Information Sciences 588 (2022) 405–424

Contents lists available at ScienceDirect

Information Sciences

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

Price graphs: Utilizing the structural information of financial time series for stock prediction

Junran Wu [a], Ke Xu [a], Xueyuan Chen [a], Shangzhe Li [b], Jichang Zhao [c,*]

[a] State Key Lab of Software Development Environment, Beihang University, China
[b] School of Mathematics Science, Beihang University, China
[c] School of Economics and Management, Beihang University, China

ARTICLE INFO

Article history:
Received 17 June 2021
Received in revised form 18 December 2021
Accepted 25 December 2021
Available online 31 December 2021

Keywords:
Stock prediction
Complex network
Time series graph
Graph embedding
Structure information

ABSTRACT

Great research efforts have been devoted to exploiting deep neural networks in stock prediction. However, long-term dependencies and chaotic properties are still two major issues that lower the performance of state-of-the-art deep learning models in forecasting future price trends. In this study, we propose a novel framework to address both issues. Specifically, in terms of transforming time series into complex networks, we convert market price series into graphs. Then, structural information, referring to temporal point associations and node weights, is extracted from the mapped graphs to resolve the problems regarding long-term dependencies and chaotic properties. We take graph embeddings to represent the associations among temporal points as the prediction model inputs. Node weights are used as a priori knowledge to enhance the learning of temporal attention. The effectiveness of our proposed framework is validated using real-world stock data, and our approach obtains the best performance among several state-of-the-art benchmarks. Moreover, in the conducted trading simulations, our framework further obtains the highest cumulative profits. Our results supplement the existing applications of complex network methods in the financial realm and provide insightful implications for investment applications regarding decision support in financial markets.

1. Introduction

Financial time series prediction, in particular the stock prediction, which aims at forecasting the future trends of stock prices, is one of the key foundational techniques in investment and has attracted tremendous attention from various fields [27,42,40]. Various categories of methods (e.g., technical analysis, machine learning and deep learning models) and data sources (e.g., historical market price data, common funds and financial news) have been adopted for stock prediction [27,39,41], via the modeling of the relationship between the historical behavior and future trend [10,30]. While most of these traditional efforts, although equipped with state-of-the-art prediction algorithms, concentrate only on the structures among time series, the structures within time series are hardly investigated for stock prediction. With increasing literature leveraging complex network methods for characterizing dynamical systems derived from time series, complex networks have already served as promising and versatile tools for adopting structural information embedded within the temporal points of time series and providing new insights for stock prediction [38,49].

* Corresponding author.
E-mail address: jichang@buaa.edu.cn (J. Zhao).

https://doi.org/10.1016/j.ins.2021.12.089
0020-0255/

In the current literature regarding stock prediction, without glimpsing at the structural information of stock time series, most research efforts have been devoted to various types of Internet information sources and dynamic indicators derived from stock prices [30,40]. Currently, deep learning has become an effective method to refine multiaspect features from complex financial time series. Numerous deep learning frameworks have been proposed in the literature with the aim of predicting asset prices [27,10]. To capture the interactions among multiple variables, attention mechanisms have also been deployed for time series forecasting [27,13,5]. However, the long-term dependencies of financial time series in experiments still have not been fully captured due to the complex temporal evolution of the interactions among all the temporal points. The value of a data point at time t is not likely independent of its temporally neighboring points or historical points that occur far before t. Such associations are denoted as short-term and long-term dependencies, respectively in deep learning [2,13]. In the financial realm, it is analogously vital to utilize the dependencies among the values in a given series, and the long-term memories have also been uncovered and validated on daily, weekly, monthly, and annual stock returns [20]. On the other hand, financial time series often represent chaotic and complex price movement behaviors [9]. As a result, stock prices are shown in non-stationary time series, and have abrupt changes or unexpected reversals that are taken as outliers in modeling and undermine the generalization ability of learning models. In particular, chaotic properties seriously challenge these models and make them assign incorrect weights to points that are unhelpful for forecasting further trends. Consequently, the predictive capability of current models is limited, as it is profoundly undermined by both of the abovementioned issues.

Because complex network theory and nonlinear time series analysis are generally considered domains of complex systems science, the adoption of complex network methods has become a popular approach in nonlinear time series analysis, thereby allowing fundamental questions regarding long-term dependencies and chaotic properties in time series prediction to be addressed [26,38,49]. For example, protein structural classes have been successfully predicted by mapping protein time series into recurrent networks [25]. Moreover, the virtual graphs derived from response time series of a marine system have helped forecast system catastrophes [48], suggesting that the topological characteristics of time series graphs actually contain latent information to ascertain the future states of a chaotic trajectory. For financial time series, an exchange rate series has been shown to convert into scale-free and hierarchically structured graphs [45]. Therefore, in the context of various graphs that are converted from time series, structural information, referring to the associations among temporal points and the node weights, provides promising assistance for financial time series prediction. First, given the existence of explicit edges among distant nodes in converted graphs, the long-term dependencies in a time series can be directly captured through the associations among temporal points [26]. By bridging distant temporal points, long-range information can be delivered more quickly through these edges, thereby preventing information vanishing in recurrent deep learning. Additionally, by identifying prominent content from chaotic time series, the weights of graph nodes can provide additional knowledge for temporal attention to tackle chaotic properties of financial time series [38].

In this paper, we propose a novel graph-based framework for stock prediction. Our framework consists of two main modules. The first is a time series embedding module, which is used to map time series into graphs and extract structural information from the corresponding graphs. In this module, given a financial time series, which includes not only four kinds of stock prices (the closing price, high price, low price, and opening price) but also the volume and amount of share trading at each time interval [34], the visibility graph (VG) algorithm is employed for time series transformation [17], in which the mapped time series graph is denoted as the price graph. To overcome the above fundamental questions regarding prediction (e.g., long-term dependencies and chaotic properties), struc2vec is further adopted to learn node embeddings to preserve the associations among temporal points [28], and the collective influence (CI) algorithm is employed for measuring the node weights of price graphs [22]. The second part is a prediction module based on neural networks. Specifically, to complement the loss of temporal sequences in the structural information of stock time series, attention-based recurrent neural networks (RNNs) are employed to resolve the implicit long-term dependencies and the chaotic evolution of the temporal points. Then, a self-attention layer is used to fully model the structure among stocks. Finally, with the obtained hidden representation of each stock, the corresponding movement direction is extracted from a final nonlinear fully connected layer.

To inspect the power of our framework, in this paper, we conduct numerous experiments on real-world market data from the Chinese market index (China Securities Index 300, i.e., CSI-300), which contains 300 stocks with data from 2010 to 2019. The experimental results demonstrate the effectiveness of the structural information extracted from price graphs for the task of stock prediction. A trading simulation is also performed based on signals produced by the prediction module to validate the profitability and stability of our framework. The contributions of our work can be summarized as follows:

- For the first time, by utilizing complex network methods that bridge time series and graphs, we leverage the structural information obtained from market price data for stock prediction.
- We develop a novel framework based on the structural information embedded in price graphs, and this framework is capable of addressing fundamental questions regarding long-term dependencies and chaotic properties in stock prediction.
- We empirically reveal the effectiveness of structural information and the proposed framework for stock prediction on real-world data, i.e., our approach outperforms state-of-the-art baselines in terms of testing accuracy and obtains the highest average return (47.91%) in trading simulations.

The remainder of this paper is organized as follows. Section 2 discusses the related literature. Section 3 specifies the details of our proposed framework and structural information extraction methodology. Section 4 depicts the experimental settings, including the data, compared baselines and model parameters. The subsequent section reveals the experimental prediction results. We conduct a further analysis and market trading simulation in Section 6. Finally, Section 7 concludes this work and presents some limitations about future research directions.

2. Related work

In this section, we review the relevant literature streams regarding financial series prediction and graph learning to position our research vis-á-vis the findings from extant research.

2.1. Financial time series prediction

Before deep learning methodologies became popular, statistical and machine learning models were universally adopted for financial time series prediction because of their good interpretation capabilities. Commonly used statistical models include the autoregressive moving average (ARMA), generalized autoregressive conditional heteroskedasticity (GARCH) and nonlinear autoregressive exogenous (NARX) models, which were employed to verify assumptions regarding the finance market, and predictions were accordingly based on these verified assumptions. However, chaotic behaviors often appear in financial time series. As a result, the predictive capability of the models mentioned above is undermined because of their inability to shape the evolutionary process in financial systems [9]. To improve the modeling of chaotic time series, several nonlinear learning models have been developed. In particular, machine learning methods provide a strong capability to learn the underlying relationships among patterns between features and targets [35,40]. These methods focus on maximizing the prediction accuracy based on a wide variety of models. However, a limitation exists among these methods, that is the adoption of a predefined nonlinear framework which may not be consistent with the true underlying nonlinear form [27]. In addition, the effectiveness of traditional statistical models or machine learning algorithms mostly relies on the quality of the input features, which enables improper features to possibly undermine model performance because of chaotic properties of financial time series.

In recent years, extensive studies have been undertaken to solve financial time series prediction problems using deep learning methods [27,10,13] because of their powerful expression ability [15]. RNNs [29], deep neural networks especially developed for sequence data, have been in vogue because of their superior performance in capturing nonlinear relationship. However, traditional RNNs are insufficient in capturing long-term dependencies owing to the issue of gradients vanishing [2]. Based on "memory cells" that are designed to preserve information for a longer time, a special type of RNN, Long short-term memory (LSTM), has been developed and proven to be useful in predicting stock returns [14,10]. Therefore, LSTM is constantly employed for sequential data or financial time series prediction and performs better than RNNs [10]. However, the shortage in capturing long-term dependencies still exists; that is, the performance of LSTM networks deteriorates rapidly when the length of the input sequence increases [27]. Furthermore, owing to the noise amplification in the model recurrence process, chaotic properties of financial time series worsen the prediction performance [14].

Because the concept of attention represents the human intuition by which some portions of data are given more emphasis than others, deep learning methods based on attention mechanisms are widely used to learn the complex dependencies among features in time series tasks. Based upon this, a dual-stage attention-based RNN (DARNN) was proposed [27]. In the first encoder stage, an input attention mechanism automatically extracts the crucial input features at each iteration based on the previous hidden state of the encoder, which gives greater emphasis to more informative features from chaotic series. In the second decoder stage, a temporal attention mechanism is designed to select crucial encoder hidden states by referring to all time steps, thereby rebalancing the information at each temporal point to capture long-term dependencies. In another recent work, a cross-attention stabilized fully convolutional neural network (CA-SFCN), which also adopts variable and temporal attention, was proposed to classify multivariate time series [13]. While these models implicitly capture certain long-term dependencies, there is still rare attention that explicitly exploits the structure within time series, i.e., the direct links among distant temporal points. Through the bridging of distant temporal points to form edges, long-range information could be directly obtained through these edges to prevent information vanishing in traditional deep learning. We believe that more accurate predictions can be obtained by capturing this explicit structural information of the given time series.

