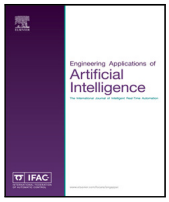
Engineering Applications of Artificial Intelligence 126 (2023) 106849

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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



A representation-learning-based approach to predict stock price trend via dynamic spatiotemporal feature embedding☆

Bowen Pang [a,b], Wei Wei [a,b,c,d,*], Xing Li [a,b], Xiangnan Feng [e], Chao Li [f,g]

[a] School of Mathematical Sciences, Beihang University, Beijing, 100191, China
[b] Key Laboratory of Mathematics Informatics Behavioral Semantics, Ministry of Education, Beijing, 100191, China
[c] Institute of Artificial Intelligence, Beihang University, Beijing, 100191, China
[d] Zhongguancun Laboratory, Beijing, 100094, China
[e] Center for Humans and Machines, Max Planck Institute for Human Development, Berlin, 14195, Germany
[f] Department of Mathematics and Computer Science, Hengshui University, Hengshui, 053000, China
[g] School of Information Science and Engineering, Yanshan University, Qinhuangdao, 066004, China

ARTICLE INFO

Keywords:
Graph neural network
Graph learning
Representation learning
Stock price trend prediction
Time series

ABSTRACT

Stock price trend prediction is a fascinating but difficult research topic. Recently, GNN-based models have been continuously proposed, which are believed to be more effective since they consider the information about stocks themselves and the information between stocks. However, the graph data are often static, unstructured and not all-inclusive, which cannot dynamically reflect all relationships between stocks. Therefore, we propose a novel model ***PriceExploration-Network*** (PE-Net), effectively utilizing both temporal and cross-sectional information contained in price to predict the price trend. PE-Net only requires the price data effectively saving the trouble of fetching alternative data and is able to capture the dynamic implicit relations between stocks by combining clustering techniques and GAT architecture. The effectiveness of PE-Net is examined on real-world S&P 500 constituents and the results demonstrate that PE-Net can outperform state-of-the-art models w.r.t. both accuracy and AUC.

1. Introduction

Stock market is an important part of human economic activities. There are plenty of methods created to describe the price movements or select the most profitable stocks, and all of these are related to one topic *price prediction*, which is undoubtedly full of difficulties and challenges. Traditional financial theories represented by EMH (Efficient Market Hypothesis) (Fama, 1965) and random walk theory (Osborne, 1959) state that the stock price is unpredictable and investors cannot acquire excess return by mining historical information. However, more and more theories and practices make challenge on this view (Jegadeesh and Titman, 1993; Lo and MacKinlay, 2011; Haugen and Lakonishok, 1987), which also motivates researchers to explore more kinds of price prediction methods.

In previous studies, dominant methods consider this problem from the perspective of time series. They treat the price sequences as time series and use different types of models to discover the features hidden in the temporal structure. For instance, statistical time series analysis models are the most classic and widely used, represented by ARIMA (Autoregressive Integrated Moving Average model) (Ariyo et al., 2014), Markov Switching (Hamilton, 1989) etc. In addition, statistical machine learning techniques are also introduced into this field. They feed various indicators of individual stocks at each time point into models like SVM (Support Vector Machine) (Lee, 2009) and Decision Tree (Khaidem et al., 2016) to forecast the future price trend. Recently, with the booming of deep learning, neural network methods have gained growing attention from researchers. Typically, RNN (Recurrent Neural Network) (Rumelhart et al., 1986) and its variants such as LSTM (Long Short Term Memory) (Hochreiter and Schmidhuber, 1997) and GRU (Gated Recurrent Unit) (Cho et al., 2014) can be well used to deal with the price prediction problem due to its design for processing sequential data.

Despite the plethora of existing methods based on temporal information, the lack of leveraging the cross-sectional information is a common defect. However, according to some literatures in finance (Moskowitz

☆ This work was supported by the National Natural Science Foundation of China (Grant Nos. 62276013, 62141605, 62050132), the Beijing Natural Science Foundation, China (Grant No. 1192012) and the Fundamental Research Funds for the Central Universities, China.

* Corresponding author.
E-mail addresses: pangbw@buaa.edu.cn (B. Pang), weiw@buaa.edu.cn (W. Wei), sy1809120@buaa.edu.cn (X. Li), xfeng@mpib-berlin.mpg.de (X. Feng), lichao@hsnc.edu.cn (C. Li).

https://doi.org/10.1016/j.engappai.2023.106849
Received 24 October 2022; Received in revised form 16 January 2023; Accepted 20 July 2023
Available online 9 August 2023

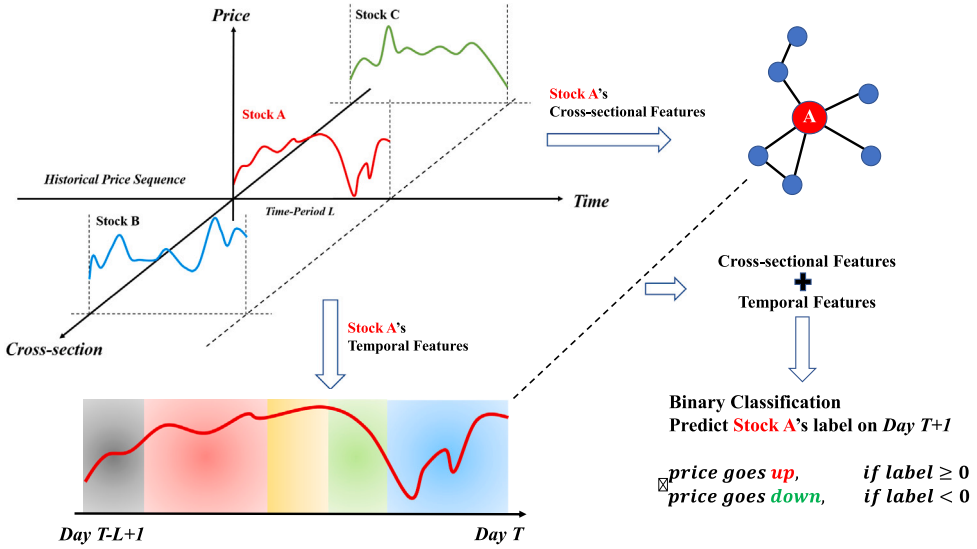


Fig. 1. The schematic diagram of the **conception** of our work.

and Grinblatt, 1999; Cohen and Frazzini, 2008), *Momentum Spillover* effect is widespread among different companies, which indicates that the return of a listed company's stock can be influenced by other companies similar or linked to it. The interrelation between companies is often presented in a graph structure, but the non-Euclidean property of this type of data makes it hard for traditional models to process. Fortunately, with the emergence of GNN (Graph Neural Network) (Zhou et al., 2020) models such as GCN (Graph Convolutional Network) (Kipf and Welling, 2016) and GAT (Graph Attention Network) (Veličković et al., 2017), researchers start to utilize these powerful tools to exploit the cross-sectional features of stocks for better forecasting effect (Chen et al., 2018; Feng et al., 2019; Hsu et al., 2021; Chen et al., 2021). Nevertheless, the graphs they are based on are mainly existing or explicitly-defined graphs such as sector-industry network or supply chain network. These graphs are often static and the dynamic interactions between stocks cannot be fully exploited from them. Also, the data for these graphs is even uneasy to obtain and process, which makes it more difficult to apply and extend these methods.

