Information Processing and Management 60 (2023) 103358

Contents lists available at ScienceDirect

Information Processing and Management

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

A stock rank prediction method combining industry attributes and price data of stocks

Huajin Liu, Tianlong Zhao, Suwei Wang, Xuemei Li *

School of Software, Shandong University, Jinan 250101, China

ARTICLE INFO

Keywords:
Temporal Convolutional Network
Attention mechanism
Matrix factorization
Stock prediction
Learning to rank

ABSTRACT

Stock forecasting has always been challenging as the stock market is affected by a combination of factors. Temporal Convolutional Network (TCN) based on convolutional structure has been widely used in time series prediction in recent years, but the dilated causal convolution structure leaves it unable to effectively learn the dependencies between data at different time points. This paper proposes a method for stock ranking prediction. To enhance the ability of TCN to handle dependencies within series, we first develop a channel-time dual attention module (CTAM). In conjunction with TCN to process complex historical stock price data, CTAM can adaptively learn the importance of multiple price nature series of stocks and model the dependencies between the data at different times. On the other hand, due to the market industry rotation, some stocks with specific industry attributes may become market preference for a period time. To apply the industry attributes to the stock prediction, we construct an industry-stock Pearson correlation matrix and extract a vector that fully characterizes the industry attributes of stocks from it through a matrix factorization algorithm. Furthermore, the historical market preference is modeled according to the industry attribute of the stocks to generate the dynamic correlation between stocks and market preference, and this correlation is combined with the historical price features extracted by TCN for stock ranking prediction. We conduct experiments on three datasets of 950 constituent stocks of the Shanghai Stock Exchange Index, 750 constituent stocks of the Shenzhen Stock Exchange 1000 Index and 486 stocks of the S&P500 to demonstrate the effectiveness of the proposed method. On the Shanghai Stock Exchange Index dataset, the Investment Return Ratio (IRR) obtained by using the predict results of our method to guide the exchange reached 1.416, and the Sharpe Ratio (SR) reached 2.346. On the Shenzhen Stock Exchange Index dataset, the IRR reached 1.434 and the Sharpe ratio reached 2.317. On the S&P500, the IRR reached 1.491 and the Sharpe ratio reached 2.031.

1. Introduction

Technology for time series analysis has been widely used in many fields, such as finance, economy, meteorology and hydrology, signal processing and engineering technology. In the financial field, the use of time series analysis technology for stock predicting is of great significance in formulating stock trading strategies, avoiding investment risks, obtaining investment returns and other aspects. Therefore, it has attracted a lot of researchers to study related methods. Nevertheless, as the result of many uncertain factors, such as the economic situation, company prospects, investor psychology, and policies, stock price changes are a complex dynamic, nonlinear, and time varying system, making the task of predicting them challenging.

* Corresponding author.
E-mail addresses: huajin_liu@mail.sdu.edu.cn (H. Liu), tianlongzhao@mail.sdu.edu.cn (T. Zhao), 201513451@mail.sdu.edu.cn (S. Wang), xmli@sdu.edu.cn (X. Li).

https://doi.org/10.1016/j.ipm.2023.103358
Received 5 September 2022; Received in revised form 19 February 2023; Accepted 14 March 2023
Available online 4 April 2023
0306-4573/