2.2. Time series graphs

With increasing literature leveraging complex network methods for characterizing dynamical systems derived from time series, the adoption of complex network methods has become an active realm of nonlinear time series analysis, which has provided a guideline for addressing fundamental questions regarding long-term dependencies and chaotic properties in time series prediction [26,38,49]. By transforming time series into graphs, researchers are capable of measuring the structural properties of time series and capturing the hidden structures embedded within chaotic temporal points. Based on different rationals, there are three main kinds of methods designed to map a time series into a graph: (1) recurrence networks (RNs), which emphasizes the mutual statistical similarities or metric proximities among various segments of the time series [8]; (2)

VGs, which depict local convexities or record-breaking properties within a signal time series [17], and (3) transition networks (TNs), which profiles the transition probabilities between discrete states [24]. Among these three complex methods, there are many parameters that need to be optimized in RNs and TNs when converting time series. Although, there is also literature discussing the optimal choice of these parameters of RNs and TNs, a simple algorithm (VG) is much preferred for financial time series analysis because of the parameter-free nature [49]. In particular, the associated VG refines certain important features from the original time series, i.e., the structure within the time series [49].

Chaotic time series widely exist in various scenarios, including finance, biology, and meteorology, and there has been a wide range of recent applications of complex network methods for such chaotic data [25,48,45]. As reported in [25], the structural classes of proteins have been successfully predicted by transforming protein time series into recurrence networks. Moreover, another recent study showed that the virtual graph derived from the response time series of a marine system can be used to forecast system catastrophes [48]. Furthermore, in financial studies, the degree distributions of mapped graphs derived from the growth rates of gross domestic product series were found to be scale-free, and the degree distributions of converted graphs derived from the growth rates of three industry series were shown to be almost exponential [45,38]. Another study found that the markets in developed countries differ significantly from developing countries through analyzing the time series graphs mapped from stock market [4]. Specifically, the complexity of developing markets is disturbed and relatively low over some periods, whereas it is more stable and stronger over time for mature stock markets, suggesting a stronger long-range price memory and indicating that transforming financial time series into graphs can effectively capture chaotic characteristics of stocks.

Therefore, in the context of various graphs converted from time series, previous complex network methods naturally provided promising assistance for financial time series prediction. First, owing to the existence of explicit edges among distant nodes in converted graphs, the long-term dependencies in time series graphs can be directly captured through the associations among temporal points. Additionally, by identifying prominent content from chaotic time series, the weights of graph nodes provide additional knowledge for learning temporal attention to tackle chaotic properties of financial time series. However, despite the great potential of structural information for financial time series forecasting, little attention has been paid to the employment of complex network methods for obtaining more accurate predictions. Moreover, how to integrate the structural information of time series networks and deep learning methods remains an open problem.

2.3. Graph embedding

To employ graph structures for deep learning, graph embedding has aroused much research interest. With the advantage of preserving node content, graph structure, and additional information, graph embedding is capable of embedding graph nodes into latent, low-dimensional spaces [47]. After obtaining the new node representations, conventional vector-based learning methods can be conveniently and efficiently employed for graph analysis tasks; this inspires us to take advantage of structural information for time series prediction.

Traditional efforts in learning low-dimensional vectors for vertices in networks have been considerably successful with respect to performing prediction and classification tasks [28,6,46]. Based on the proximity of nodes derived from learned embeddings, the future interactions between users can be extracted from a social network [33]. In addition, through dynamically computing a user's recent preferences by referring to the embedding of check-in points of interest (POIs), a location-based model was introduced to help discover attractive and interesting POIs [43,19]. For financial problems, with a bipartite network constructed from mutual fund shareholding data in the real world, the intrinsic properties of stocks were extracted from the latent space to optimize technical indicators and target the critical factors of market crashes [18]. In addition to the interrelationships among stocks, various networks, which consists of financial institutions, cryptocurrencies, stock indices and stock sectors, are constructed for investment/portfolio assistances [3,16,32]. While excellent performance has been confirmed, these target graphs may not be applicable to more refined problems, such as daily stock prediction, because these graphs contain only coarse-grained information. Moreover, due to the connection among nodes established on assets, the weaved graphs are time invariant or evolve with a low frequency (e.g., monthly or seasonally in general); that is, the embedded information cannot change in a timely manner with the dynamics of the whole market, which may result in lower effectiveness in terms of daily stock prediction. Therefore, these target graphs are often adopted to identify the market states over a period or assist other methods that are capable of timely reaction [31,3,32]. With the emergence of time series graphs, more refined structural information can be obtained, which suggests great potential for financial time series prediction.

The extant literature on complex networks and graph embedding implies that the structural information extracted from time series graphs is capable of tackling issues regarding long-term dependencies and chaotic properties. In this paper, based on this structural information obtained from time series graphs, we propose a new framework to obtain more accurate stock predictions.

3. Proposed framework

This paper presents a novel trend prediction framework for stock prices. To conquer the shortcomings of the financial time series forecasting methods mentioned above, a module based on complex network methods and graph embedding is introduced to extract structural information from mapped graphs that structurally connect distant price points. The

predictions of stock trends are obtained from several attention-based layers and a fully connected classification layer. Fig. 1 illustrates the proposed framework, which is constructed with two modules. The first module is a time series embedding module, which takes raw market price data as inputs and aims to extract structural information, referring to the associations among temporal points and the node weights. The second module is a prediction module. Based on deep learning methods, stock representations are learned from structural information by incorporating temporal sequences for stock trend forecasting. The different modules of the proposed framework are discussed below in turn.

3.1. Time series embedding module

In this module, we convert raw market price data into time series graphs and measure the weights of graph nodes. Second, we explore the topological properties of the constructed graphs associated with raw market data, namely, the structural representations.

3.1.1. Time series graphs

As discussed above, there are three main types of complex network approaches for mapping individual time series into graphs, i.e., RNs, VGs and TNs. Among these three complex methods, the VG algorithm is widely employed for financial time series analysis because it is not influenced by any algorithmic parameters and maps time series into scale-free graphs [17]. Thus, we take the VG algorithm to map raw market price data into time series graphs, i.e., price graphs.

Consider a stock price series p_t with a length of T at time t. Each vertex in the converted graph corresponds to a data point in the original series. Based on the principle that if two data points can mutually be seen in the bar chart of the corresponding time series, an edge between the two points is established; put differently, two temporal points in series can be connected by a straight line when there are not any intermediate data heights that intersect this "visibility line". In a formal manner, a VG can be transformed from a time series under the next visibility specification [17]: given two data points (t_i, p_i) and (t_j, p_j) where $p_i, p_j > 0$ in a time series, there is a visibility line that connects the two data points in the converted graph if and only if all data points (t_k, p_k) such that $t_i < t_k < t_j$ satisfies

$$p_k < p_i + \frac{t_k - t_i}{t_j - t_i}(p_j - p_i). \tag{1}$$

For graphs converted from stock prices, each vertex represents the daily price under real-world trading circumstances. We can view the graph edges as signals of no abrupt price changes occurring during the period between two temporal points. Furthermore, on the basis of Eq. 1, we can find that the shortest path between any two temporal points in the converted graph is definitely shorter than their original time interval, which indicates that time series graphs are adept at capturing long-term dependencies because the information embedded in early prices can be obtained sooner by later points without long-range transfers and information vanishing.

VGs are connected graphs and the construction process is invariant even if a series of basic transformations is performed on the original series, such as vertical and horizontal translations [17]. An example of the VG algorithm can be seen in Fig. 2a, where we present a converted VG and the original time series with 20 data points. As seen, temporal points are connected by edges as long as they can see each other. Furthermore, we can measure the weight of every temporal point for chaotic properties settling after we obtain the graph structure within the time series. In traditional complex network analysis, many metrics are used to identify vital nodes, such as degrees, k-cores, and cluster coefficients. In this study, the CI algorithm is selected due to its low computational consumption yet excellent performance in characterizing the node influence with regard to the influence as a collective attribute, instead of a local feature such as the node's degree [22]. In the CI algorithm, a $Ball(i, l)$ contains several nodes within a ball of radius l (the length of the shortest path) around node v_i. The frontier of this ball is defined as $\partial Ball(i, l)$. Then, the CI index of node v_i of radius l is

$$CI_l(i) = (d_i - 1) \sum_{j \in \partial Ball(i,l)} (d_j - 1), \tag{2}$$

where d_i is the degree of node v_i and l is a nonnegative integer and given according to the graph diameter (normally less than the diameter) [1]. A time series graph with CI as the node weight converted from the time series in Fig. 2a is shown in Fig. 2b. In this figure, nodes with higher weights are marked with deeper colors and greater sizes.

In this phase, we use the vector P_t to denote the historic state of a stock at time t, where $P_t \in \mathbb{R}^{6\times T}$ and consists of the raw market price data (the closing price, high price, low price, opening price, amount and volume), and T is the length of the look-back window of t. Through the VG algorithm, converted price graphs $\mathbb{G}_t$ are obtained, where $\mathbb{G}_t = [G_t^C, G_t^O, G_t^H, G_t^l, G_t^A, G_t^V]$ and each graph has T nodes. Furthermore, before extracting structural information, we measure the node weights CI_t of the converted graphs through the CI algorithm, where $CI_t \in \mathbb{R}^{6\times T}$.