In this paper, we transform the price prediction problem into a binary classification problem, i.e., each stock is classified into a positive or negative category to represent the price rise or fall respectively. As it is shown in Fig. 1, our conception is to combine both the temporal and cross-sectional features of stocks but by solely mining stocks' historical price data. Specifically, we propose a neural-network-based model ***PriceExploration-Network*** (PE-Net). PE-Net can be divided into four modules, *preprocessing module*, *temporal embedding module*, *cross-sectional embedding module* and *prediction module* respectively. In *preprocessing module*, we extract the time-series motifs contained in price sequence and reconstruct the sequence to eliminate noise. Then, in *temporal embedding module*, we use GRU (Cho et al., 2014) to embed each stock's adjusted price sequence into a representation space to acquire the temporal embedding. After that, in *cross-sectional embedding module*, we put the computed temporal embedding vector of each stock into a GNN framework to update the embedding based on the correlation of stocks on cross-section. Noteworthily, in this module, we firstly use time series clustering techniques to construct a stock graph structure simply based on adjusted price sequences, instead of using a priori relationship which is popular in mainstream methods. Then, we apply GAT (Veličković et al., 2017) to process this graph at both intra- and inter-cluster level and update stocks' embedding vectors. Finally, in our *prediction module*, we concatenate all the embedding vectors for each stock to obtain the final representation and use a fully-connected layer to map it into a two-dimensional vector, which contains the probabilities of upward and downward movements. To sum up, the contribution and significance of our work is three-fold:

- A novel neural-network-based model PE-Net is proposed, which provides a complete framework to dynamically represent stocks by exploring the temporal and cross-sectional features contained in price sequences and predict the stocks' future price trend accordingly.
- At temporal level, PE-Net reconstructs price sequences by summarizing the temporal motifs to capture the price evolving patterns, which greatly contributes to the extraction of temporal features. At cross-sectional level, PE-Net possesses a hierarchical design for dynamic relationship modeling and feature integration by combining clustering technique and GAT, which effectively exploits the latent relationship between stocks and benefits the information aggregation on cross-section.
- PE-Net is tested on S&P 500 constituents and compared with other models. The experimental results illustrate that PE-Net can outperform the state-of-the-art models w.r.t. both accuracy and AUC (Area Under Curve). Also, the ablation study and module test demonstrate the effectiveness of the specific module design, and the trading simulation discusses the application of PE-Net to real-world trading.

The rest of the paper is organized as follows. Section 2 summarizes the related work on price prediction using neural network methods. Section 3 introduces our model PE-Net, its components and usage in detail. Section 4 describes the experiments and analyzes the results. Finally, we conclude our work in Section 5.

2. Related work

Using neural network methods to make stock price prediction has become a focused research area in recent years. Surveys Jiang (2021) and Kumbure et al. (2022) provide a comprehensive review of the application of machine learning techniques, especially neural network methods, in stock market prediction. In 1990s, Matsuba first introduced ANN (Artificial Neural Network) into this field (Matsuba, 1991). He

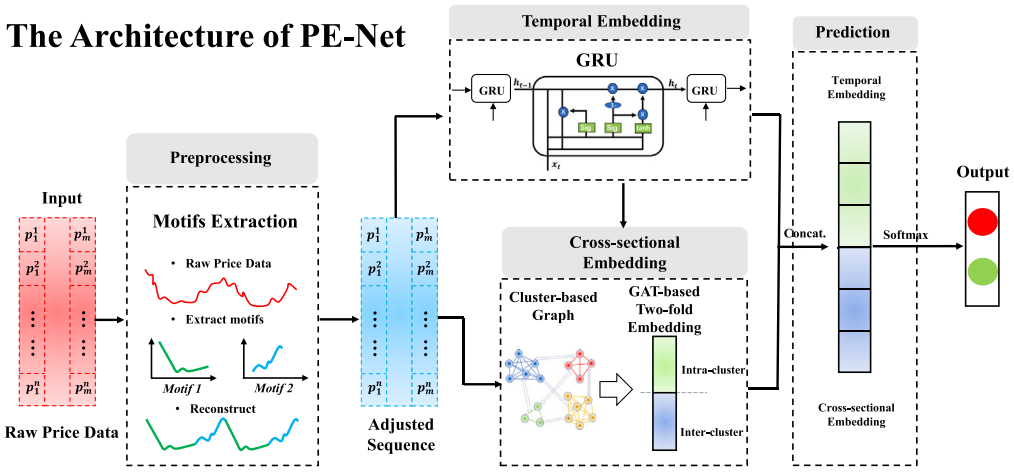


Fig. 2. The schematic diagram of **PE-Net**. As is shown, the entire workflow of PE-Net is implemented by four modules: ***preprocessing module***, ***temporal embedding module***, ***cross-sectional embedding module*** and ***prediction module***.

believed that the process of stock price movement was a complex non-linear function, and ANNs could be used to simulate this process to forecast price trend. Due to the sequence property of stock price series, RNN (Rumelhart et al., 1986) and its variants are the most widely used models currently. The basic idea behind these methods is to feed stock feature time series into a RNN-based model and output a representation vector for downstream task. The stock features used can come from many sources, e.g. raw price data, technical indicators, fundamental factors, sentiment, news etc. (Yao et al., 2018; Li et al., 2017, 2020b). Additionally, there are also some works expanding the classic RNN models and proposing new model architectures to better suit the prediction task. For instance, Cheng et al. (2018) proposed an attention-based LSTM model to predict price movement and design trading strategies. Zhang et al. (2017) proposed a novel recurrent architecture State-Frequency Memory (SFM) to capture the uncovered patterns of price time series in both time and frequency domains. Wu et al. (2021) combined LSTM with CNN (Convolutional Neural Network) to extract useful features from leading indicators like futures and options in order to gain better predictive effectiveness.

Thanks to the rise of GNN (Zhou et al., 2020) models in the past five years, more and more researches focus on using GNN models to exploit information contained in inter-stock relation data, which is often presented in graph structure. For instance, Chen et al. (2018) constructed a graph for a pool of Chinese stocks based on their investment facts and applied graph convolutional network model to process it. Feng et al. (2019) proposed a temporal graph convolution model to deal with a heterogeneous relationship graph, which contained two kinds of edges representing sector-industry relation and wiki-company-based relation respectively. They aimed at recommending the highest yielding stocks and tested their model on NYSE and NASDAQ stock pools. Chen et al. (2021) improved the original GCN model and combined it with Dual-CNN (Li et al., 2020a) to capture the features of both individual stock and the stock market. In addition to GCN (Kipf and Welling, 2016) and its variants, GAT (Veličković et al., 2017) has also been considered by the researchers. Hsu et al. (2021) argued that not all of the relations between stocks were accessible so the pre-defined graphs could not fully reflect stocks' interactions. Therefore, they designed a FinGAT model to extract the hidden relationship between stocks using GAT framework. As a summary, Wang et al. (2021) categorized different types of financial graphs and the relevant GNN-based methodology applied in recent years. In general, these GNN-based models rely on a graph structure to describe the interaction or similarity between stocks and apply GNN methods to extract structural features, which will be used later to update the node representation usually coming from the output of RNN-based models.

3. Model architecture

In this section, we introduce the architecture of PE-Net in detail. The whole framework is presented in Fig. 2. As it shows, PE-Net consists of four modules, which are ***preprocessing module***, ***temporal embedding module***, ***cross-sectional embedding module*** and ***prediction module*** respectively. In general, the whole workflow goes like this:

- At each time point, PE-Net preprocesses each stock's price sequence over a period of time in *preprocessing module* to capture its historical trend and outputs the adjusted sequences for downstream task.
- PE-Net feeds the adjusted sequences into *temporal embedding module* and outputs the temporal representation for each stock in stock pool. This representation contains the temporal information of each stock and is also used as initialization for subsequent operations.
- In *cross-sectional embedding module*, PE-Net firstly clusters all the stocks and constructs a graph based on the clustering result. The graph contains two kinds of edges representing intra- and inter-cluster relations respectively. Then, PE-Net extracts cross-sectional information from the graph structure and accordingly updates each stock's representation vector, which is the output of this module.
- In *prediction module*, PE-Net concatenates each stock's temporal and cross-sectional embedding vector to gain its final representation, which is then fed into the Softmax-MLP layer to output the forecasted price up and down probabilities.