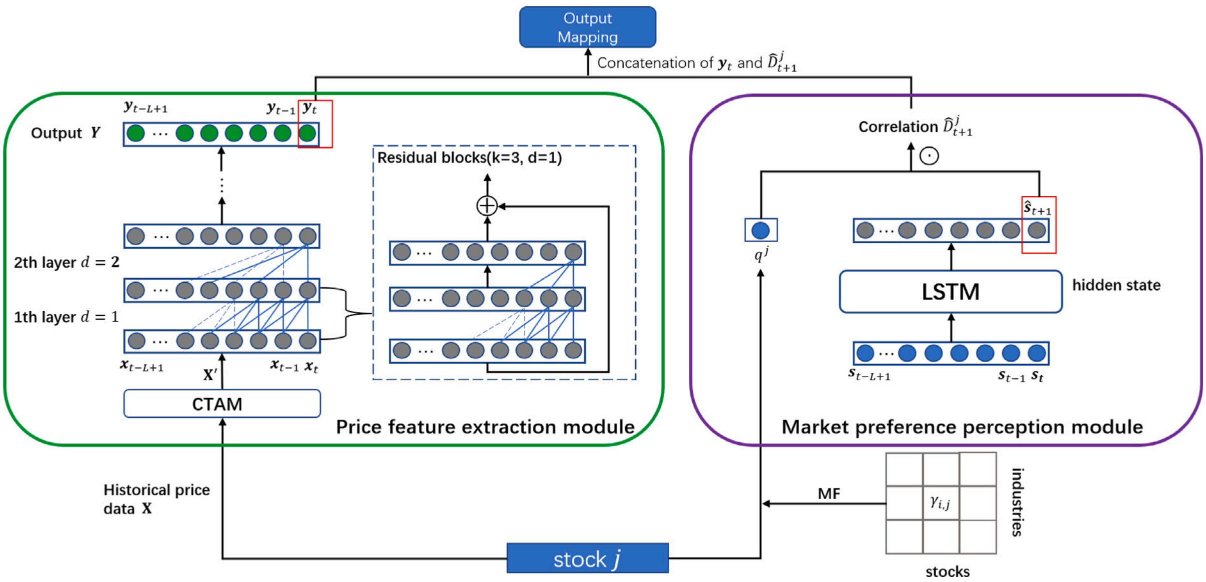


Fig. 1. Overall architecture of the proposed model, including price feature extraction module, market preference perception module and output mapping module.

Since the financial time series has a certain degree of autocorrelation (Tsay, 2005), it means that there is a dependence between the data at different time in the series. Capturing the autocorrelation over different time intervals in series is one of the keys to modeling time series. In essence, a time series model is a mathematical model that can explain the autocorrelation in a time series. In recent years, deep learning methods such as Long Short-Term Memory network (Hochreiter & Schmidhuber, 1997) (LSTM) and Gated Recurrent Unit (GRU) have been widely used in stock predicting. The structure of GRU allows it to adaptively capture dependencies in large data series without discarding information from earlier parts of the series. Therefore, modeling the dependencies in stock price series using such deep learning methods improves the accuracy of stock prediction. Bai et al. (2018) proposed the Temporal Convolutional Network (TCN) based on CNNs. Compared with RNN structure models such as LSTM and GRU, TCN has lower memory requirements (Lara-Benítez et al., 2021; Ma et al., 2021), the structure of dilated causal convolution enables TCN to demonstrate longer effective memory on time series prediction (Deng et al., 2019), which makes TCN more and more popular for time series prediction (Ang et al., 2020; Lara-Benítez et al., 2021). However, since the convolution of each layer uses a convolution kernel of fixed size and step size, TCN's ability to model the dependencies between data at different time in the series is slightly insufficient (Huang & Hain, 2021; Liu et al., 2021), which limits TCN's ability to process the dependencies in the stock price series. In addition, in the stock market, a variety of price nature, such as the lowest price, the highest price, the trading volume and the turnover amount, are regarded as the most primitive and high-quality data. These nature represent various information that affect the trend of the stock price from different aspect (Fama, 1970). For example, the lowest price and the highest price can reflect the volatility of the stock price. The volume nature can reflect the capital activity of the transaction. In order to comprehensively utilize this information, it is necessary to dynamically distinguish the importance of series of different price nature in prediction models.

Most of the deep learning methods make predictions based on the historical price, volume or their derivative data of stocks. Industry-related factors will also affect stock prices (Farrell, 1974). Certain industry attributes, such as "cyclical", "low valuation" and "low risk", are of great concern to experienced investors when trading stocks. For example, a consumer industry, such as pharmaceuticals, food and beverages, usually has low risk and stable profitability. The stocks of these industries will attract investors when the market fluctuates and investment risk is high. Industries such as steel and non-ferrous metals are usually highly cyclical. Investors are more likely to purchase stocks in these industries at low points of the cycle and sell at high points. Thus, analyzing industry attributes of stocks and using them appropriately for predicting can help improve model predicting performance. Although some investors can divide the stock's industry attributes based on their experience, the result of this artificial division is difficult to quantify and lacks unified standards. Moreover, a stock can be associated with more than one industry, and different industries may have similar attributes because of certain connections, which means that the stock may have a complex industry attributes. Therefore, how to judge the industry attributes of stocks and their formulaic representation is a challenge. In addition, how to combine the industry attributes of stocks with the stock price data is also worth studying. The generalized solution is the tensor-based methods, which can dynamically capture the correlation between different sources of information (Li, Chen et al., 2016). However, the industry attributes of stocks extracted in this paper is static feature and do not apply to the dynamic tensor-based methods. Therefore, how to combine the static industry attributes and the dynamic stock's price feature is also a problem to be solved in this paper.