[1] In general, l was set to 3 and achieved sufficient performance in [22], while we assign 2 to l because our mapped graphs with 20 nodes are far smaller than the graphs in [22]

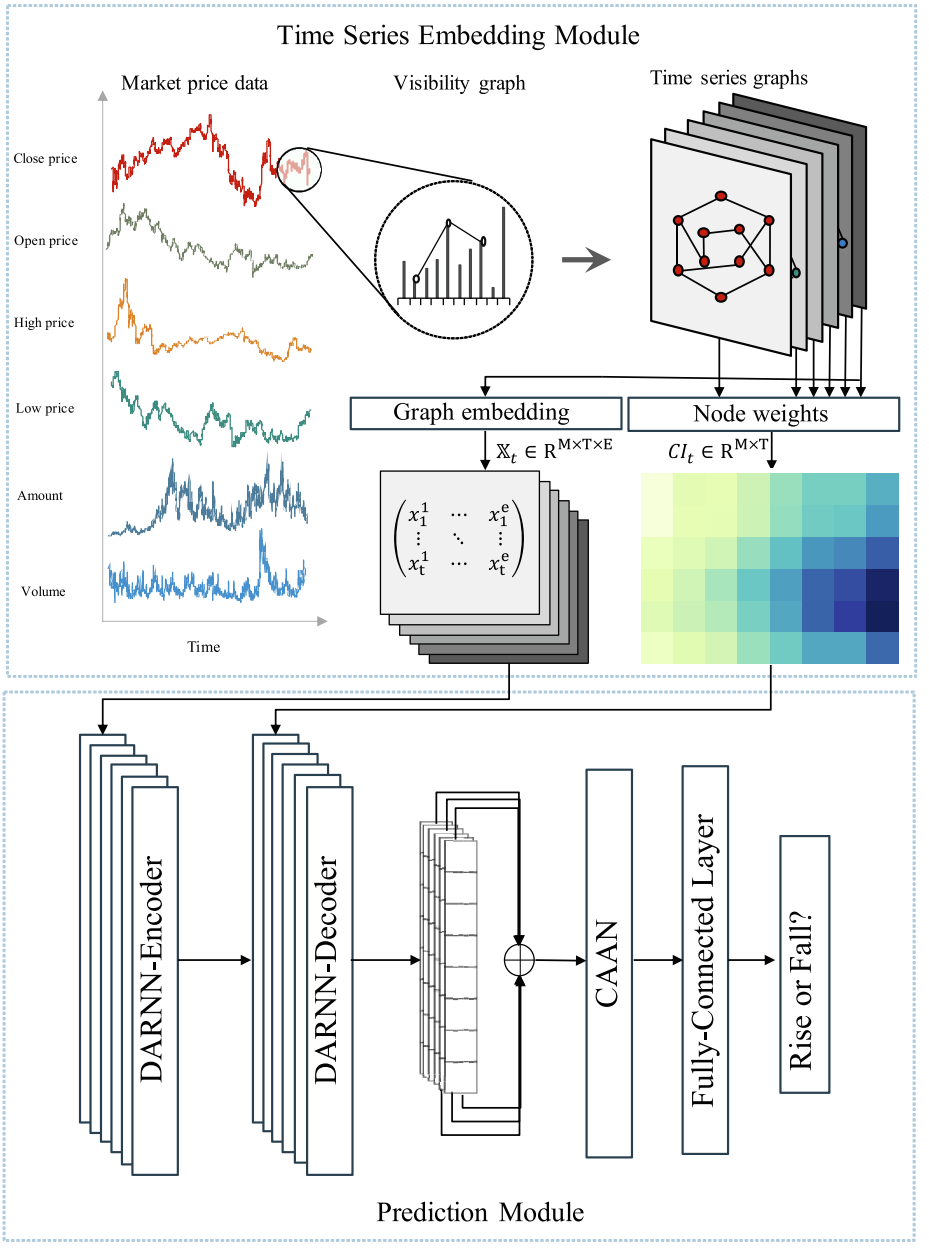
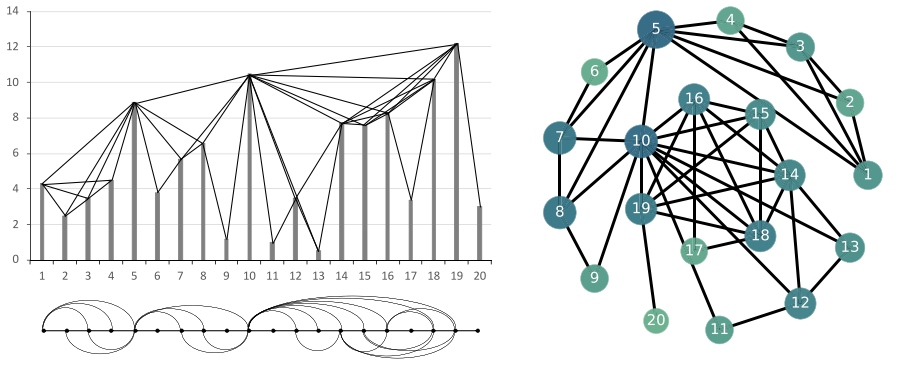


Fig. 1. Computational flow of the proposed framework. Let $P_t \in \mathbb{R}^{M\times T}$ denote the history price data for a stock at time t, the node vectors after embedding is $\mathbb{X}_t \in \mathbb{R}^{M\times T\times E}$ and node weights is $CI_t \in \mathbb{R}^{M\times T}$, where M is number of stock quote data (six here), T is the length of the lookback window and E is the embedding size of graph nodes.

3.1.2. Graph embedding

In this paper, graph embedding based on vector representations is further employed, as this not only complements the loss of mapped price graphs in the temporal sequence but also incorporates structural information into deep learning methods. To retain the structural properties of graphs regarding associations among their temporal points, we adopt struc2vec [28] in this paper. Struc2vec is a skip gram-based graph embedding algorithm that aims to learn a mapping $g : v \in V \mapsto \mathbb{R}^{|V|\times d}$. According to the structural similarities based on nodes' k-hop neighborhoods, struc2vec is able to learn a vector-based representations that capture the structural roles of the nodes. Specifically, the method involves executing numerous random walks over the graph from each node. The co-occurrences of nodes in a short window are captured based on the sequences of these walks, which can be used to tackle the diffusion in the neighborhood around every node in the graph and explore the local topology structure around a vertex. The embedding method is designed to learn a representation that enables the estimation of the possibility of a node u being shown together with other nodes in the subwindow of a short random walk:



(a) Visibility graph (b) Time series graph with node weights

Fig. 2. Instance of a time series containing 20 temporal points and the associated VG derived from the VG algorithm..

$$\max_{g} \sum_{u \in V} \log Pr(N(u)|g(u)), \tag{3}$$

where $N(u)$ represents the neighboring nodes of node u. The likelihood of a vertex v appearing together with u is estimated by using a softmax function:

$$Pr(v|u) = \frac{exp(g(v) \cdot g(u))}{\sum_{v_i \in V} exp(g(v_i) \cdot g(u))}. \tag{4}$$

Furthermore, to measure the structural similarities between nodes, struc2vec [28] generates a series of weighted assistant graphs $\boldsymbol{g}_k, k = \{1, 2, \ldots, k^*\}$ that are derived from the original graph, in which the assistant graphs capture the structural similarities between nodes' k-hop neighborhoods, and k^* is the diameter of original graph. More concretely, each assistant graph is a weighted undirected complete graph. In regard to the weights of edges, $R_k(v)$ is defined as the ordered degree sequence with nodes that are precisely k hops from v, and $w_k(v,u)$ that measures the edge weights in an assistant graph $\boldsymbol{g}_k$ is recursively defined as

$$w_k(v,u) = w_{k-1}(v,u) + d(R_k(v), R_k(u)), \tag{5}$$

where $w_0(v,u)$=0 and $d(R_k(v), R_k(u))$ measures the difference between $R_k(v)$ and $R_k(u)$. In addition, struc2vec can generate vertex sequences based on the edge weights in these weighted assistant graphs. Then, to maximize the probability of neighboring nodes in the local area appearing together with the central node, a neural network is trained by using the skip gram architecture. Finally, the output of the hidden layer of the trained neural network is obtained as the embedding of nodes. Through embedding price graphs into vector representations, struc2vec further enriches the information of time points by considering the global dependencies in representations and enhances the description of raw time series. The long-term dependencies among temporal points can be captured owing to the feature of struc2vec in structural similarity measurement.

In this phase, with the converted price graphs $\mathbb{G}_t$, we obtain the tensor representations $\mathbb{X}_t$ for all graphs through struc2vec at time t, where $\mathbb{X}_t = [X_t^C, X_t^O, X_t^H, X_t^L, X_t^A, X_t^V], X_t \in \mathbb{R}^{T \times E}$ and E is the embedding size.

3.2. Prediction module

In this section, we introduce the classification module, which is constructed with two main parts. The first key part has several (e.g., six here) DARNNs with node weights for the temporal attention network [27]. The goal of the first part is to automatically extract input series representations by incorporating temporal information and structural information [2]. Based on this model, a stock representation r_t is obtained from the combination of several DARNNs' outputs for each stock at time t. The second key part is a cross-asset attention network (CAAN), which is used to describe the interrelationships among the stocks [37]. Finally, the classification results are given by a fully connected classification layer. To further illustrate the detailed data transformation in the prediction module, Fig. 3 shows the input and output for each step [3].

[2] Temporal information is obtained through organizing the model inputs in the order of time sequences.
[3] We omit the presentation of the historical price input of DARNN-Decoder to emphasize the structural information input.

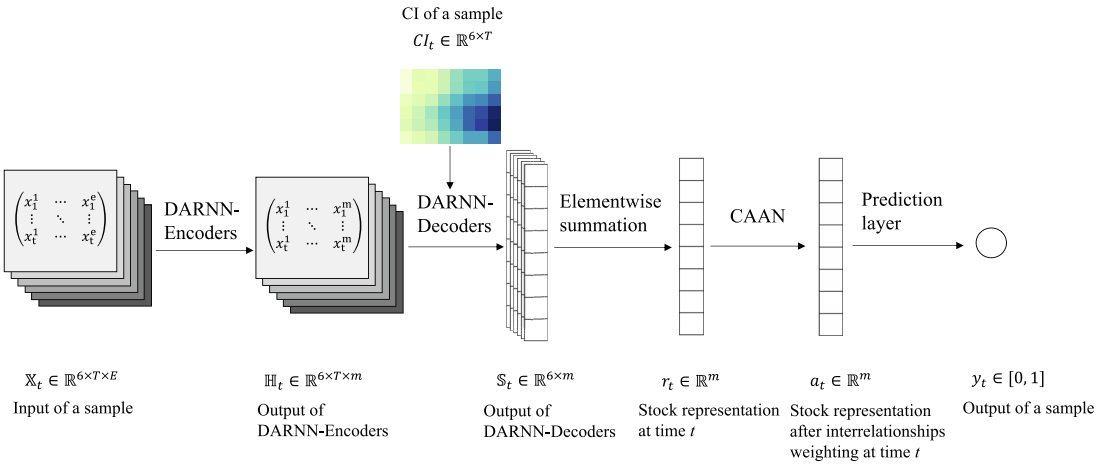


Fig. 3. Computational flow of the prediction module. Here, we take a sample at time t as an example, in which T is the lookback window of historical prices at time t. E is the embedding size of struc2vec for each graph node. In the experiments, we set hidden vectors of our deep learning method to a fixed size m.

