observe that the fluctuation of the model performance is low. We choose to pay attention to the number of force heads as 8. We also note that increasing the number of attention heads will make the model training more stable.



(a) Dimension of final embedding d_{att} (b) Dimension of encoding output d_{enc}

(c) Dimension of the attention vector **q** (d) Number of attention head h

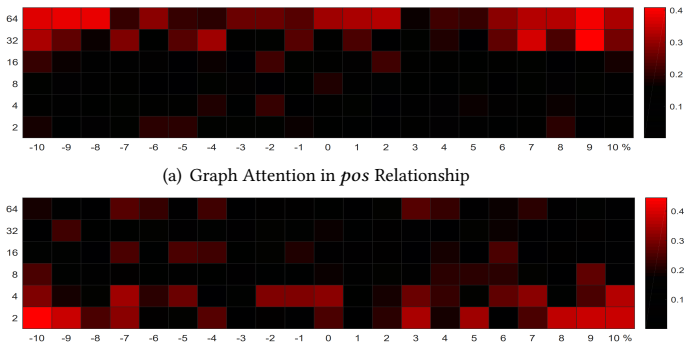
Figure 3: Prediction performance of THGNN in terms of dimension of final embedding d_{att}, dimension of encoding output d_{enc}, dimension of the attention vector q, and number of attention head h.

4.6 Interpretability of Graph Neural Network

The stock price fluctuation of listed companies is not only related to the historical price information of the company, but also affected by the companies related to it. Therefore, we need to integrate dynamic and heterogeneous relevant information as input to the prediction model. In our technology, the relationship between each company changes dynamically over time. The strength of the relationship between companies, that is, the proportion of contribution to the message also changes over time. Our time chart attention mechanism can dynamically adjust the importance of each company in the diagram. In addition, our heterogeneous attention mechanism can dynamically adjust the importance of each source. Therefore, our model can help to predict stock price volatility more accurately, and the experimental results verify the superiority of our model performance.

Then, in order to explore the interpretability of our proposed model, we extract the attention weight of the graph in the process of the model prediction. We counted the attention weight of all nodes in the process of message delivery on the relational graph. Under different daily returns and node degrees, we take the mean value of attention weights and visualize the statistical results, which are reported in Figure 4.

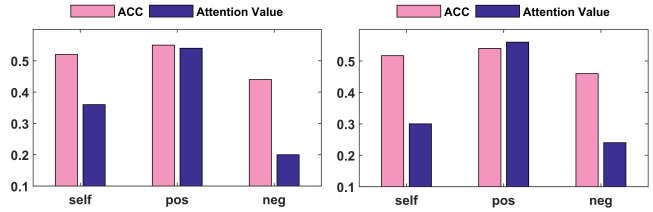
Figure 4 (a) shows the attention weights on the *pos* diagram. We can see (on the y-axis) that the nodes with higher degrees have higher average attention weights. This shows that in the process of message delivery on the *pos* graph, the degree is higher, that is, the nodes with more neighbors will contribute more messages to their neighbors. We also found that (on the x-axis), companies with larger fluctuations in daily returns also had higher average attention weights. This shows that more volatile price changes will contribute more information to their neighbors, which also means that price fluctuations will produce momentum spillover effects.



(a) Graph Attention in *pos* Relationship

(b) Graph Attention in *neg* Relationship

Figure 4: Visualization of attention weights, X-axis denotes the daily return of stocks. Y-axis denotes the average degree in each company relation graph.



(a) ACC values on S&P500. (b) ACC values on CSI300.

Figure 5: Prediction performance of single message source and corresponding attention value.

According to Figure 4 (b), we can see that on the *neg* graph, with a lower degree node, the average attention weight is higher. This indicates that during message delivery on the *neg* graph, nodes with lower degrees contribute more messages to their neighbors.

For more interpretable experimental results, we also visualize each relationship's attention weights and show the corresponding performance when using only one relationship. Specifically, we trained our proposed THGNN at two datasets, and counted the mean value of the attention weights in heterogeneous graph attention layer. Then, we used each single relationship's message as the input of the prediction model to obtain the prediction performance, as illustrated in Figure 5.