With this design, PE-Net is able to mitigate the effects of noise contained in raw price sequences, represent each stock by comprehensively considering its temporal and cross-sectional characteristics with no need for predefining the graph structure and predict the future price trend based on the representation result.

3.1. Preprocessing module

Stock price data is commonly known to have a low signal-noise ratio, which makes it relatively difficult to recognize useful patterns directly from the raw data. Therefore, using some de-noising methods to preprocess the price data will be helpful for downstream task. Enlightened by technical analysis in practice, we hope to summarize the frequent patterns, or more exactly *temporal motifs*, contained in each stock's price sequence and reconstruct the sequences using extracted

motifs as it is recommended in Wen et al. (2019). A brief schematic diagram of this module can be seen in the above Fig. 2.

Specifically, the motif-extraction algorithm we use is named *BeatLex* proposed by Hooi et al. (2017). The basic idea of this algorithm is segmenting the original price sequence into subsequences and assigning a certain motif for each subsequence. Given subsequence $P_{a:b}$, we calculate the MDTW (Modified Dynamic Time Warping) distance (see Hooi et al. (2017)) between it and every motif in current motif set. If the smallest MDTW distance does not exceed the threshold, the corresponding motif is assigned to this subsequence and updated with the information of this subsequence to ensure that the motif can dynamically reflect the information of newly-added subsequence. Otherwise, if the smallest MDTW distance exceeds the threshold, a new motif is created based on this subsequence and is appended to the motif set. In *BeatLex* algorithm, the length of the subsequence ranges from s_{min} to s_{max}. However, in our model, we limit the length to a certain value in order to obtain the motifs of the same length. This operation is necessary for the following sequence reconstruction and clustering operation.

After summarizing the motifs, we reconstruct each price sequence based on its own motif set. Specifically, the motifs of all subsequences of this price sequence is concatenated in temporal order to generate the final adjusted motif-based sequence.

3.2. Temporal embedding module

After acquiring the reconstructed motif sequence of each stock, we then set out to exploit the temporal characteristics contained in these adjusted sequences. We adopt GRU (Gated Recurrent Unit) (Cho et al., 2014) to help finish this task. GRU is a widely used variant of RNN models. Compared with another popular model LSTM (Hochreiter and Schmidhuber, 1997), GRU has a simpler internal architecture and fewer parameters, so it is easier to train and to some degree reduces the risk of overfitting.

On each time point, the adjusted sequences of all the stocks on cross-section are fed into the GRU model. For each stock's sequence, the calculation inside the GRU is shown in Eqs. (1)–(4):

$$r_t = \sigma(W_r \cdot [h_{t-1}, s_t]), \tag{1}$$

$$z_t = \sigma(W_z \cdot [h_{t-1}, s_t]), \tag{2}$$

$$\tilde{h}_t = tanh(W_{\tilde{h}} \cdot [r_t * h_{t-1}, s_t]), \tag{3}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t, \tag{4}$$

where s_t denotes the element of input sequence at time step t, h_{t-1} denotes the output of the hidden layer (a.k.a. hidden state) at previous time step $t-1$, r_t, z_t, $\tilde{h}_t$ are the intermediate variables, h_t denotes the final output of hidden layer at time step t and σ, $tanh$ are two non-linear activation functions defined in Eqs. (5) and (6):

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{5}$$

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}. \tag{6}$$

As we can see, each element s_t in the adjusted sequence goes through three phases of computation inside the GRU. Firstly, as the input of time step t, s_t is sent to compare with the output of previous time step h_{t-1} to decide how much historical information needs to be kept and meanwhile s_t together with h_{t-1} is also used to determine how much historical information needs to be updated. These two computations are both finished with the combination of a linear transformation and a sigmoid activation. Secondly, an intermediate variable $\tilde{h}_t$ is computed based on time step t's input s_t, the proportion of s_t needed to be kept and previous time step's output h_{t-1}. The $\tilde{h}_t$ can be viewed as the preliminary output of this time step. Finally, $\tilde{h}_t$ and h_{t-1} are fused together proportionally to generate the output of this time step.

To improve the ability of this module, we employ a two-layer GRU in our design, in which the second layer's input is the hidden state of the first layer. The final output of this module is the last layer's last hidden state for each stock on cross-section.

3.3. Cross-sectional embedding module

This module is designed to capture the correlation between stocks and update the representation for each stock. Firstly, PE-Net clusters all the stocks in the stock pool based on their respective adjusted sequences, which are the output of *preprocessing module*. Then, PE-Net derives a graph structure from the clustering result. Finally, PE-Net uses GAT to analyze the graph and outputs the updated representation for each stock.

3.3.1. Graph construction

To start with, we apply SOM (self-organizing maps) (Kohonen, 1990) to cluster all the stocks on cross-section. For any input sample, i.e. the adjusted sequence of any stock in our case, the goal is to find one neuron best matching the input sample among all the neurons in the competition layer. The *matching* is measured by Euclidean distance between the input sample and weight vector. For instance, stock i's adjusted sequence is $(x_i^1, x_i^2, \dots, x_i^k)$ and the weight vector between it and any neuron j is $(w_{ij}^1, w_{ij}^2, \dots, w_{ij}^k)$, so the distance can be written as $D(s_i, n_j) = \sum_{n=1}^{k}(x_i^n - w_{ij}^n)^2$. As such, we calculate the distance between stock i and each neuron to find the neuron having the minimal distance to stock i and set this neuron as the best-matching neuron. Stock i is then assigned to this neuron and the corresponding weight vector is updated accordingly.

The update rule is based on the idea that the weights of neurons inside the best-matching neuron's neighborhood are updated accordingly, while the weights of those outside remain the same. In our model, we use Gaussian function to set the neighborhood and the whole update process is shown in Eqs. (7) and (8):

$$h_{i,c} = exp(-\|c - i\|^2 / 2(\sigma)^2), \tag{7}$$

$$w^i(t+1) = w^i(t) + \alpha(t) \cdot h_{i,c} \cdot (x(t) - w^i(t)), \tag{8}$$

where t denotes the current iteration step, $h_{i,c}$ denotes the distance between neuron i and best-matching neuron c, $x(t)$ denotes the current input sample and $\alpha(t)$ denotes the learning rate, which decays over time. The training process of SOM is done through iteration and a rule of thumb for this, as is mentioned in Kohonen (1990), is at least 500 times the number of competition neurons. Also, the number of competition neurons is recommended to set as $5 * \sqrt{N}$, where N denotes the number of samples (Tian et al., 2014). Noteworthily, with SOM to do clustering, the total number of clusters does not need to be set in advance. The adaptivity of SOM allows it to determine the cluster number by itself and automatically adjust the number according to different input samples.

After clustering, we next move on to construct a graph based on it. Specially, two types of relations are defined, which are intra-cluster and inter-cluster relations respectively. Within each cluster, there is a connecting edge between each two stock nodes, which represents the intra-cluster relation between them. Similarly, this process can also be repeated at cluster level. we can also repeat this process at cluster level. Every cluster is connected with each other and the edge indicates the existing inter-cluster relation. Remarkably, the whole graph is not fully-connected because these two types of edges cannot be treated equally. A schematic diagram of the final cluster-based graph is as shown in Fig. 3(A).