3.2.1. Structural information learner

The structural information learner contains multiple DARNNs that incorporate node weights. DARNNs [27] are employed as our essential components not only because of their capability of selecting crucial variables and temporal points but also because of their excellent performance regarding time series forecasting in comparison to LSTM [14] and attention-based LSTM [1]. Without the loss of any generality, we show the learning process in one of the six DARNN models for illustration. Each DARNN is an encoder-decoder network and has the same learning process with different inputs. Specifically, the encoder is basically an LSTM that learns the input sequences as a hidden representation that considers input attention. For time series prediction, letting $X = (x_1, x_2, \cdots, x_T) \in \mathbb{R}^{T\times E}$ denote the input sequence, where $x_t \in \mathbb{R}^E$ and E is the graph embedding size; then, the encoder is used to recursively learn a hidden vector from X:

$$h_t = \text{LSTM}(h_{t-1}, x_t), t \in [1, T], \tag{6}$$

where $h_t \in \mathbb{R}^m$ is the hidden vector encoded by LSTM at time step t and m is the size of each hidden vector.

As attention mechanisms are widely employed in deep neural networks, a DARNN integrates the input attention into the encoder stage, which can automatically select the appropriate input series to give more emphasis to informative features obtained from chaotic series. Letting $x^k = (x_1^k, x_2^k, \cdots, x_T^k)^\top \in \mathbb{R}^T$ denote the k-th input series; then, the input attention is calculated through a deterministic attention model, which adopts the hidden state h_{t-1} and cell state s_{t-1} of the encoder LSTM unit with

$$c_t^k = v_c^\top \tanh(W_c[h_{t-1}; s_{t-1}] + U_c x^k) \tag{7}$$

and

$$\alpha_t^k = \frac{\exp(c_t^k)}{\sum_{i=1}^{n} \exp(c_t^i)}, \tag{8}$$

where $v_c \in \mathbb{R}^T, W_c \in \mathbb{R}^{T\times 2m}$ and $U_c \in \mathbb{R}^{T\times T}$ are learnable parameters. At time step t, the importance regarding the k-th input series is estimated by α_t^k. To normalize the attention weights, a softmax is applied to c_t^k. Thus, the updating of input series with attention weights can be formulated as

$$\tilde{x}_t = (\alpha_t^1 x_t^1, \alpha_t^2 x_t^2, \cdots, \alpha_t^n x_t^n). \tag{9}$$

Then the updating of the hidden state at time t is

$$h_t = \text{LSTM}(h_{t-1}, \tilde{x}_t). \tag{10}$$

By measuring the attention weight of each input series, the encoder can adaptively select crucial series rather than assuming all the input series are equal. Taking the obtained structural information $\mathbb{X}_t$ as input, hidden features $\mathbb{H}_t = [H_t^C, H_t^O, H_t^H, H_t^L, H_t^A, H_t^V] \in \mathbb{R}^{6\times T\times m}$ for six prices can be obtained from the DARNN-Encoders.

Together with the encoder that applies the input attention, a decoder that incorporates temporal attention is also proposed to automatically select appropriate hidden vectors of the encoder from among all time intervals, as these can implic-

itly and appropriately capture the long-term dependencies within a time series. More concretely, the temporal attention among encoder hidden states is also calculated through a deterministic attention model, which adopts the hidden state $h'_{t-1} \in \mathbb{R}^m$ and cell state $s'_{t-1} \in \mathbb{R}^m$ of the decoder LSTM unit with

$$d_t^i = v_d^\top \tanh(W_d[h'_{t-1}; s'_{t-1}] + U_d h_i), \ 1 \leqslant i \leqslant T \tag{11}$$

and

$$\beta_t^i = \frac{\exp(d_t^i)}{\sum_{j=1}^{T} \exp(d_t^j)}, \tag{12}$$

where $U_d \in \mathbb{R}^{m\times m}, W_d \in \mathbb{R}^{m\times 2m}$ and $v_d \in \mathbb{R}^m$ are parameters to be learned and m is the size of each hidden state. Similarly, β_t^i is a temporal attention weight that measures the importance regarding the i-th encoder hidden state, thereby rebalancing the information at each temporal point to capture long-term dependencies and filter out chaotic temporal points. A softmax is also applied to d_t^i to normalize the attention weights. Thus, with the obtained temporal attention, all encoder hidden states $\{h_1, h_2, \cdots, h_T\}$ are weightedly summed as a context vector e_t:

$$e_t = \sum_{i=1}^{T} \beta_t^i h_i. \tag{13}$$

The context vector e_t varies for each time step.

In this basic decoder, the temporal weights modeled by Eqs. 11 and 12 are directly learned from the hidden encoder representation. In fact, as we discussed in the previous section, we can use priori knowledge to enhance the temporal weight learning process. Given the previously mentioned node weights of the price graphs, we use the node weights as the knowledge-based attention over the temporal points to highlight informative points and address chaotic properties of price series. Then, for each hidden state h_i, we update the attention weight d_t^i as

$$\tilde{d}_t^i = v_d^\top \tanh(W_d[h'_{t-1}; s'_{t-1}] + U_d h_i + w_d CI_i), \ 1 \leqslant i \leqslant T, \tag{14}$$

where w_d is a learnable parameter and CI_i denotes the node weight of temporal point i. In this way, the relative importance levels of temporal points in the price graphs are introduced as weights to enhance attention learning and chaotic properties settling. The nodes that obtain critical positions in the time series contribute more to the final sample representations and make more accurate predictions. After reweighing the temporal attention, the combination of the updated context vector $\tilde{e}_t$ with the stock price series $\{p_1, p_2, \ldots, p_{T-1}\}$ is denoted as $\tilde{p}_{t-1} = w_p[p_{t-1}; \tilde{e}_{t-1}] + b_p$, where $[p_{t-1}; \tilde{e}_{t-1}] \in \mathbb{R}^{m+1}$ is a concatenation of the historical price and context vector, and $w_p \in \mathbb{R}^{m+1}$ *and* $b_p \in \mathbb{R}$ are parameters to learn. Accordingly, the newly $\tilde{p}_{t-1}$ is adopted to update the hidden feature of LSTM in decoder as $h'_t = \text{LSTM}(h'_{t-1}, \tilde{p}_{t-1})$. Finally, the output of the DARNN-Decoder can be formulated as

$$S = \tilde{w}[h'_T; \tilde{e}_T] + \tilde{b}, \tag{15}$$

where $S \in \mathbb{R}^m$ and $[h'_T; \tilde{e}_T] \in \mathbb{R}^{m+m}$ is a concatenation of the hidden feature h' and the updated context vector $\tilde{e}$ at the last step of LSTM. The parameters $\tilde{w} \in \mathbb{R}^{m\times 2m}$ and $\tilde{b} \in \mathbb{R}^m$ map the concatenation to the size of the decoder hidden states.

With the hidden features $\mathbb{H}_t$ learned from the DARNN-Encoder, the stock representations $\mathbb{S}_t = [S_t^C, S_t^O, S_t^H, S_t^L, S_t^A, S_t^V]$ for six price series at time t are computed through several DARNNs, where $S_t \in \mathbb{R}^m$ and m is the hidden size of each DARNN's decoder. Then, with an elementwise summation layer, six representations are merged into one $r_t = \sum_{l\in\{C,O,H,L,A,V\}} S_t^l \in \mathbb{R}^m$ as the stock representation at time t.

3.2.2. The CAAN model

While acquiring the stock representations, we adopt a CAAN based on self-attention to utilize the interrelationships among stocks. Specifically, based on the merged stock representation r_i [4] of stock i, we calculate three vectors $q^i = W_q r^i, k^i = W_k r^i$ and $v^i = W_v r^i$ as the query vector, key vector and value vector, respectively, where W_q, W_k, and W_v are learnable parameters. The interrelationships between stock i and other stocks within a batch are computed by using the query vector q^i of stock i to query the key vectors of other stocks, i.e.,

$$l^{ij} = \frac{q^{i\top} \cdot k^j}{\sqrt{D_k}} \tag{16}$$

and

[4] The omittance of time t does not cause the loss of any generality

$$\gamma^{ij} = \frac{\exp(l^{ij})}{\sum_{j'=1}^{I} \exp(l^{ij'})}, \tag{17}$$

where D_k is a rescaling parameter set in line with [36] and I is the size of each sample batch. Afterwards, we use the inter-relationships γ^{ij} to weighted the value vectors v_j into an attention representation

$$a^i = \sum_{j=1}^{I} \gamma^{ij} \cdot v^j \tag{18}$$

for stock i. Finally, to forecast the future price trend, the scalar prediction score $\hat{y}^i$ for stock i is computed by a feed-forward layer and a sigmoid transformation:

$$\hat{y}^i = \text{sigmoid}(W_{fc} a^i + b_{fc}) \tag{19}$$

where W_{fc} and b_{fc} are the linear parameter and the bias to be learned, respectively. Finally, based on the computation flow in Fig. 3, the learning and optimization process of our prediction module are summarized in Algorithm 1.

Algorithm 1: Optimization of our prediction module within a sample

Input: input sample $\mathbb{X} \in \mathbb{R}^{6\times T\times E}$, corresponding $CI \in \mathbb{R}^{6\times T}$, historical price series $P \in \mathbb{R}^{6\times T}$, labeled target y
Output: predicted label $\tilde{y}$ of input
1: Denote all parameters of the prediction model as $\mathbb{W}$;
2: Initialize the parameters $\mathbb{W}$;
3: **for** $epoch \in [1, \ldots, maxIteration]$ **do**
4: $\mathbb{H} = \text{DARNN} - \text{Encoders}(\mathbb{X})$, where $\mathbb{H} \in \mathbb{R}^{6\times T\times m}$;
5: $\mathbb{S} = \text{DARNN} - \text{Decoders}(\mathbb{H}, CI, P)$, where $\mathbb{S} \in \mathbb{R}^{6\times m}$;
6: $r = \sum_{i=0}^{6} \mathbb{S}_i$, where $r \in \mathbb{R}^m$;
7: $a = \text{CAAN}(r)$, where $a \in \mathbb{R}$;
8: $\tilde{y} = f_{prediction}(a)$, where $\tilde{y} \in \mathbb{R}$;
9: // computing loss on a batch sample;
10: $\mathscr{L} = \text{LossFunction}(y, \tilde{y})$;
11: Update hidden parameters $\mathbb{W}$ with gradient decent $(\mathscr{L}|\mathbb{W})$;
12: **end for**

4. Experiments

To evaluate the proposed framework, we use stock data from the China A-share market. In the following subsections, we introduce the details of the data and the compared methods. We also depict the detailed experimental settings of the training and testing process for our proposed framework and baselines.