In response to the above problems, we develop a deep learning-based stock ranking prediction model. The model consists of three parts: price feature extraction module, market preference perception module and output mapping module. Fig. 1 shows the model structure. Price feature extraction module use TCN as a feature extractor to extract historical price features of stocks from multi-nature series of stock price. Among them, for data to be input into TCN, a newly designed channel-temporal dual attention module (CTAM) is firstly used for preprocessing. CTAM consists of a channel attention layer and a temporal attention layer, as shown in Fig. 3. The purpose is to calculate the attention weight of each nature series in the channel dimension to distinguish the importance of different natures; then in the time dimension, model the distant position's dependencies within the multi-nature series based on self attention machanism (Vaswani et al., 2017). In addition, since the price trends of industries with similar attributes usually have high correlation, in order to represent the industry attributes of stocks, we design an industry-stock Pearson correlation matrix based on market performance and extract the vector representing the stock's industry attributes through matrix factorization. As the correlation matrix contains the correlation between stocks and different industries, as well as between different industries, so the vector obtained from it allows for a more comprehensive representation of the industry attributes of stocks. Following the acquisition of industry attributes for stocks, we further construct the market preference perception module, which dynamically captures historical market preferences according to the industry attributes for stocks, and uses LSTM to predict future market preferences to generate a dynamic correlation between stocks and future market preference. The outputs of price feature extraction module and market preference perception module are then concatenated and fed into the output mapping module to determine the stocks's ranking score. To validate the proposed method, we conduct extensive experiments on three datasets, the constituent stocks of the Shanghai Stock Exchange, the constituent stocks of the Shenzhen Stock Exchange 1000 Index, and the constituent stocks of the S&P500. The experiments show that the proposed model outperforms existing methods.

Our work contributions are summarized as follows:

- Design a channel-temporal dual attention module (CTAM) and combine it with the TCN module to construct the price feature extraction module. CTAM can effectively model the dependencies between data in different time in the multi-nature series of stock prices, thereby improving the price feature extraction module's feature extraction ability.
- The "Industry-Stock Pearson Correlation Matrix" is constructed, and a vector representing industry attributes of the stocks is extracted from it. Stock's industry attributes can be fully represented by the vector, which incorporates the correlation information between stocks and industries.
- Develop a module to integrate the industry attributes of stocks into existing deep learning methods for stock prediction. This module can dynamically capture changes in historical market preferences from an industry perspective and generate dynamic correlations between stocks and future market preferences.

The rest of this paper is organized as follows. In Section 2, we review the relevant work of stock predicting. In Section 3, we present some of the basic algorithms used in our work and the structure of our model. Section 4 presents our experimental results on different datasets using different methods, and the conclusions are given in Section 5.

2. Related works

Stock prediction methods mainly include traditional statistics-based methods as well as deep learning-based methods. Traditional stock predicting methods such as autoregression (Li, Leng et al., 2016) (AR), Autoregressive Integrated Moving Average Model (Box et al., 2015) (ARIMA), etc., typically represent the stock time series as a linear stochastic process, and use historical data to fit this stochastic process for predicting purposes. For example, Singh et al. (2010) used a vector autoregressive (VAR) model to analyze North American, European, and Asian stock market price and volatility spillovers. Ariyo et al. (2014) developed an ARIMA-based stock predicting model that showed good predicting results on datasets obtained from the New York Stock Exchange and the Nigerian Stock Exchange. Although these time series analysis models are widely used, their linear structure makes them unable to capture the complex nonlinear information of stock prices. Therefore, for stock prediction task, the traditional method still has some defects (Abu-Mostafa & Atiya, 1996; Liu, Guo et al., 2022).

In recent years, with the development of artificial intelligence technology, stock prediction methods based on machine learning, especially deep learning, have become a research hotspot (Li et al., 2020; Rathore et al., 2022). Researchers are increasingly using deep neural networks to make stock predictions since these models can mine nonlinear information in stock price series (Akita et al., 2016; Gao, 2016). Among them, the recurrent neural network (RNN), particularly the long short-term memory network (LSTM), is widely used in stock prediction because it can identify long-term dependencies in stock price series (Rather et al., 2015). The time series of stocks contain multi-frequency trading patterns, so Zhang et al. (2017) proposed a state frequency memory (SFM) model based on LSTM to capture these patterns. Using Discrete Fourier Transform (DFT), the SFM model decomposes the current input into multiple frequency components, and enters the storage unit together with the previously decomposed state frequencies. In both forward single-step and forward multi-step predictions, this method performs well. As the input series length increases, traditional LSTM performance will deteriorate. In response to this problem, some works have introduced an attention mechanism to capture longer-term dependencies in time series. For example, Qin et al. (2017) proposed Dual-stage attention mechanism based recurrent neural network (DA-RNN), which adaptively selects the hidden states of the past time steps through the attention mechanism. The experimental indicate that DA-RNN has a good performance in stock prediction. Neural networks based on recurrent structures have been the main choice for time series predicting for a long time in the past, but Bai et al. proposed a convolution-based time series prediction model, Temporal Convolutional Network (TCN). The dilated causal convolution structure of TCN allow it to have a longer effective memory and a better time series prediction effect than RNN structures. Based on this, Deng et al. (2019) first applied TCN to the stock predicting task, proving its efficiency. These deep learning methods obtain their feature representation mainly from stock price properties (such as close, trading volume, etc.) and their derived data for prediction. Due to the singularity of the data sources, the discriminative ability of the obtained features is still insufficient.