3.3.2. Cross-sectional embedding

In the previous section, a heterogeneous graph is defined with two types of edges. However, what relations these edges represent and

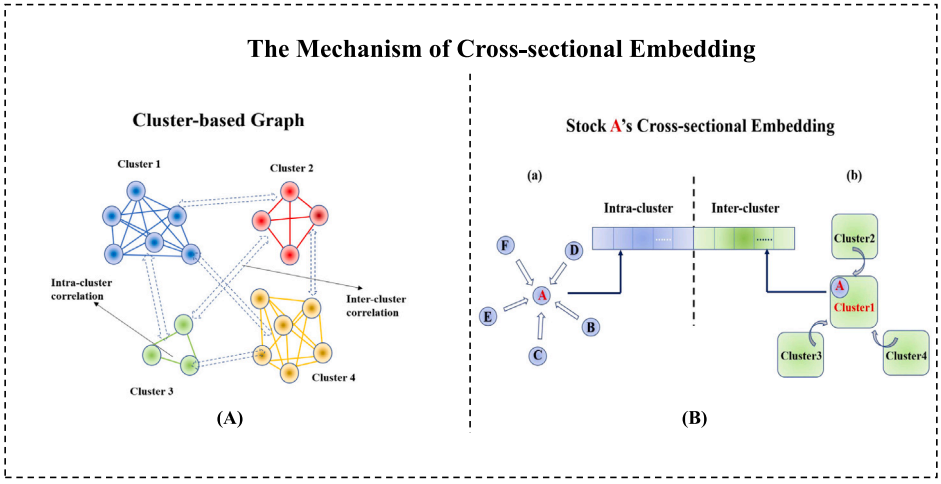


Fig. 3. The schematic explanation of the **cross-sectional embedding module**: subgraph (A) presents the structure of the derived cluster-based graph; subgraph (B) demonstrates the two-fold embedding happening on cross-section.

how we quantify them are still not clear. Also, the existing explicit relationship data is mostly static, which cannot reflect the dynamics of correlations in real world. Therefore, we choose not to define any explicit relations but apply GAT (Veličković et al., 2017) method to dynamically learn the latent intra- and inter-cluster correlations. Specifically, the use of GAT in this module is two-fold.

A. Intra-cluster embedding

Firstly, GAT is applied at intra-cluster level, which is shown in Fig. 3(B)(a). For center node stock i, its representation vector is updated according to Eq. (9):

$$Embed^{IntraC}(s_i) = \sigma(\sum_{s_j \in N(s_i)} \alpha_{ij} \cdot W \cdot Embed^{ts}(s_j)), \tag{9}$$

where $N(s_i)$ is the generalized neighborhood of stock i (including all its neighbors and itself), $Embed^{ts}(s_j)$ is the temporal embedding of stock j, σ is the non-linear activation function and α_{ij} is the attention-based weight vector whose computation is presented in Eqs. (10) and (11):

$$\alpha_{ij} = softmax(e_{ij}), \tag{10}$$

$$e_{ij} = LeakyReLU(a^T \cdot [W \cdot Embed^{ts}(s_i) \parallel W \cdot Embed^{ts}(s_j)]), \tag{11}$$

where $\parallel$ denotes the concatenation operation.

Noteworthily, e_{ij} is a measure of the similarity of node i and j, and researchers can also select other types of measure such as dot product or even a neural network layer. In sum, after this step, the embedding vector of each stock is updated by effectively aggregating the information of its one-order neighbor.

B. Inter-cluster embedding

Secondly, GAT is applied at inter-cluster level, which is exhibited in Fig. 3(B)(b). The key step here is the vector representation for each cluster. For a given cluster, we perform an element-wise maximization operation on the embedding vectors of all the stocks in this cluster and use the computed vector as the representation of this cluster. Notably, element-wise mean operation is also viable, so are other methods which can in some way reflect the node information in it. After representing each cluster as vector, GAT is then used to update each cluster's embedding vector according to Eq. (12):

$$Embed^{InterC}(c_i) = \sigma(\sum_{c_j \in N(c_i)} \alpha_{ij} \cdot W \cdot Embed^{InterC}(c_j)). \tag{12}$$

Finally, each stock's intra-cluster embedding is concatenated with the embedding of the cluster where it belongs to generate the final cross-sectional representation. This operation is presented in Eq. (13).

$$Embed^{cs}(s_i) = [Embed^{IntraC}(s_i) \parallel Embed^{InterC}(c)], \quad \forall \text{ stock } s_i \text{ and } s_i \in \text{ cluster } c. \tag{13}$$

In practice, we also apply multi-head attention mechanism in this module, which can enhance GAT's ability to extract structural information (Veličković et al., 2017). The multi-head's result of intra-cluster GAT is concatenated together while that of inter-cluster GAT is averaged element-wisely.

3.4. Prediction module

By combining the results of *temporal embedding module* and *cross-sectional embedding module*, PE-Net finally represents each stock according to Eq. (14):

$$Embed(s) = [Embed^{ts}(s) \parallel Embed^{cs}(s)]. \tag{14}$$

Then, a fully-connected layer is applied to map each embedding vector into a two-dimensional vector and the Softmax function is applied to calculate the expected price up and down probabilities. Softmax function is a non-linear function widely used in deep learning, which can be taken as a generalization of sigmoid function. Its definition is as shown in Eq. (15):

$$Softmax(v_j) = \frac{e^{v_j}}{\sum_{i=k}^{K} e^{v_k}}, \tag{15}$$

where v is a k-dimensional vector and v_k denotes its kth component.

4. Experiments

4.1. Experimental settings

4.1.1. Dataset

We use daily price data on 198 constituent stocks of S&P 500 for a total of 700 days from 2011 to 2013 (Cheng and Li, 2021). The whole dataset is divided into 3 parts, out of which the first 560 days' data is used for training, the following 70 days is used for validation and the last 70 days is used for test.

4.1.2. Learning and evaluation

Following the paradigm for binary classification problem, we assign two labels i.e. $\{0, 1\}$ to all the samples and apply cross-entropy as our loss function. Specifically, if the closing price of the current day is less than that of the previous day, then we label this day as 0, otherwise 1. For each input sample $(x_1, x_2, \ldots, x_n)$, its label depends on the label of day $n+1$, which is 1 if day $n+1$'s label is 1, and 0 otherwise. As such, we label all the samples and select *Cross Entropy* as the loss function, whose definition is shown in Eq. (16):

$$Loss(p, q) = -\sum_i p(i) log(q(i)), \tag{16}$$

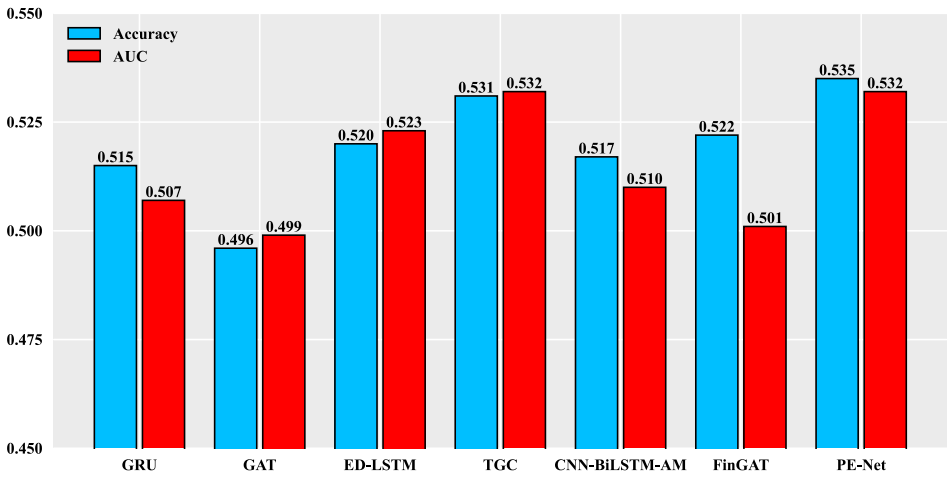


Fig. 4. Comparison between PE-Net and baseline models on S&P 500 constituents w.r.t **accuracy** and **AUC**. It demonstrates that PE-Net is the best-performing model in terms of both these evaluation metrics.

where p denotes the groundtruth distribution and q denotes the estimated or predicted distribution. We select the most widely-used *Accuracy* and *AUC* (Area Under Curve) as our evaluation metrics.