4.1. Data

We collect the daily quote data of CSI-300 component stocks from the China A-share market from January 1, 2010, to December 31, 2019 to cover comprehensive patterns in price trends and avoid the external shock from the coronavirus disease 2–10 (COVID-19) on model validation. In particular, the China stock market has great research value, and considerable research attention has been devoted to it [44,3,16], and in which the CSI-300 index selects the most liquid A-share stocks and aims to reflect the overall performance of the China A-share market [21]. In addition to the explicit influences from the supply and demand of investors, the price of a stock can also be changed owing to some firm operations, including stock splits, rights offerings and dividend payouts or distributions, and the prices need to be adjusted after these actions. In this case, the adjusted price is more insightful for investigating historical returns because it reflects an accurate performance of a firm's market value beyond the raw market trading prices. The adjusted daily quote data are obtained from Tushare [5], an open source financial data package [40]. The data contain daily stock prices, including closing prices, low prices, high prices, opening prices, amounts and volumes.

[5] http://tushare.org.

4.2. Baselines

Due to the ubiquity of deep neural networks, most recent efforts regarding stock prediction have been devoted to leveraging LSTM [23], CNNs [21,13] and attention-based approaches [27,13] with market price data as inputs. Competitive performance is also achieved by these novel methods. Note that among these state-of-the-art prediction models, to the best of our knowledge, none of them leverage the structural information extracted from time series graphs. To inspect the power of our framework, which investigates such important information, our proposed framework is compared to the following models:

- LSTM: a basic LSTM network used to predict the future trends of stock prices based on historical price data [23].
- DARNN: a dual-stage attention-based RNN that employs input attention and temporal attention in the encoder and decoder stages, respectively [27].
- DARNN-SA: an extension of the DARNN that employs a self-attention layer between the output of the DARNN and the prediction layer.
- MFNN: a multifilter deep learning model that integrates convolutional and recurrent neurons for feature extraction with respect to financial time series and stock prediction [21].
- CA-SFCN: a fully convolutional network (FCN) incorporating cross attention (CA), in which CA is also used as dual-stage attention for the variable and temporal dimensions (with temporal attention first) [13].

In particular, in line with their original model inputs, all five baseline methods take market price data as inputs. In addition, we further make the comparison between our proposed framework with the classical time series approaches, i.e., ARMA and GARCH models.

4.3. Parameter settings

To prevent data snooping, experimental data sets are strictly split according to the sample dates. For example, we use the data from Jan. 2010 to Dec. 2018 as the training and validation sets and the rest as the test set, which includes the whole year of 2019 [6]. During the training process, 70% of the samples are randomly selected as the training set and the remaining 30% of the samples as the validation set to take full advantage of the historical information. For each stock, we take the historical information with a window size of 20 days [27,11] to predict the price trend of the next day. The target of our experiments is defined as follows:

$$y = \begin{cases} 1\ (\uparrow), & p^c_{t+1} > p^c_t \\ 0\ (\downarrow), & \text{otherwise}, \end{cases} \tag{20}$$

where p^c_t is the stock closing price at time t. Furthermore, in the experiment, the test period is set to a three-month sliding window; thus, 4 test periods are obtained from the test set in 2019 and sequentially denoted as 2019(S1), 2019(S2), 2019(S3) and 2019(S4). The details of our datasets are listed in Table 1. In summary, our training/validation set has more than a half million samples and is almost balanced. In addition, we have 72,545 samples in total for the test set, and each test period has more than 10,000 samples. We evaluate the prediction performance with the accuracy metric and employ binary cross entropy [7] to measure the loss between $\hat{y}$ and y.

In this paper, we select the optimal hyperparameters with grid search to obtain the best performance for all models. Specifically, we tune the sizes of the hidden representations within {32, 64, 128, 256} and the sizes of the minibatches within {32, 128, 256}. We use the adaptive moment estimation (Adam) optimizer with an initial learning rate of 1e-3. Following the computation flow in Algorithm 1 and Fig. 3, we employ the most popular deep learning library in the research area, PyTorch which can automatically optimize learnable parameters through backpropagation, to implement our prediction module [7]. For all baselines, we train the models in an end-to-end manner from raw quote data with a *z-score* normalization function using the standard deviation and mean calculated for each sample. For the parameters in struc2vec, we set a walk-length of 10 and a window-size of 3, which are consistent with the parameters set in [28]. The classical time series approaches, i.e., ARMA and GARCH, are optimized through *Auto-Arima* in Python.

5. Results

In this section, we show the empirical results obtained from numerical experiments. Further back-test experiments are also conducted to demonstrate the effectiveness of our proposed framework.

[6] We also test our framework on CSI-300 across 2015 to further inspect its capability under an extreme market environment, and stable and competitive performances can still be obtained by our proposed framework.

[7] The code of our framework is available at https://github.com/BUAA-WJR/PriceGraph

Table 1
Summary statistics of CSI-300 for the training, validation and test datasets. The dataset for training and validation has more than a half million samples and is almost balanced. The four seasons for the test set have 72,545 samples in total, and each season has more than 10,000 samples.

Dataset		#Sample	↓		↑	
			#Sample	(%)	#Sample	(%)
Train/Val	2010-2018	530,284	276,533	52.15	253,751	47.85
Test	2019(S1)	16,992	7,591	44.67	9,401	55.33
	2019(S2)	17,765	9,534	55.36	8,231	46.33
	2019(S3)	19,488	10,789	55.36	8,699	44.64
	2019(S4)	18,300	9,142	49.96	9,158	50.04

5.1. CI distribution of price graphs

To examine the quality of the price graphs obtained from the daily quote data of stocks, we take the CI distributions to assess whether the converted price graphs can capture the intrinsic properties of stocks and whether CIs are capable of distinguishing between graph nodes. Specifically, for each market price dataset, we split the converted price graphs into two groups by the corresponding target value and calculate the CI for each graph from 2010 to 2019. For graphs with target 1, we denote this group as *Rise* and color them red. For graphs with target 0, we denote this group with *Fall* and mark them green. Fig. 4 shows the CI distributions of a stock sampled from CSI-300 component stocks with the ticker symbol 002624. We find that the CI distributions of the two groups across the six market price datasets can be distinguished from each other. Furthermore, in addition to the visual differences, we also employ the Kolmogorov–Smirnov test to examine whether the two groups of distributions are significantly distinct in terms of statistics. We can see in Fig. 4 that all p-values are less than 0.001; that is, the two groups of CIs have different distributions, which indicates that the price graphs converted from stock quote data can leverage certain intrinsic properties of stocks and separate rises from falls. In addition to the CI distributions within two groups, we also inspect the discriminative ability of other structural indicators of nodes, such as degrees. However, they are not as distinguishable as CI within two groups, which indicates that CI can not only achieve more accurate or more discriminative attention weighing in model learning but also help the model control the possible biases when facing abnormal nodes caused by chaotic properties of financial time series. At the same time, this is consistent with the excellent performance of CI in the influence description.

5.2. Training set performance

We further assess the representational power of our proposed framework by comparing its training accuracies with that of the baselines. Models with higher representational power should have higher training set accuracy. Fig. 5 shows the training accuracy curves of our proposed framework and all five baselines on the same training set from 2010 to 2018 [8]. First, our proposed framework is capable of obtaining the highest training accuracy, which implies that certain properties of stock quote data can be captured by the structural information derived from price graphs. In comparison, despite utilizing the same learning component, DARNN-SA still cannot defeat our proposed framework. DARNN-SA yields a significant accuracy gain over the DARNN when fitting training data, which confirms the positive effect of the interrelationships between stocks [37]. Compared with our proposed framework, CA-SFCN achieves competitive performance on the training dataset with only six market price datasets, which confirms the superior capability of component-based CNNs for capturing short-range dependencies [13]. In particular, LSTM severely underfits the training dataset. In our experiments, in addition to feature engineering approaches, such as the CNNs in the MFNN and CA-SFCN, attention mechanisms also have remarkable predictive capabilities.

5.3. Test set performance

Next, we compare the achieved test accuracies to further evaluate our proposed framework. Although the training results do not directly reveal the generalization capability of structural information, it is not farfetched to expect that our proposed framework with strong representational power can accurately capture certain properties and thus generalize well. Table 2 compares the test accuracies of our proposed framework and the state-of-the-art baselines. The findings indicate that the performance on the test set is in line with that on the training set; that is, our proposed framework consistently outperforms the state-of-the-art baselines on the test set, obtaining the best accuracy. This reveals that the structural information contained in stock time series is capable of addressing fundamental questions regarding long-term dependencies and chaotic properties. Compared with that of the second-best method, CA-SFCN, the recall rate of our proposed framework is lower in the second season of 2019, and the precision is lower in the third season of 2019. In regard to the classical time series approaches, ARMA and GARCH do not even achieve competitive performance with the base deep learning method (i.e., LSTM). Although high precision and recall are preferred in the Chinese stock market, which allows longing only on stocks,

[8] Although we train all models with longer epochs, Fig. 5 shows only the accuracies for the first 500 epochs because all models converge by this point.