It is clear that *self* and *pos* message resources achieve better prediction performance than *neg*. Moreover, the *pos* message resource contribute importantly to the prediction model, which proves the contribution to the prediction model. The reason might be that the influence between companies in the similar price movement is relatively useful for prediction future price movement. Although the *neg* message resource has an unsatisfactory performance when predicting price movement single, it still contributes to our proposed THGNN in achieving the state-of-the-art performance according to Table 3. We can see that temporal graph attention layer can reveal the difference between nodes and weights then adequately, and the heterogeneous graph attention can adjust the contribution of each message resource adaptively. The result demonstrates the effectiveness of graph-structure information and the interpretability of proposed graph neural network model.

(a) List of selected stocks

(b) Historical price and buy/sell signals

(c) Related listed companies

Figure 6: The desktop interface of investment portfolio based on our proposed THGNN method. It includes price-relevant listed companies from historical data and visualization of how does our method makes predictions on buying and selling. The part (a) lists the stocks held by our method in order from highest to lowest. And part (b) shows the 'buy' and 'sell' signals generated by our trading protocols. Then part (c) lists the listed companies related to China National Petroleum Corporation (CBPC: 601857) and shows which ones have higher correlation to this company according to our generated stock correlation graph.

4.7 System Implementation

In this section, we introduce the implementation detail of our proposed methods. We first show the settings of model employment and training strategy. Then we show the web-based application, which shows how our proposed method gives customers informative advice. Our proposed THGNN is re-trained every trading day. To handle the large-scale dataset, we leverage mini-batch gradient descent optimization with a batch size of 256 and a learning rate of 3e-4. The model is implemented by PyTorch and deployed in Python and Java. Besides, we use distributed Scrapy [32] to obtain historical stock data and utilize Neo4j [18] as the graph database to store relational graphs.

Figure 6 shows the interface of our desktop application. The upper left part of Figure 6 is the list of stocks to be held based on the THGNN strategy. The lower left part of Figure 6 reports the price change curve of China National Petroleum Corporation (CBPC: 601857). It contains buy and sell points suggested by our THGNN. B denotes buying and S denotes selling. It can be seen that our model provides three buy signals and two sell signals. The lower right part of Figure 6 reports the relevant companies of CBPC. Companies with high stock price volatility correlations are marked with red arrowhead. The results shows that our investment strategy provides informative advice through a relational graph approach.

5 CONCLUSION AND DISCUSSION

In this paper, a new temporal and heterogeneous graph neural network model is proposed for financial time series prediction. Our method addresses the limitations of the existing works of graph neural networks by adjusting the message contribution ratio of each node through temporal and heterogeneous graph attention mechanism. We evaluate the effectiveness of the proposed method comprehensively by comparing it with the most influential graph-based and non-graph-based baselines. In addition, THGNN performs better than other baselines in the actual investment strategy, and the results show that our approach based on dynamic heterogeneous graph can obtain a more profitable portfolio strategy than those based on static or homogeneous graph structure.

In conclusion, this paper is the first time to model the inter-company relationship into a heterogeneous dynamic graph, and apply it to the financial time series prediction problem. This is beneficial to the more extensive research and innovation of graph-based technology in the financial field. On the one hand, we model the company relation graph as the real dynamic heterogeneous graph; on the other hand, we improve the financial time series prediction model through the latest graph neural network technology. Besides, there is still room for improvement in our work on generating real-life company relation graphs. In the future, we will focus on improving the modeling of the company's relation to help the prediction model obtain more accurate training input graph data.

ACKNOWLEDGMENTS

Dawei Cheng is supported by the NSFC 62102287, Ying Zhang is supported by ARC DP210101393.

REFERENCES

[1] Klaus Adam, J. Nicolini, and A. Marcet. 2008. Stock Market Volatility and Learning. *IO: Theory* (2008).

[2] Michel Ballings, D. V. Poel, Nathalie Hespeels, and Ruben Gryp. 2015. Evaluating multiple classifiers for stock price direction prediction. *Expert Syst. Appl.* 42 (2015), 7046–7056.