The hyperparameters of PE-Net are input sequence length l representing the time span of historical price data fed into it and the temporal embedding dimension D acting as a bridge between the *temporal embedding* and *cross-sectional embedding modules*. We apply gird search to find out the optimal hyperparameter combination. And in order to eliminate the randomness, under each hyperparameter setting, we train PE-Net for 10 times and select the best 5 performing models in terms of their performances on validation set. Then, we average the experimental results of the best 5 models with optimal hyperparameters on test set and use the average as the final reported result.

4.1.3. Software and programming

Our whole experiments are conducted with Python and Python-based deep learning framework Pytorch. Especially, the construction of GNN parts is completed using Pytorch-based library DGL. Most computations involving neural networks are done on NVIDIA GeForce GTX 1650.

4.1.4. Baseline models

To better examine the effectiveness of PE-Net, we compare it with some other models. The list of baseline models is shown below:

- **GRU**: single GRU model without any motifs or cluster design. It only accepts the raw price sequence as input (Cho et al., 2014);
- **GAT**: single GAT model without any motifs or cluster design. It directly takes the raw price sequence as the initial node representation and the graph structure it uses is a fully-connected graph derived from the stock pool (Veličković et al., 2017) ;
- **ED-LSTM**: a state-of-the-art event-driven LSTM model, which effectively utilizes the fundamental and media news information to make prediction (Li et al., 2020b);
- **TGC**: a state-of-the-art GNN-based model, which applies a GCN-like model to deal with a pre-defined heterogeneous graph, whose edges represent sector-industry and wiki-based relations respectively (Feng et al., 2019);
- **CNN-BiLSTM-AM**: a state-of-the-art model utilizing CNN, bi-directional LSTM, and attention mechanism to extract various stock features for prediction (Lu et al., 2021);
- **FinGAT**: a state-of-the-art model, which applies GNN to exploit latent relations between stocks based on a sector-industry network (Hsu et al., 2021);
- **PE-Net**: the model proposed in this paper.

4.2. Experimental results

4.2.1. Comparison

The comparison result between our model PE-Net and baseline models is shown in Fig. 4. As we can see, PE-Net achieves the best performance in terms of prediction accuracy. The two most important modules of PE-Net are temporal embedding and cross-sectional embedding modules, which are based on GRU and GAT models respectively. However, the accuracy results of GRU and GAT indicate that simply using either of them is not enough to well predict the price trend compared with our model, whose accuracy outperforms them by 3.9% and 7.9% respectively. This is a strong proof of the effectiveness of PE-Net leveraging both temporal and cross-sectional information. Furthermore, PE-Net is still the relatively better one w.r.t. accuracy, when competing with the state-of-the-art models. The improvement of PE-Net compared with ED-LSTM is significant, even though ED-LSTM considers more types of information except price, like fundamental indicators and media textual data. In addition, PE-Net also outperforms TGC w.r.t. accuracy, albeit only by 0.8%, but TGC is built on a pre-defined graph, which consists of the information of sector-industry and firm's events. This is also the case for FinGAT, which is based on a pre-defined sector-industry graph but is also outperformed by PE-Net significantly. For CNN-BiLSTM-AM, the gap between it and PE-Net on accuracy is even larger, which is another strong proof of the superiority of PE-Net. As for AUC, the situation is quite similar. PE-Net outperforms all the baseline models except TGC, which shares the same AUC result with it. In brief, the proposed PE-Net is the best-performing one w.r.t. both accuracy and AUC metrics.

Given that PE-Net only needs price data to make predictions, the experimental result described above is somewhat encouraging. It proves that we can simply use price data to achieve comparable or even better prediction effect than other methods, where the data used is often more extensive and complex. From the perspective of model architecture, PE-Net uses SOM and GAT to dynamically capture the latent correlations and information transferring weights between stocks. This design is more reasonable and effective than using explicit relationship networks represented by TGC and FinGAT, which are usually static and not all-inclusive. Also, from the perspective of data usage, PE-Net provides a new angle for stock prediction problems, which is to more efficiently extract the various information contained in price data rather than to look for alternative data. On the one hand, alternative data is relatively difficult to collect and process, e.g. the media textual data used in ED-LSTM and wiki-relation data used in TGC. On the other hand, considering the low signal-noise ratio of financial data, more types of data tend to be accompanied by more noises, which might in turn

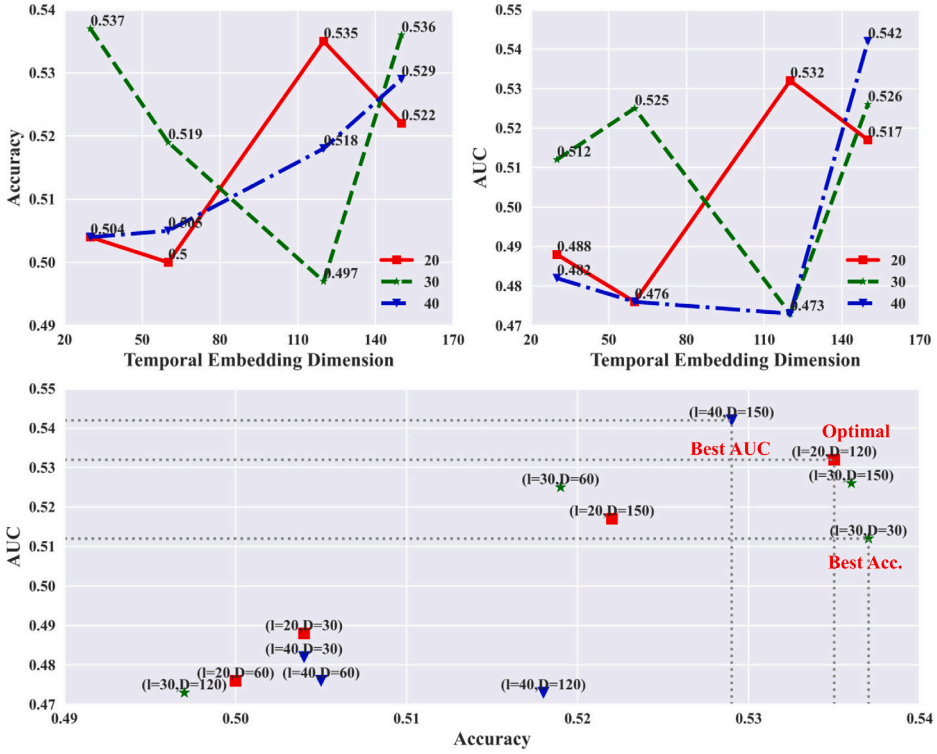


Fig. 5. Experimental results on test set under different **hyperparameter** settings.

affect the performance of prediction. Noteworthily, the motif-extraction method applied in PE-Net is aimed to eliminate noise. Therefore, this might be another critical reason for the good performance of our model.

4.2.2. Hyperparameter sensitivity

As it is mentioned before, there are two hyperparameters in PE-Net: input sequence length l representing the time scale of historical price data considered and temporal embedding dimension D, which is a key intermediate variable connecting the *temporal embedding* and *cross-sectional embedding modules*. Since the input series are used to predict the price movement for the next 1 day, the time period to look back should not be set too long. Thus, we select l from $\{20, 30, 40\}$, which means the numbers of trading days in 1, 1.5 and 2 months respectively. As for temporal embedding dimension D, we select it from $\{30, 60, 120, 150\}$. Note that the selection is entirely empirical because there is rarely any theoretical guidance here. Due to the use of 6 attention heads in our GAT design, the actual intermediate representation dimension ranges from 180 to 900. The accuracy and AUC results of PE-Net under different hyperparameter settings on test set are presented below in Fig. 5. The two subplots in the upper half illustrate the variation of accuracy and AUC with the change of l, while the subplot in the lower half presents the position of each hyperparameter combination with accuracy and AUC as the horizontal and vertical coordinates.