In addition to the price data, some works also mine potential information from other aspects for stock prediction (Chen et al., 2022; De Fortuny et al., 2014). Schumaker and Chen (2009) deeply analyzed the financial news, used text analysis techniques to mine effective information from it and guide the task of stock prediction. Liu, Yang et al. (2022) analyzed the impact of a company's patent activity on the company's stock price in the market. The experiment found that patent activity had a positive effect on the stock price movements of about 30% of manufacturing companies. In addition to mining other information, how to effectively fuse these information is also important. Li, Chen et al. (2016) designed a tensor based computing framework and proposed a Global Dimensionality Reduction (GDR) algorithm to capture the static and dynamic interconnections among three types of information (Firm-specific mode, Event-specific mode, Sentiment-specific mode). In addition, stocks are not isolated in the market. Based on the momentum spillover effect, the company's stock price will be affected by its affiliates, so it is of great significance to mine the relationship between stocks for prediction. Wang et al. (2023) constructed multimodal and multitemporal market information of stocks into tensor form, and use tensor principal analysis model to fully exploit the essential correlations between stocks. The development of graph deep learning method also provides a new way to explore the relationship of stocks. Cheng and Li (2021) used the attention mechanism to infer the potential relationships between stocks to construct the relation graph, and introduce a gate mechanism to model the interference of the company's attributes on the momentum spillover effect, so as to extract effective relationship embeddings. Feng et al. (2019) constructed stock industry sector and wiki relationship graphs and extracted latent features for describing the relationship between stocks via TGA module and combined price features for stock ranking prediction. Relational graph-based methods mine the industry-level connections of stocks based on their affiliation with industries and explicit relationships between industries, but fewer studies mine the deeper industry attributes of stocks from their price-movement correlations with industries.

To sum up, although there have been many studies using deep neural network models for stock prediction, the ability to predict is limited by two factors. The first is the model's own limitations. For example, the model based on RNN structure has the problem of gradient descent and gradient explosion, and the modeling ability for long time series is slightly insufficient; TCN makes it difficult to effectively learn the dependencies between different time data in the series because each layer of convolution operation uses the same size convolution kernel and step. The second is whether the data selection is sufficient and whether the hybrid model built for processing multi-source data can be applied. For the task of stock predicting, more and more related works have been focused on how to mine effective features from the information outside the stock price. In addition to the patent, news content, and other factors, mining more effective factors for stock predicting from other aspects also involves how to reasonably build a hybrid model to capture effective features from multi-source data. This paper introduces a channel-temporal dual attention module to enhance TCN's ability to model dependencies between multiple nature series and between different time data within each series. In addition, a market preference perception module is developed to extract the industry attributes of stocks from the correlation between industry and stock price series as an effective factor for prediction. Table 1 shows the key characteristics and limitations of some related methods.

3. Method

In this section, firstly, the problem to be solved by the proposed model is formulated, followed by the construction method of the industry-stock Pearson correlation matrix and how to extract the industry attributes vector of stocks, and finally the three modules of the model are explained.