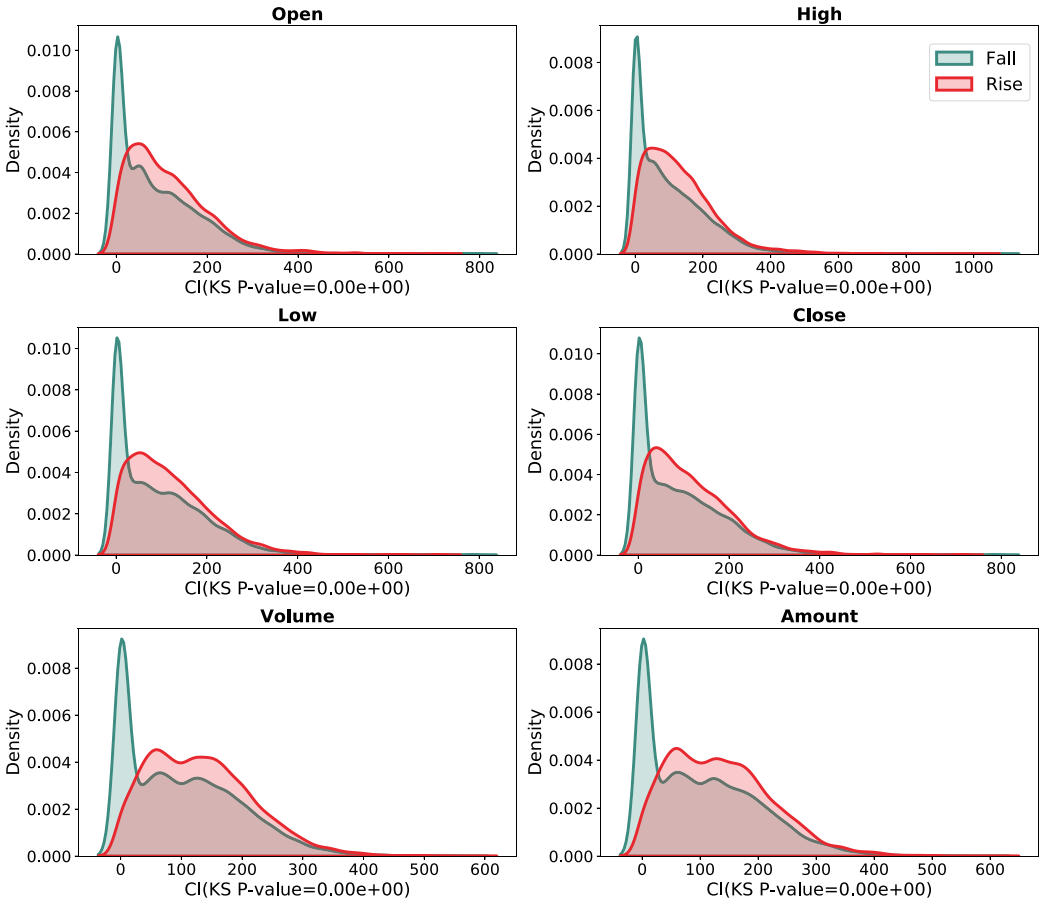


Fig. 4. CI distributions of price graphs converted from a sample stock with six market price datasets from 2010 to 2019. We also examine the two groups of distributions in the *Rise* and *Fall* directions by a Kolmogorov–Smirnov test (KS test), where the KS test p-values show that the two groups of distributions are distinct (i.e., all p-values are less than 0.001).

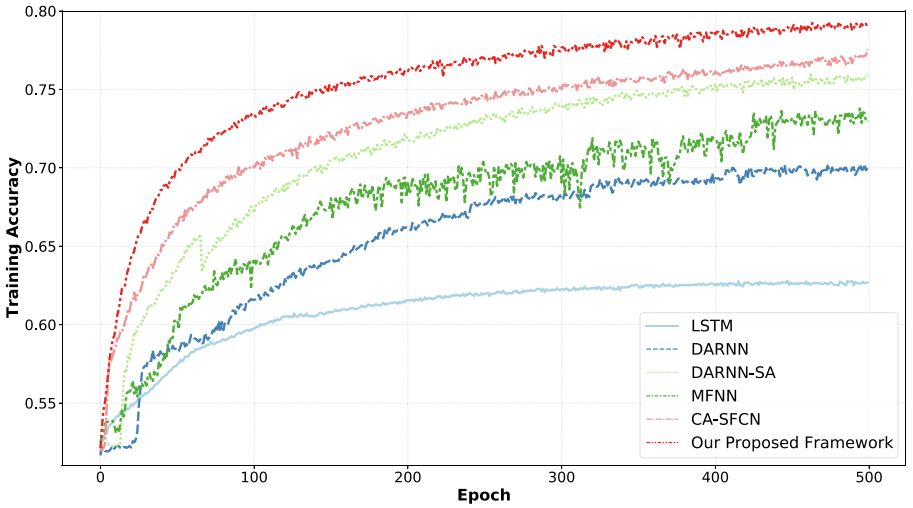


Fig. 5. Training set performances of our proposed framework and baselines..

Table 2
Results (%) of our proposed framework and the baselines. All models predict price trend labels at the next time step. The best-performing results are highlighted with boldface. Our proposed framework outperforms all the state-of-the-art baselines on the test accuracies.

	2019(S1)				2019(S2)				2019(S3)				2019(S4)			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
ARMA	50.15	54.96	42.81	48.13	50.75	46.61	42.51	44.46	49.89	44.26	39.91	41.97	50.07	49.40	41.77	45.26
GARCH	50.28	54.90	44.68	49.26	50.66	46.74	45.74	46.23	50.43	45.05	41.76	43.34	50.36	49.75	42.24	45.69
LSTM	57.94	64.88	52.26	57.89	59.88	58.98	44.04	50.43	56.71	52.23	35.33	42.16	54.75	55.97	44.92	49.84
DARNN	60.87	68.27	54.69	60.73	62.03	61.48	48.29	54.09	60.62	58.10	61.33	59.67	61.54	68.34	63.31	65.73
DARNN-SA	64.32	71.72	58.63	64.52	66.23	66.60	54.40	59.89	65.47	65.00	59.09	61.9	65.63	72.34	68.92	70.59
MFNN	61.21	68.28	60.81	64.33	63.00	65.30	51.40	57.52	62.74	67.68	58.75	62.9	64.69	67.19	57.52	61.98
CA-SFCN	65.51	72.82	60.10	65.85	67.21	67.61	**73.52**	70.44	66.10	**77.81**	68.32	**72.76**	67.30	70.24	73.38	71.78
Our framework	**67.48**	**75.24**	**61.45**	**67.65**	**68.46**	**69.81**	71.67	**70.73**	**68.34**	67.86	**73.77**	68.09	**67.91**	**77.51**	**73.78**	**75.60**

Notes. Precision, recall and the F1 measure are metrics calculated in the upward direction.

consistently high accuracies are still able to ensure that profitable predictions are generated by our framework. Furthermore, it is worth mentioning that our proposed framework yields significant improvements in all evaluation metrics over those of DARNN-SA with six market price datasets as inputs; this again emphasizes the effectiveness of the structural information obtained from price graphs and the proposed framework for stock prediction.

6. Discussion

To further assess the performance of the proposed framework, the predictive powers of structural information and the node weights of price graphs are tested. Furthermore, a trading strategy is also implemented based on the signals produced by the prediction model to validate the profitability and stability of our framework in more realistic scenarios.

6.1. Ablation analysis

To test the effectiveness of structural information, the contribution of each type of market price data is inspected by reducing the complexity while preserving the predictive capability of our framework. Borrowing the idea of feature selection from traditional machine learning methods [12], the training and computational time can be saved by reducing the size of the input tensor for deep learning approaches. The input of the proposed framework corresponds to the number of market price data types. Therefore, we conduct an ablation study with only one price graph embedding (PGE) derived from each corresponding time series of price data as input; these embeddings are denoted as Close-PGE, Open-PGE, High-PGE, Low-PGE, Amount-PGE and Volume-PGE. To simply exhibit the effectiveness of structural information, accuracy, the most universal indicator in prediction tasks, is adopted to inspect the performance of each type of market price data [40]. The accuracies are shown in the upper panel of Table 3. As can be seen, the performances of the models tested on of stock price embeddings, i.e., close, open, high and low prices, are consistently better than those of the models tested on amount and volume embeddings. This proves that price trend prediction can be better performed by models with more direct inputs of price. In particular, the model of Close-PGE acquires the highest accuracies in this ablation analysis except for the second season of 2019. These results also indicate that the structural information obtained from different types of market price data related to stock prices contains distinct predictive power for predicting price trends. The results from our proposed framework represent an increase in predictive power through the extraction of structural information from a variety of market price data types.

Next, we delve deeper into the effect of node weights obtained from price graphs in terms of mitigating chaotic properties of financial time series. Specifically, we conduct a battery of additional experiments based on the DARNN and DARNN-SA. As shown in Eq. 14, we update the temporal attention by adding node weights as knowledge-based attention over temporal points. Therefore, we add the same node weights to the second decoder stages of DARNN and DARNN-SA while retaining the original model input and six market price datasets. As shown in the lower panel of Table 3, the two variants are denoted as DARNN-NW and DARNN-SA-NW, respectively. Compared to the original versions without node weights (see Table 2), the adjusted models yield obvious increases in accuracies for the four seasons. Thus, we can conclude that by using node weights to enhance temporal weight learning, we can refine crucial information from chaotic series and obtain better or at least comparable prediction results.

6.2. Why CI works

To obtain further insights into how CI overcomes chaotic properties induced bias, a representative example of how the model decoder shifts its temporal attention from noisy points to real influential points is shown in Fig. 6. Given the chaotic properties of financial series, stock prices are shown in non-stationary time series and have abrupt changes or unexpected reversals. Fig. 6a shows such a typical price series, in which an abrupt price change or unexpected reversal appears at Day 15 and results in the largest price deviation from its five-day moving average. In addition, as can be seen in Fig. 6b, the Day 15 in the corresponding price graph also serves as the node of the largest degree. However, in Fig. 6b, the real informative (larger CI values) nodes are marked with deeper colors and greater sizes. Thus, we know that the Day 18 is actually the node with the greatest collective influence in this price graph followed by Day 8 and Day 7, suggesting the indispensable signals that they carry in the trending prediction of other nodes. In the process of temporal attention learning in Eq. 11, we expect that the model could pay more attention to the real informative nodes that carry significant trending signals instead of abrupt changes. However, as can be seen in Fig. 6c, the model without CI regularization just follows the price change and is fully attracted by Day 15 rather than Day 18, implying that the nodes with the largest degrees resulted by abrupt changes in price would be incorrectly targeted as informative nodes in the graph regardless of the scant trending information they actually carry. To fix this disadvantage of the vanilla model, we employ the CI of nodes to enhance the temporal weight learning process as in Eq. 14. As a result, Fig. 6d shows that the learned model agrees to dispense most temporal weights to Day 18 and also pays more attention to Day 8 and Day 7 rather than Day 15, which is indeed less informative for trending prediction.

Table 3
Ablation analysis accuracy of each learning model (%). The increases in accuracy after introducing node weights are presented in parentheses behind accuracies.