As we can see, both these hyperparameters can cause changes in the performance of PE-Net and these changes are reflected differently in different metrics. As for accuracy, when l is fixed at 40, the performance of PE-Net gets better as the embedding dimension increases. However, this is not the case when l is set as 20 or 30, where the model's accuracy performance experiences a fluctuation with an increasing D. Similarly, the fluctuation also appears w.r.t AUC as the hyperparameter varies. The highest AUC value for $l = 20$ is achieved as D reaches 120, but for $l = 30$ and 40, the spike emerges when D is set as 150. Additionally, if we review accuracy and AUC together, the optimal hyperparameter setting would be $(l = 20, D = 120)$, in which case PE-Net accomplishes a good level in terms of both these metrics. However, if we focus on one specific metric and try to maximize it, we can turn to other hyperparameter combinations like $(l = 40, D = 150)$ for the best AUC and $(l = 30, D = 30)$ for the best accuracy. Overall, PE-Net can be adapted to different scenarios by changing its hyperparameter settings, and the most appropriate setting totally depends on the user's target like the best accuracy, the best AUC or the balance between them. Notably, in the previous *Comparison* section, the results we show is under the $(l = 20, D = 120)$ setting.

4.2.3. Ablation study

The comparison results of PE-Net with GRU and GAT presented in *Comparison* section have already illustrated the usefulness of the combination of our *temporal* and *cross-sectional embedding modules*. Here, we work further to examine the effectiveness of other parts of PE-Net through ablation study, i.e. motif extraction in *preprocessing module*, intra-cluster embedding in *cross-sectional embedding module* and inter-cluster embedding in *cross-sectional embedding module*. The details of each model are described as follows:

- **PE-Net**: the model proposed in this paper;
- **without Motif-Extr.**: the rest of PE-Net removing the *preprocessing module*, which takes the raw price sequence directly as the input;
- **without Intra-Embd.**: the rest of PE-Net removing the intra-cluster embedding operation, which directly uses the output of *temporal embedding module* as the initial representation for inter-cluster embedding;
- **without Inter-Embd.**: the rest of PE-Net removing the inter-cluster embedding operation, which directly transfers the output of intra-cluster embedding into the final fully-connected layer without adding inter-cluster information.

We test them on the same S&P 500 dataset under 3 sets of hyperparameter settings i.e. $(l = 20, D = 120)$, $(l = 30, D = 30)$, and $(l = 40, D = 150)$, which are the optimal, the best accuracy and the best AUC combinations respectively. The experimental results are shown in Table 1.

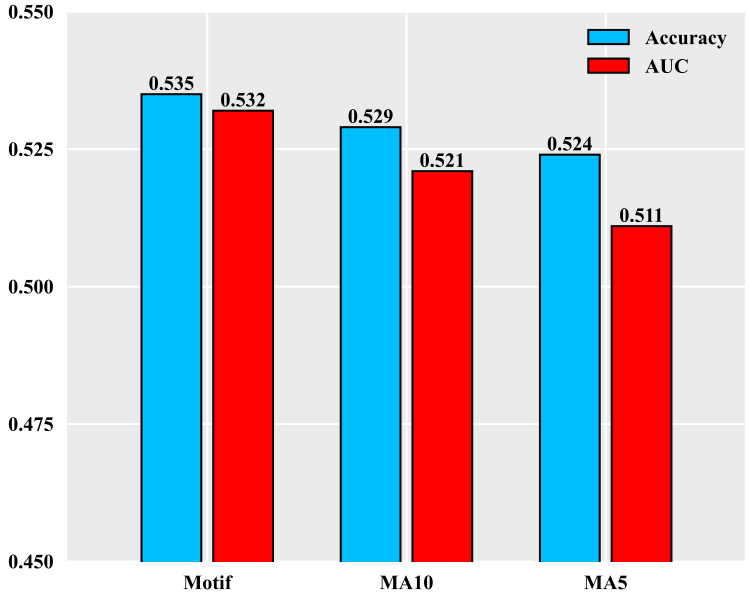


Fig. 6. The performance of PE-Net with **motif-extraction**, **10-day moving average** and **5-day moving average** on S&P 500 constituents.

Table 1
Comparison between PE-Net and **ablation-study-related** models on S&P 500 constituents under different hyperparameter settings.

	(l = 30, D = 30)		(l = 20, D = 120)		(l = 40, D = 150)	
	Accuracy	AUC	Accuracy	AUC	Accuracy	AUC
PE-Net	**0.537**	**0.512**	**0.535**	**0.532**	**0.529**	**0.542**
without Motif-Extr.	0.528	0.511	0.522	0.514	0.518	0.523
without Intra-Embd.	0.513	0.487	0.514	0.503	0.503	0.506
without Inter-Embd.	0.515	0.497	0.528	0.513	0.521	0.510

As it illustrates, PE-Net with complete architecture remains the best performing one among all the tested models w.r.t. accuracy and AUC under every hyperparameter setting. And the removal of intra-cluster embedding operation leads to the most drastic drop on both metrics, indicating that this operation is crucial to our model. From the angle of finance, it further implies that leveraging the information of the peer group of a firm, in which "peer" is not necessarily explicitly defined, can help us predict that firm's price movement. As for inter-cluster embedding, its absence also causes the decrease of accuracy and AUC, and this decrease is particularly sharp on accuracy when PE-Net is under $(l = 30, D = 30)$ setting. Finally, the results also point out that our *preprocessing module* is of great importance as the model without it cannot perform as well as the complete model. This is also understandable from finance since frequent price pattern or trend is often what we really need and care about instead of the exact price values.

4.2.4. Effectiveness of motif extraction

The significant drop in both accuracy and AUC of PE-Net after removing *preprocessing module* has already demonstrated the indispensability of this module. In this section, we continue to examine the effectiveness of this module. In financial data processing, *moving average* is one of the most commonly used operations, especially when dealing with price data. Therefore, we apply 5-day and 10-day moving average to preprocess price sequence respectively, and feed the processed data into PE-Net as a comparison with the original motif-extraction operation. Fig. 6 shows the performances of PE-Net with motif-extraction, 10-day moving average and 5-day moving average.

As we can see, the advantages of PE-Net with motif-extraction over those with moving average are obvious, especially w.r.t AUC. This result can be regarded as a solid proof that the motif-extraction designed in PE-Net is a better trend-capturing method than the classic moving average operation in this scenario. We argue that this might be because the aim of moving average is to smooth the data and therefore it is not capable of capturing the short-term trend when the whole sequence is relatively volatile.

4.2.5. Effectiveness of stock clustering

Recall that the graph construction method we propose in *cross-sectional embedding module* is based on clustering, where the historical adjusted sequences are fed and processed by SOM to cluster the stocks. In this section, we aim to analyze the clustering results in depth and try to figure out whether such results make sense. In particular, we compare the clustering results with real-world sector classification to discover the connections between them.

We focus on the stock pools during test period containing 198 stocks within 70 days. According to GICS (Global Industry Classification Standard), all these stocks can be grouped into 12 sectors, which are Financials, Consumer Discretionary, Industrials, Health Care, Information Technology, Consumer Staples, Utilities, Energy, Materials, Communication Services, Medical and Real Estate respectively. Based on this, we review the sector group for each stock in each cluster each day and count the total number of times different sectors co-occur in the same cluster. Then, we apply row-max–min normalization to the counting result and visualize it with a heatmap, which is exhibited in Fig. 7. In this heatmap, each row represents the normalized number of times a certain sector co-occurs with the others during test period and the color becomes lighter as the number of co-occurrences increases.