[3] Chi Chen, Li Zhao, Jiang Bian, Chunxiao Xing, and Tie-Yan Liu. 2019. Investment Behaviors Can Tell What Inside: Exploring Stock Intrinsic Properties for Stock Trend Prediction. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2019).

[4] Yingmei Chen, Zhongyu Wei, and Xuanjing Huang. 2018. Incorporating Corporation Relationship via Graph Convolutional Neural Networks for Stock Price Prediction. *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (2018).

[5] Dawei Cheng, Zhibin Niu, and Liqing Zhang. 2020. Delinquent events prediction in temporal networked-guarantee loans. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[6] Dawei Cheng, Xiaoyang Wang, Ying Zhang, and Liqing Zhang. 2020. Graph Neural Network for Fraud Detection via Spatial-temporal Attention. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[7] Dawei Cheng, Fangzhou Yang, Xiaoyang Wang, Y. Zhang, and Liqing Zhang. 2020. Knowledge Graph-based Event Embedding Framework for Financial Quantitative Investments. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020).

[8] Dawei Cheng, Fangzhou Yang, Sheng Xiang, and Jin Liu. 2022. Financial time series forecasting with multi-modality graph neural network. *Pattern Recognition* 121 (2022), 108218. https://doi.org/10.1016/j.patcog.2021.108218

[9] Dawei Cheng, Yiyi Zhang, Fangzhou Yang, Yi Tu, Zhibin Niu, and Liqing Zhang. 2019. A dynamic default prediction framework for networked-guarantee loans. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2547–2555.

[10] Rui Cheng and Qing Li. 2021. Modeling the Momentum Spillover Effect for Stock Prediction via Attribute-Driven Graph Attention Networks. In *AAAI*.

[11] Kyunghyun Cho, B. V. Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *ArXiv* abs/1406.1078 (2014).

[12] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. [n.d.]. Using Structured Events to Predict Stock Price Movement: An Empirical Investigation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014, October 25-29, 2014*. 1415–1425.

[13] M. Dixon, D. Klabjan, and J. Bang. 2017. Classification-based financial markets prediction using deep neural networks. *Algorithmic Finance* 6 (2017), 67–77.

[14] Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. 2019. Enhancing Stock Movement Prediction with Adversarial Training. In *IJCAI*.

[15] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal Relational Ranking for Stock Prediction. *ACM Transactions on Information Systems (TOIS)* 37 (2019), 1 – 30.

[16] Gartheeban Ganeshapillai, John Guttag, and Andrew Lo. 2013. Learning connections in financial time series. In *International Conference on Machine Learning*. PMLR, 109–117.

[17] S. Hochreiter and J. Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9 (1997), 1735–1780.

[18] Florian Holzschuher and René Peinl. 2013. Performance of graph query languages: comparison of cypher, gremlin and native access in Neo4j. In *EDBT '13*.

[19] Joel F Houston, Chen Lin, and Zhongyan Zhu. 2016. The financial implications of supply chain changes. *Management Science* 62, 9 (2016), 2520–2542.

[20] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[21] Zhigang Jin, Ya-Chi Yang, and Yuhong Liu. 2019. Stock closing price prediction based on sentiment analysis and LSTM. *Neural Computing and Applications* 32 (2019), 9713–9729.

[22] Raehyun Kim, Chan Ho So, Minbyul Jeong, Sanghoon Lee, Jinkyu Kim, and Jaewoo Kang. 2019. HATS: A Hierarchical Graph Attention Network for Stock Movement Prediction. *ArXiv* abs/1908.07999 (2019).

[23] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2015).

[24] Qing Li, Jinghua Tan, Jun Wang, and HsinChun Chen. 2020. A Multimodal Event-driven LSTM Model for Stock Prediction Using Online News. *IEEE Transactions on Knowledge and Data Engineering* (2020), 1–1. https://doi.org/10.1109/TKDE.2020.2968894

[25] Wei Li, Ruihan Bao, Keiko Harimoto, Deli Chen, Jingjing Xu, and Qi Su. 2020. Modeling the Stock Relation with Graph Network for Overnight Stock Movement Prediction. In *IJCAI*.