	The effectiveness of structural information															
	2019(S1)				2019(S2)				2019(S3)				2019(S4)			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Close-PGE	61.24	67.32	57.47	62.01	63.69	64.57	66.31	65.43	62.23	61.74	64.54	63.11	60.54	66.82	67.15	66.98
Open-PGE	60.34	65.43	55.97	60.33	63.17	63.73	59.42	61.50	61.97	61.41	63.27	62.33	59.33	64.33	63.54	63.93
High-PGE	60.89	66.47	56.41	61.03	64.13	65.54	62.76	64.12	60.36	59.58	55.76	57.61	60.12	65.97	64.27	65.11
Low-PGE	61.59	66.12	57.57	61.55	62.69	62.91	65.34	64.10	60.83	60.09	64.17	62.06	59.64	62.13	66.56	64.27
Amount-PGE	58.65	63.44	52.15	57.24	60.27	60.22	63.96	62.03	59.41	58.32	61.91	60.06	58.47	63.14	64.85	63.98
Volume-PGE	59.15	63.79	52.86	57.81	60.52	60.25	62.77	61.48	58.37	59.13	57.35	58.23	59.37	62.57	54.13	58.04
	The effectiveness of node weights															
	2019(S1)				2019(S2)				2019(S3)				2019(S4)			
	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
DARNN-NW	61.33 (**+ 0.46**)	68.76	57.33	62.53	62.94 (**+ 0.91**)	63.42	50.74	56.38	61.89 (**+ 1.27**)	58.11	63.86	60.85	62.13 (**+ 0.59**)	68.48	63.53	65.91
DARNN-SA-NW	65.67 (**+ 1.35**)	73.43	57.49	64.49	67.22 (**+ 0.99**)	67.06	55.2	60.55	65.64 (**+ 0.17**)	64.47	62.36	63.40	66.72 (**+ 1.09**)	70.36	69.87	70.11

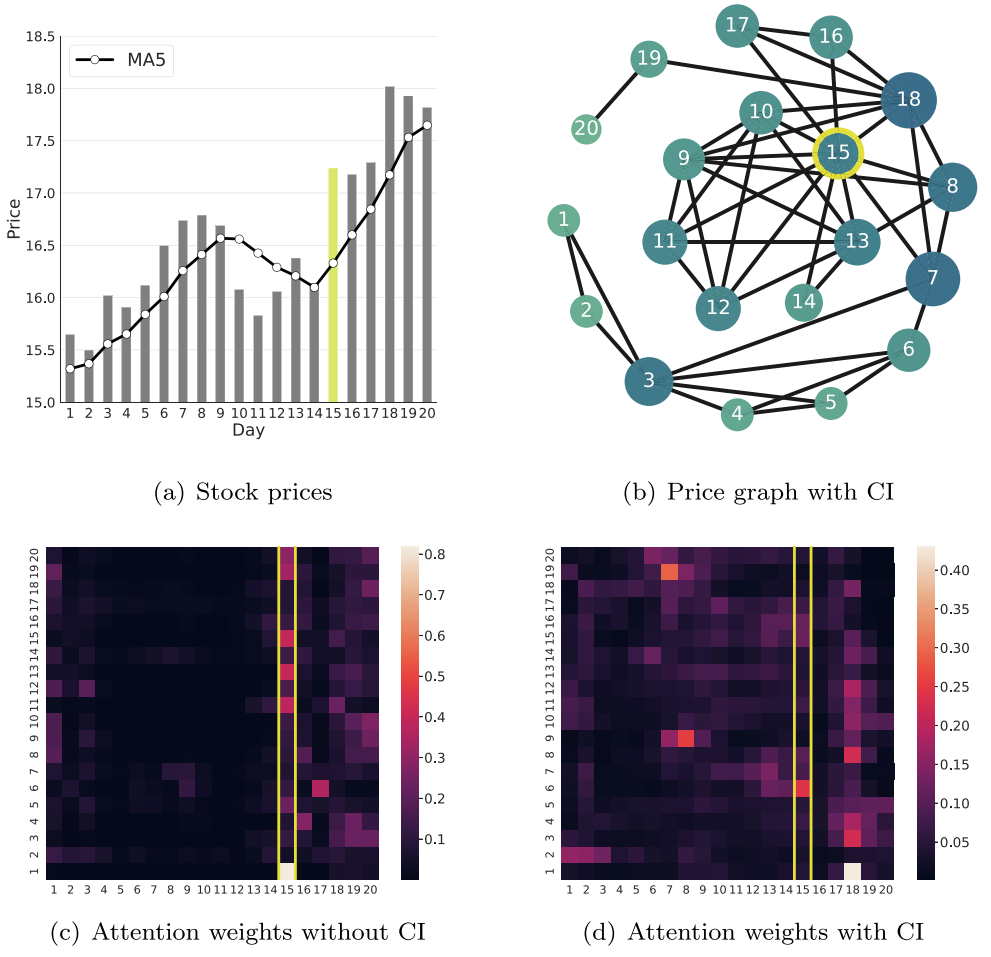


Fig. 6. Illustration of how the CI algorithm can help the model capture real informative nodes. Day 15 presents the largest price deviation from its five-day moving average (MA5), occupies the node with largest degree in the corresponding price graph and attracts most of the temporal attention of the model without CI regularization (Eq. 11). After updating the attention with CI as in Eq. 14, the learned model dispenses most temporal weights to Day 18 and also pays more attention to Day 8 and Day 7 instead of Day 15.

6.3. Trading simulation

To further check the performance of our proposed framework, multiple back-tests are conducted by simulated trading of chosen stocks in our test set for the four seasons of 2019. The estimation strategy carries out trading with a daily frequency. In the process of simulation, according to the prediction of our framework, a simple rule is set to develop the trading strategy: if a rising trend of a stock price is given by our framework, we will take a long position on that stock; while if a falling trend of a stock price is predicted, we will take a short position for that stock. All stocks are evenly invested in and held for one day. In particular, all short/long operations open the position at the closing price of the prediction day and close the position at the closing price of the next day. Under the circumstance of no transaction costs, the cumulative profits are reinvested on the next trading day. Although the transaction costs affect the final profit, the relative position based on model profit does not change owing to the use of the same trading strategy. We also calculate the average returns of the component stocks of CSI-300 by holding every stock evenly as the baseline, thereby indicating the overall market trend. All models' net value curves in the periods of simulations are shown in are shown in Fig. 7. Among all the baseline methods, our proposed framework gains the best profits (47.91% return on average), even during the second season of 2019 when the market falls into a downturn. In particular, the market had a long rising period during the first season of 2019, and all of the models failed to gain enough profit to defeat the market, except for our framework, CA-SFCN and DARNN-SA.

In summary, all of these results indicate that the structural information derived from price graphs by referring to the associations among temporal points and node weights can address the fundamental questions regarding long-term dependencies and chaotic properties of time series; moreover, the learned prediction model can help investors make more accurate trading decisions.

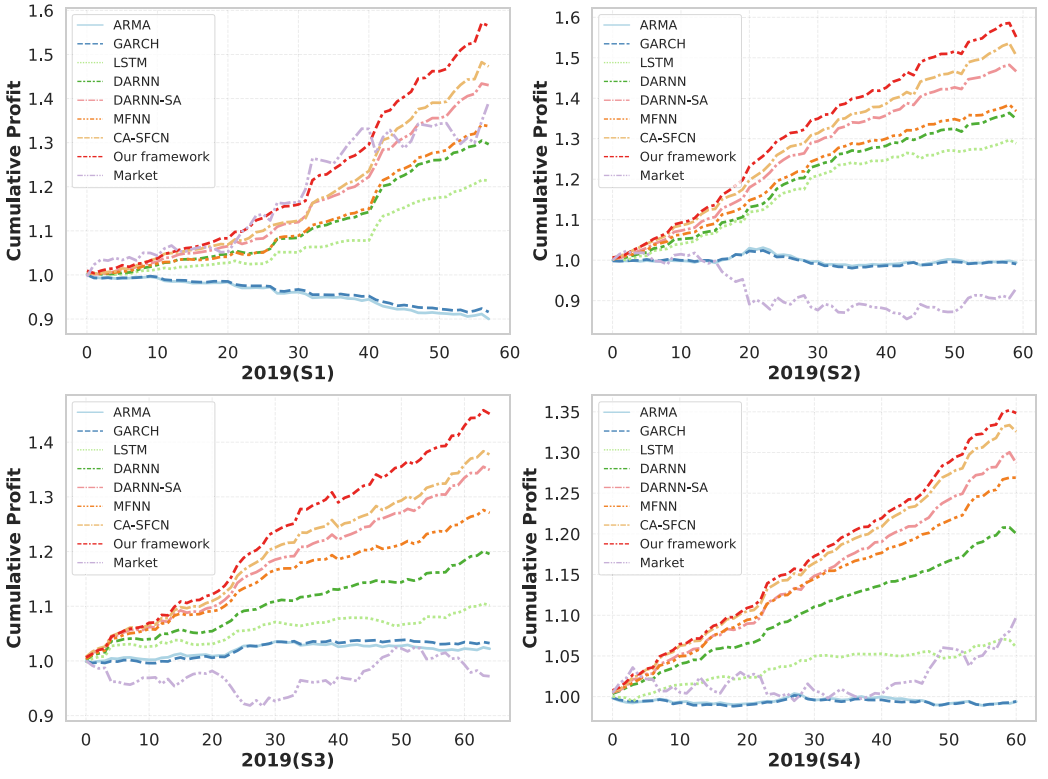


Fig. 7. The cumulative profit curves of our proposed framework and different baselines for the four seasons of 2019..

7. Conclusion

In this paper, based on time series graphs converted from market price data, we propose a novel framework to address fundamental questions by using structural information extracted from price graphs. Through this framework, deep learning models collaborating with structural information achieve competent performance and display practical capabilities in stock prediction and trading. By employing models with attention mechanisms, we find that the long-term dependencies of the values in time series can be captured via structural information. Furthermore, by identifying prominent content from chaotic time series, models employing graph node weights as additional knowledge for temporal attention are capable of tackling chaotic properties of financial time series and achieving better stock prediction performance. The results in this paper highlight the role of complex network methods for characterizing dynamical systems derived from time series. Compared with raw financial information such as market price data, structural information finely weaved from financial time series is verified to be superior for prediction tasks.

The superior performance of our proposed framework supplements the existing financial research on stock prediction by enriching the representations of time series through complex networks. Compared with previous studies that pay attention to networks among different stocks, sectors or markets [44,31,3,16,32], the time series graphs based on temporal points from stock prices have more fine-grained information about the current states of stocks, which sufficiently helps us address the two fundamental questions regarding long-term dependencies and chaotic properties in daily stock prediction. Notably, our framework is not limited to these observations mentioned in this study (i.e., market price data), other series related to price changes with time could also be new inputs to our framework. Accordingly, in addition to stock price prediction, our framework can also be extended and applied to other financial scenarios.