As we can see, our clustering result reflects the correlation between sectors to some extent. For instance, in Industrials row, Financials, Utilities and Communication Services have the lightest color except Industrials itself, which implies that these 3 sectors are most relevant to Industrials. This is reasonable because the business chain of companies in Industrials sector involves finance, communication services and public utilities. In addition, we can also see a relatively strong correlation between Health Care and Medical in Health Care row, which corresponds to the fact that these two sectors often share

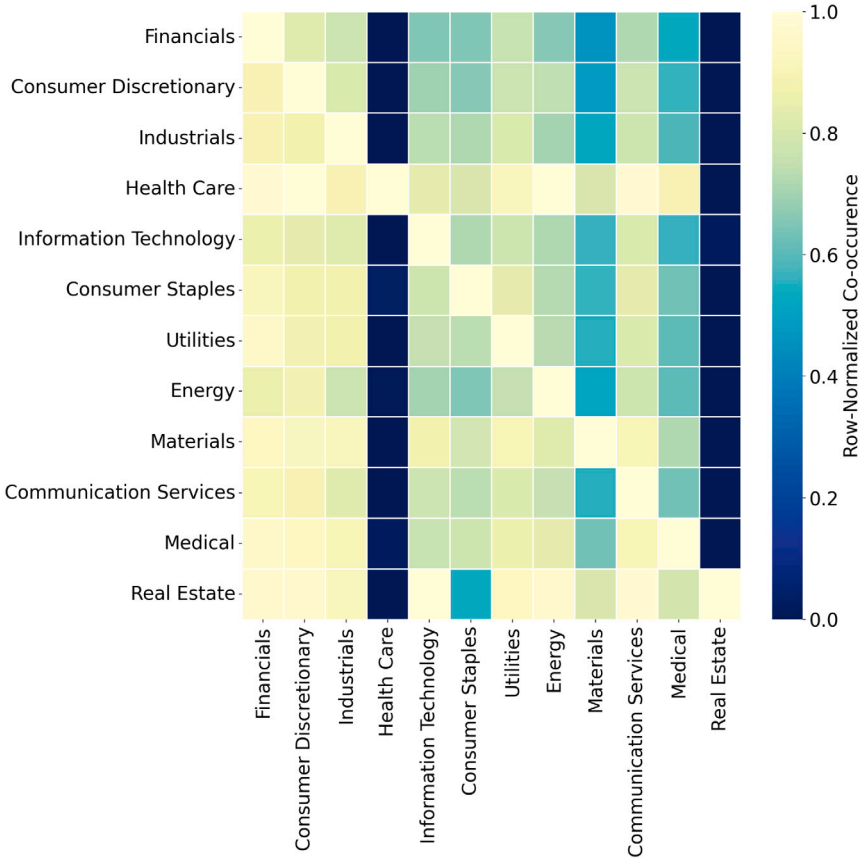


Fig. 7. Row-normalized co-occurrence.

the common customers and businesses. Furthermore, in the last row, which represents the case for Real Estate, we can find that Real Estate is strongly correlated with Financials and Consumer Discretionary. Considering that these 3 sectors all have a close tie with households' income and consumption, this is also understandable. Overall, this co-occurrence analysis partly shows that our clustering results can reflect the basic sector relevance in real world.

In addition to the macro sector perspective, we also analyze the co-occurrences between individual stocks. Fig. 8 shows the co-occurrence statistics charts of CMS (CMS Energy Corp.) and KR (The Kroger Co.). As is demonstrated in Fig. 8(A), HON (Honeywell International Inc.), SRE (Sempra Energy) and FE (FirstEnergy Corp.) are the three companies having the most co-occurrences with CMS Energy Corp. And except for HON, which belongs to Industrials sector, all other companies belong to Utilities sector including CMS itself. Moreover, even though Industrials and Utilities are two different sectors, they have a close connection with one another as is mentioned before. And if we turn to Fig. 8(B), we can find the similar situation. While the targeted company KR (The Kroger Co.) belongs to Consumer Staples sector, the other 3 companies marked, EL (The Estee Lauder Companies Inc), MO (Altria Group Inc) and COST (Costco Wholesale Corp), all belong to Consumer Staples sector as well. These results fairly illustrate that our clustering method has captured the sector information of individual stock.

In a word, according to the above analysis, we argue that the clustering method we propose in PE-Net manages to extract the real-world sector information contained in individual stock's price sequence without using any sector data beforehand. This is a strong proof of the rationality of the clustering result and the effectiveness of our stock clustering method. Additionally, it is also worth noting that sector information is only part of the information captured by our clustering method and it must also capture plenty of other potential information, which is not easy to explain due to the complexity of neural networks.

4.3. Trading simulation

In order to test whether the predictive ability of PE-Net can contribute to the trading in practice, we conduct a trading simulation on the test set. Note that PE-Net aims to predict the price trend of the next day, so it can be used for *market timing*, which is an investment strategy that determines when to enter or exit a market by predicting future price movements. Specifically, we form an investment pool to include all those 198 stocks and compare the PE-Net based market timing strategy (Trade according to the trading signal generated by PE-Net, i.e., buy if the price is predicted to rise and sell otherwise. And all stocks are traded with equal weight.) with the buy-and-hold strategy (Buy all stocks with equal weight at the beginning of the test period, hold them, and sell them at the end of the test period.) The value invested in both these strategies is unit 1 at the beginning of the test period. Let $\{v_0, v_2, \dots, v_T\}$ denote the sequence of daily value during the test period. Then, the daily return sequence can be expressed as $\{r_1, r_2, \dots, r_T\}$, where r_t is calculated as $(v_t - v_{t-1})/v_{t-1}$. A total of five evaluation metrics are considered, which are

- Return: the annualized return defined as

$$(\frac{1}{T}\sum_{t=1}^{T} r_t) * 252;$$

- Volatility: the annualized volatility defined as

$$\sqrt{(\frac{1}{T}\sum_{t=1}^{T}(r_t - \bar{r})^2) * 252},$$

where $\bar{r}$ is the mean of $\{r_1, r_2, \dots, r_T\}$;

- MDD: the maximum drawdown defined as the largest consecutive value drop from a peak to a trough in $\{v_1, v_2, \dots, v_T\}$;

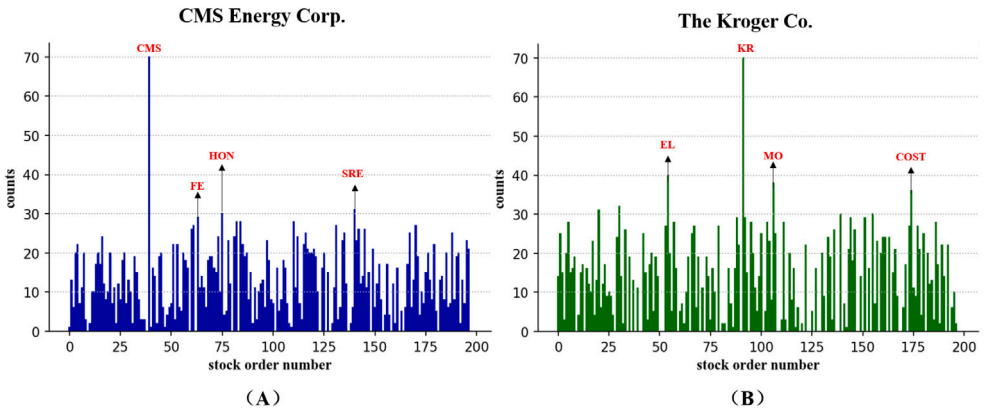


Fig. 8. (A) The co-occurrences between **CMS Energy Corp.** and the other stocks; (B) the co-occurrences between **The Kroger Co.** and the other stocks;.

Table 2
The performance of the market-timing strategy and buy-and-hold strategy on test set.

	Return	Volatility	MDD	Sharpe	Calmar
market timing	**0.268**	**0.303**	**4.89%**	**0.884**	**5.478**
buy-and-hold	0.221	0.338	5.23%	0.655	4.236

- Sharpe: the annualized Sharpe ratio defined as Return/Volatility[1];
- Calmar: the annualized Calmar ratio defined as Return/MDD.