[26] Zhige Li, Derek Yang, Li Zhao, Jiang Bian, Tao Qin, and Tie-Yan Liu. 2019. Individualized Indicator for All: Stock-wise Technical Indicator Optimization with Stock Embedding. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2019).

[27] Xin Liang, Dawei Cheng, Fangzhou Yang, Yifeng Luo, Weining Qian, and Aoying Zhou. 2020. F-HMTC: Detecting Financial Events for Investment Decisions Based on Neural Hierarchical Multi-Label Text Classification.. In *IJCAI*. 4490–4496.

[28] Qikai Liu, Xiang Cheng, Sen Su, and Shuguang Zhu. [n.d.]. Hierarchical Complementary Attention Network for Predicting Stock Price Movements with News. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*. 1603–1606.

[29] Weiwen Liu, Yin Zhang, Jianling Wang, Yun He, James Caverlee, Patrick PK Chan, Daniel S Yeung, and Pheng-Ann Heng. 2021. Item Relationship Graph Neural Networks for E-Commerce. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[30] Xiao-Yang Liu, Jingyang Rui, Jiechao Gao, Liuqing Yang, Hongyang Yang, Zhaoran Wang, Christina Dan Wang, and Guo Jian. 2021. FinRL-Meta: Data-Driven Deep ReinforcementLearning in Quantitative Finance. *Data-Centric AI Workshop, NeurIPS* (2021).

[31] Simone Merello, Andrea Picasso Ratto, L. Oneto, and E. Cambria. 2019. Ensemble Application of Transfer Learning and Sample Weighting for Stock Market Prediction. *2019 International Joint Conference on Neural Networks (IJCNN)* (2019), 1–8.

[32] Daniel N. Myers and James W. McGuffee. 2015. Choosing Scrapy. *Journal of Computing Sciences in Colleges*.

[33] World Federation of Exchanges database. 2021. Market capitalization of listed domestic companies. https://data.worldbank.org/indicator/CM.MKT.LCAP.CD accessed 25-July-2021.

[34] Ashwin Paranjape, Austin R. Benson, and Jure Leskovec. 2017. Motifs in Temporal Networks. *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (2017).

[35] Ramit Sawhney, S. Agarwal, Arnav Wadhwa, and R. Shah. 2020. Deep Attentive Learning for Stock Movement Prediction From Social Media Text and Company Correlations. In *EMNLP*.

[36] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Shah. 2020. Deep Attentive Learning for Stock Movement Prediction from Social Media Text and Company Correlations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 8415–8426.

[37] Y. Tu, Li Niu, Junjie Chen, Dawei Cheng, and Liqing Zhang. 2020. Learning From Web Data With Self-Organizing Memory Module. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 12843–12852.

[38] Yi Tu, Li Niu, Weijie Zhao, Dawei Cheng, and Liqing Zhang. 2020. Image cropping with composition and saliency aware aesthetic score map. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 12104–12111.

[39] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *ArXiv* abs/1706.03762 (2017).

[40] Yaowei Wang, Qing Li, Zhexue Huang, and Mark Junjie Li. [n.d.]. EAN: Event Attention Network for Stock Price Trend Prediction based on Sentimental Embedding. In *Proceedings of the 11th ACM Conference on Web Science, WebSci 2019, Boston, MA, USA, June 30 - July 03, 2019*. 311–320.

[41] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

[42] Sheng Xiang, Dong Wen, Dawei Cheng, Ying Zhang, Lu Qin, Zhengping Qian, and Xuemin Lin. 2021. General graph generators: experiments, analyses, and improvements. *The VLDB Journal* (2021).

[43] Wentao Xu, Weiqing Liu, Chang Xu, Jiang Bian, Jian Yin, and Tie-Yan Liu. 2021. REST: Relational Event-driven Stock Trend Forecasting. *Proceedings of the Web Conference 2021* (2021).

[44] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do Transformers Really Perform Bad for Graph Representation?. In *NeurIPS*.

[45] Dawei Zhou, Lecheng Zheng, Jiawei Han, and Jingrui He. 2020. A Data-Driven Graph Generative Model for Temporal Interaction Networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2020).