Our results also offer noteworthy implications for investors and policy makers. In this study, not only the transformation of financial series based on the VG algorithm but also the proposed deep learning model for financial series prediction have important inspirations for investors and supervisors in practice. Firstly, based on the structural information obtained from stock prices, our proposed framework outperforms the state-of-the-art models that take the raw market prices as input in testing accuracies. Additionally, the highest cumulative net values in the trading simulation further highlight the effect of our framework in profit promotion. In this context, our work has important value in decision-making support for investors, especially institutional investors. Secondly, our work is also helpful for supervision. The superior performance of the obtained results shows that the structural information of time series can be excellently preserved when performing complex

network transformation, which makes it possible to depict the entire market structure, and provides a new angle for exploring propagation dynamics and early warnings of systemic risks.

Although the effectiveness of structural information for stock prediction is verified, there are also limitations in this study that inform the directions of future research. For example, in addition to graph embeddings, various characteristics of graphs that have been investigated in recent decades, such as graph centrality, clustering coefficients, and global efficiency, could also be informative for graphs and effective for stock prediction. Moreover, feature engineering approaches, such as taking advantage of time series graphs, may lead to potential information loss during the process of structural information extraction. Therefore, given the ubiquity of graph neural networks in various time series problems, such as event forecasting, transportation prediction and recommendation, an end-to-end learning model based on a graph neural network may be superior for financial time series prediction. Both of the above limitations are promising directions for our future work.

CRediT authorship contribution statement

Junran Wu: Data curation, Investi n, Writing - original draft. **Ke X** pervision, Writing - review & editing. **Xueyuan Chen:** Visualization, Investigation. **Shangzhe Li:** Validation, Data curation. **Jichang Zhao:** Conceptualization, Methodology, Writing - review & editing.

Declaration of Competing Interest

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

Acknowledgments

This work was supported by NSFC (Grant No. 71871006).

References

[1] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: Proceedings of the 3rd ICLR, 2015.
[2] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Transactions on Neural Networks 5 (2) (1994) 157–166.
[3] E. Bouri, R. Gupta, S. Hosseini, C.K.M. Lau, Does global fear predict fear in brics stock markets? evidence from a bayesian graphical structural var model, Emerging Markets Review 34 (2018) 124–142.
[4] Cao H., Li Y.. Unraveling chaotic attractors by complex networks and measurements of stock market complexity. Chaos: An Interdisciplinary Journal of Nonlinear Science 2014;24(1):013134..
[5] W. Chen, M. Jiang, W.G. Zhang, Z. Chen, A novel graph convolutional feature based convolutional neural network for stock trend prediction, Information Sciences 556 (2021) 67–94.
[6] G. Dai, X. Hu, Y. Ge, Z. Ning, Y. Liu, Attention based simplified deep residual network for citywide crowd flows prediction, Frontiers of Computer Science 15 (2) (2021) 1–12.
[7] P.T. De Boer, D.P. Kroese, S. Mannor, R.Y. Rubinstein, A tutorial on the cross-entropy method, Annals of Operations Research 134 (1) (2005) 19–67.
[8] R.V. Donner, Y. Zou, J.F. Donges, N. Marwan, J. Kurths, Recurrence networks-a novel paradigm for nonlinear time series analysis, New Journal of Physics 12 (3) (2010) 033025.
[9] J.D. Farmer, J.J. Sidorowich, Predicting chaotic time series, Physical Review Letters 59 (8) (1987) 845.
[10] H.I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, P.A. Muller, Deep learning for time series classification: a review, Data Mining and Knowledge Discovery 33 (4) (2019) 917–963.
[11] F. Feng, H. Chen, X. He, J. Ding, M. Sun, T.S. Chua, Enhancing stock movement prediction with adversarial training, in: Proceedings of the 28th IJCAI, 2019.
[12] Guyon I., Elisseeff A.. An introduction to variable and feature selection. Journal of Machine Learning Research 2003;3(Mar):1157–1182..
[13] Y. Hao, H. Cao, A new attention mechanism to classify multivariate time series, in: Proceedings of the 29th IJCAI, 2020.
[14] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation 9 (8) (1997) 1735–1780.
[15] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (5) (1989) 359–366.
[16] Q. Ji, E. Bouri, R. Gupta, D. Roubaud, Network causality structures among bitcoin and other financial assets: A directed acyclic graph approach, The Quarterly Review of Economics and Finance 70 (2018) 203–213.
[17] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, J.C. Nuno, From time series to complex networks: The visibility graph, Proceedings of the National Academy of Sciences 105 (13) (2008) 4972–4975.
[18] Z. Li, D. Yang, L. Zhao, J. Bian, T. Qin, T.Y. Liu, Individualized indicator for all: Stock-wise technical indicator optimization with stock embedding, in: Proceedings of the 25th SIGKDD, 2019, pp. 894–902.
[19] H. Lin, G. Liu, F. Li, Y. Zuo, Where to go? predicting next location in iot environment, Frontiers of Computer Science 15 (1) (2021) 1–13.
[20] A.W. Lo, Long-term memory in stock market prices, Econometrica: Journal of the Econometric Society (1991) 1279–1313.
[21] W. Long, Z. Lu, L. Cui, Deep learning-based feature engineering for stock price movement prediction, Knowledge-Based Systems 164 (2019) 163–173.
[22] F. Morone, H.A. Makse, Influence maximization in complex networks through optimal percolation, Nature 524 (7563) (2015) 65–68.
[23] D.M. Nelson, A.C. Pereira, R.A. de Oliveira, Stock market's price movement prediction with lstm neural networks, in: IJCNN. IEEE, 2017, pp. 1419–1426.
[24] G. Nicolis, A.G. Cantu, C. Nicolis, Dynamical aspects of interaction networks, International Journal of Bifurcation and Chaos 15 (11) (2005) 3467–3480.
[25] M.H. Olyaee, A. Yaghoubi, M. Yaghoobi, Predicting protein structural classes based on complex networks and recurrence analysis, Journal of Theoretical Biology 404 (2016) 375–382.
[26] H. Pei, B. Wei, K.C.C. Chang, Y. Lei, B. Yang, Geom-gcn,, Geometric graph convolutional networks, in: Proceedings of the 8th ICLR, 2020.
[27] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, G.W. Cottrell, A dual-stage attention-based recurrent neural network for time series prediction, in: Proceedings of the 26th IJCAI, 2017, pp. 2627–2633.
[28] L.F. Ribeiro, P.H. Saverese, D.R. Figueiredo, struc2vec: Learning node representations from structural identity, in: Proceedings of the 23rd SIGKDD, 2017, pp. 385–394.
[29] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (6088) (1986) 533–536.

[30] X. Sang, Y. Zhou, X. Yu, An uncertain possibility-probability information fusion method under interval type-2 fuzzy environment and its application in stock selection, Information Sciences 504 (2019) 546–560.
[31] Shahzad S.J.H., Bouri E., Ahmad T., Naeem M.A. Extreme tail network analysis of cryptocurrencies and trading strategies. Finance Research Letters 2021a;:102106..
[32] S.J.H. Shahzad, E. Bouri, L. Kristoufek, T. Saeed, Impact of the covid-19 outbreak on the us equity sectors: Evidence from quantile return spillovers, Financial Innovation 7 (1) (2021) 1–23.
[33] H.H. Song, T.W. Cho, V. Dave, Y. Zhang, L. Qiu, Scalable proximity estimation and link prediction in online social networks, in: Proceedings of the 9th SIGCOMM, 2009, pp. 322–335.
[34] S.J. Taylor, Modelling financial time series, world scientific (2008).
[35] L.G. Valiant, A theory of the learnable, Communications of the ACM 27 (11) (1984) 1134–1142.
[36] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser L., Polosukhin I.. Attention is all you need. In: NIPS. 2017...
[37] J. Wang, Y. Zhang, K. Tang, J. Wu, Z. Xiong, Alphastock, A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks, in: Proceedings of the 25th SIGKDD, 2019, pp. 1900–1908.
[38] N. Wang, D. Li, Q. Wang, Visibility graph analysis on quarterly macroeconomic series of china based on complex network theory, Physica A: Statistical Mechanics and its Applications 391 (24) (2012) 6543–6555.
[39] Y. Wang, L. Wang, F. Yang, W. Di, Q. Chang, Advantages of direct input-to-output connections in neural networks: The elman network for stock index forecasting, Information Sciences 547 (2021) 1066–1079.
[40] J. Wu, K. Xu, J. Zhao, Predicting long-term returns of individual stocks with online reviews, Neurocomputing 417 (2020) 406–418.
[41] X. Wu, H. Chen, J. Wang, L. Troiano, V. Loia, H. Fujita, Adaptive stock trading strategies with deep reinforcement learning methods, Information Sciences 538 (2020) 142–158.
[42] C. Xie, D. Rajan, Q. Chai, An interpretable neural fuzzy hammerstein-wiener network for stock price prediction, Information Sciences 577 (2021) 324–335.
[43] M. Xie, H. Yin, H. Wang, F. Xu, W. Chen, S. Wang, Learning graph-based poi embedding for location-based recommendation, in: Proceedings of the 25th CIKM, 2016, pp. 15–24.
[44] Q. Xu, M. Li, C. Jiang, Y. He, Interconnectedness and systemic risk network of chinese financial institutions: A lasso-covar approach, Physica A: Statistical Mechanics and its Applications 534 (2019) 122173.
[45] Y. Yang, J. Wang, H. Yang, J. Mang, Visibility graph approach to exchange rate series, Physica A: Statistical Mechanics and its Applications 388 (20) (2009) 4431–4437.
[46] Q. Yu, Z. Yu, Z. Wang, X. Wang, Y. Wang, Estimating posterior inference quality of the relational infinite latent feature model for overlapping community detection, Frontiers of Computer Science 14 (6) (2020) 1–15.
[47] D. Zhang, J. Yin, X. Zhu, C. Zhang, Network representation learning: A survey, IEEE Transactions on Big Data 6 (1) (2018) 3–28.
[48] H. Zhang, D. Xu, Y. Wu, Predicting catastrophes of non-autonomous networks with visibility graphs and horizontal visibility, Mechanical Systems and Signal Processing 104 (2018) 494–502.
[49] Y. Zou, R.V. Donner, N. Marwan, J.F. Donges, J. Kurths, Complex network approaches to nonlinear time series analysis, Physics Reports 787 (2019) 1–97.