The performance of the market timing strategy and buy-and-hold strategy is presented in Table 2. As is shown, with PE-Net to do market timing, the investment can achieve greater profits with less risk compared with the investment simply based on buy-and-hold. It can be seen that the total return of market timing strategy is higher than that of buy-and-hold strategy. Also, the risk of the market timing strategy, whether measured by Volatility or MDD, is less than that of buy-and-hold strategy. Moreover, when profit and risk are considered together, the PE-Net based market timing strategy is still the one that performs better, supported by the higher Sharpe and Calmar ratio.

5. Conclusion

Stock prediction is a widely concerned problem in both academia and industry, but it is also full of difficulties and challenges. In this paper, we propose a novel neural-network-based model PE-Net and demonstrate its good prediction performance through extensive experiments on real-world dataset. The data needed by PE-Net is only historical price and the whole model is designed to extract rich information from price data. PE-Net firstly detects the motif structures contained in each price sequence and reconstructs the sequence to better reflect the price trend. Then, PE-Net clusters all the stocks based on their own adjusted sequences to derive a stock graph and uses neural-network-based methods to extract both temporal and cross-sectional information which is eventually used to calculate the price up and down probabilities. We test our model on S&P 500 constituents and the experimental results illustrate that our model outperforms the state-of-the-art models w.r.t. both accuracy and AUC. Furthermore, we also conduct ablation study and module effectiveness test to examine the utility of each module in PE-Net. And the test results confirm the effectiveness and indispensability of our specific module design.

In the future, this work could be extended in two aspects. The motif-extraction algorithm could be extended to capture the price trends on different time scales instead of single scale. This would contribute to the operations like clustering and temporal embedding. Furthermore, the clustering method could also be extended, e.g. clustering stocks based on different time spans, different frequencies or a combination of them.

[1] In standard Sharpe ratio calculation, the Return needs to be subtracted by a risk-free rate. Here, we assume the risk-free rate is equal to 0.

CRediT authorship contribution statement

Bowen Pang: Conceptualization, Methodology, Software, Formal analysis, Investigation, Resources, Data curation, Writing – original draft, Visualization. **Wei Wei:** Conceptualization, Writing – review & editing, Supervision, Project administration, Funding acquisition. **Xing Li:** Resources, Software, Validation. **Xiangnan Feng:** Resources, Data curation, Validation. **Chao Li:** Resources, Validation.

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.

References

Ariyo, A.A., Adewumi, A.O., Ayo, C.K., 2014. Stock price prediction using the ARIMA model. In: 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation. IEEE, pp. 106–112.

Chen, W., Jiang, M., Zhang, W.-G., Chen, Z., 2021. A novel graph convolutional feature based convolutional neural network for stock trend prediction. Inform. Sci. 556, 67–94.

Chen, Y., Wei, Z., Huang, X., 2018. 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. pp. 1655–1658.

Cheng, L.-C., Huang, Y.-H., Wu, M.-E., 2018. Applied attention-based LSTM neural networks in stock prediction. In: 2018 IEEE International Conference on Big Data (Big Data). IEEE, pp. 4716–4718.

Cheng, R., Li, Q., 2021. 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. pp. 55–62.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Cohen, L., Frazzini, A., 2008. Economic links and predictable returns. J. Finance 63 (4), 1977–2011.

Fama, E.F., 1965. The behavior of stock-market prices. J. Bus. 38 (1), 34–105.

Feng, F., He, X., Wang, X., Luo, C., Liu, Y., Chua, T.-S., 2019. Temporal relational ranking for stock prediction. ACM Trans. Inf. Syst. (TOIS) 37 (2), 1–30.

Hamilton, J.D., 1989. A new approach to the economic analysis of nonstationary time series and the business cycle. Econometrica 357–384.

Haugen, R.A., Lakonishok, J., 1987. The Incredible January Effect: The Stock Market's Unsolved Mystery. Irwin Professional Pub.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.

Hooi, B., Liu, S., Smailagic, A., Faloutsos, C., 2017. B eat L ex: Summarizing and forecasting time series with patterns. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, pp. 3–19.

Hsu, Y.-L., Tsai, Y.-C., Li, C.-T., 2021. FinGAT: Financial graph attention networks for recommending top-K profitable stocks. arXiv preprint arXiv:2106.10159.

Jegadeesh, N., Titman, S., 1993. Returns to buying winners and selling losers: Implications for stock market efficiency. J. Finance 48 (1), 65–91.

Jiang, W., 2021. Applications of deep learning in stock market prediction: recent progress. Expert Syst. Appl. 184, 115537.

Khaidem, L., Saha, S., Dey, S.R., 2016. Predicting the direction of stock market prices using random forest. arXiv preprint arXiv:1605.00003.

Kipf, T.N., Welling, M., 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Kohonen, T., 1990. The self-organizing map. Proc. IEEE 78 (9), 1464–1480.

Kumbure, M.M., Lohrmann, C., Luukka, P., Porras, J., 2022. Machine learning techniques and data for stock market forecasting: a literature review. Expert Syst. Appl. 116659.

Lee, M.-C., 2009. Using support vector machine with a hybrid feature selection method to the stock trend prediction. Expert Syst. Appl. 36 (8), 10896–10904.

Li, J., Bu, H., Wu, J., 2017. Sentiment-aware stock market prediction: A deep learning method. In: 2017 International Conference on Service Systems and Service Management. IEEE, pp. 1–6.

Li, R., Liang, H., Shi, Y., Feng, F., Wang, X., 2020a. Dual-CNN: A Convolutional language decoder for paragraph image captioning. Neurocomputing 396, 92–101.

Li, Q., Tan, J., Wang, J., Chen, H., 2020b. A multimodal event-driven lstm model for stock prediction using online news. IEEE Trans. Knowl. Data Eng. 33 (10), 3323–3337.

Lo, A.W., MacKinlay, A.C., 2011. A Non-Random Walk Down Wall Street. Princeton University Press.

Lu, W., Li, J., Wang, J., Qin, L., 2021. A CNN-BiLSTM-AM method for stock price prediction. Neural Comput. Appl. 33 (10), 4741–4753.

Matsuba, I., 1991. Application of neural sequential associator to long-term stock price prediction. In: [Proceedings] 1991 IEEE International Joint Conference on Neural Networks. IEEE, pp. 1196–1201.

Moskowitz, T.J., Grinblatt, M., 1999. Do industries explain momentum? J. Finance 54 (4), 1249–1290.

Osborne, M.F., 1959. Brownian motion in the stock market. Oper. Res. 7 (2), 145–173.

Rumelhart, D.E., Hinton, G.E., Williams, R.J., 1986. Learning representations by back-propagating errors. Nature 323.

Tian, J., Azarian, M.H., Pecht, M., 2014. Anomaly detection using self-organizing maps-based k-nearest neighbor algorithm. In: PHM Society European Conference, Vol. 2.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., 2017. Graph attention networks. arXiv preprint arXiv:1710.10903.

Wang, J., Zhang, S., Xiao, Y., Song, R., 2021. A review on graph neural network methods in financial applications. arXiv preprint arXiv:2111.15367.

Wen, M., Li, P., Zhang, L., Chen, Y., 2019. Stock market trend prediction using high-order information of time series. IEEE Access 7, 28299–28308.

Wu, J.M.-T., Li, Z., Herencsar, N., Vo, B., Lin, J.C.-W., 2021. A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. Multimedia Syst. 1–20.

Yao, S., Luo, L., Peng, H., 2018. High-frequency stock trend forecast using LSTM model. In: 2018 13th International Conference on Computer Science & Education. ICCSE, IEEE, pp. 1–4.

Zhang, L., Aggarwal, C., Qi, G.-J., 2017. Stock price prediction via discovering multi-frequency trading patterns. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 2141–2149.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. AI Open 1, 57–81.