3.1. Problem statement

Machine learning and deep learning methods usually approach stock predicting as a regression or classification task, i.e., predicting the price or trend (up or down) of a stock in the future. However, from the investor's perspective, neither the classification nor regression methods consider the final return, since their optimization goal is not to select the stock with the highest expected return (Feng et al., 2019). We define the stock predicting problem as a problem of predicting stock ranking. Model inputs are various price nature of the stocks (open price, close price, high price, low price, volume, etc.). For simplicity, the following are collectively referred to as "stock price data". Specifically, let M represent the number of stocks for the ranking, where each stock's price series $\boldsymbol{X} = \begin{bmatrix} x_1^1, x_2^1, \dots, x_L^1 \\ x_1^2, x_2^2, \dots, x_L^2 \\ \dots \\ x_1^N, x_2^N, \dots, x_L^N \end{bmatrix}$, $\boldsymbol{X} \in \mathbb{R}^{N\times L}$, where, L represents the length of the time series of each nature, and N represents the number of nature of the selected price data. The output is a list of rank scores for these stocks $\hat{\boldsymbol{r}}_{t+1} = \left[\hat{r}_{t+1}^1, \hat{r}_{t+1}^2, \dots, \hat{r}_{t+1}^M\right]$. $\hat{r}_{t+1}^j (\mathrm{j} = 1, 2, \dots, \mathrm{M})$ represents the ranking score of stock j at time $t+1$ predicted by the model. We define the loss function of the model in the form of Eq. (1):

$$l\left(\hat{\boldsymbol{r}}_{t+1}, \boldsymbol{r}_{t+1}\right) = \left\|\hat{\boldsymbol{r}}_{t+1} - \boldsymbol{r}_{t+1}\right\|^2 + \alpha \sum_{i=1}^{M} \sum_{j=0}^{M} \max\left(0, -\left(\hat{r}_{t+1}^i - \hat{r}_{t+1}^j\right)\left(r_{t+1}^i - r_{t+1}^j\right)\right) \tag{1}$$

Table 1
The key characteristics of prior works and Proposed.

Data used	Model structure	Methods	Key characteristics	Limitations
Price series	RNN structure	LSTM	Can find the long-term dependence in the series.	The modeling ability for long series is slightly insufficient.
		SFM	Can capture the multi frequency trading mode in the series.	
		DA-RNN	Introduce the attention mechanism and combine LSTM to capture long-term dependencies in the series.	
	Convolutional structure	TCN	Based on the dilated convolution, it shows a longer "effective memory" than the RNN structure model.	Inadequate ability to model dependencies between data at different times within the series
Combine data other than stock price	Hybrid model	Liu, Yang et al. (2022)	Analyze the company's patent activities on the stock price.	More effective feature extractors and more effective factors can be used in the future.
		Li, Chen et al. (2016)	A tensor-based method is used to fuse multiple types of information.	It can investigate the dependencies between words feature to extract useful information
		Feng et al. (2019)	Mining the relationship between stocks from the industry relationship graph and wiki relationship graph among different companies.	Integrate news and social media content with industry relations and wiki relations.
	Hybrid model	Proposed	1. Design a channel-temporal dual attention model and extract information of price series with TCN; 2. Build industry-stock Pearson correlation matrix, and extract the industry attributes of stocks through matrix decomposition algorithm.	Mine other effective information from the company's financial data and explore more powerful feature extractor in the future.

The first item in the formula punishes the real return of stocks $\boldsymbol{r}_{t+1}$ and the predicted return $\hat{\boldsymbol{r}}_{t+1}$, and the second item is used to punish the difference between the ranking order of the output yield of the model and the real ranking order. The i and j represent different stocks respectively. α is a hyperparameter, which is used to balance the two parts of the loss function.

3.2. Industry attributes of stocks

The attributes of industries are typically classified by investors and researchers based on the performance of the market and their own experiences. For example, some raw material industries (such as steel) are recognized as cyclical industries, while some industries with stable business demand and less effect on the economic cycle are classified as defensive industries. In spite of the fact that these methods are widely used, they lack a unified standard and cannot be represented digitally. A common phenomenon in the stock market is that the market preference is constantly changing, and some experienced investors will make different investment decisions at different times depending on the attributes of industries, which will result in a high correlation between price trends in certain industries with similar attributes. We proved this using cluster analysis. A list of industries and the industry's price (daily frequency) divided by "Hithink Royalflush Information Network" can be obtained directly from "Tushare". The "Hithink Royalflush Information Network" divides listed companies'stocks into corresponding industries according to their business, and calculates the price of the industry according to the constituent stocks' price in the industry. To be specific, the initial price of each industry is 1000, and the daily increase/decrease is the average of the daily increases/decreases of all the constituent stocks. Then the closing price of the industry on day t is calculated based on the initial price and the daily increase/decrease, as shown in Eqs. (2)(3),

$$\text{Price}_{t} = 1000 \times \prod_{k=1}^{t} \left(1 + \text{change}_k\right) \tag{2}$$

$$\text{change}_{t} = \sum_{i=1}^{N} s\_\text{change}_t^i \tag{3}$$

Table 2
Clustering results for industry's price increase and decrease sequences.

Cluster1	Cluster2	Cluster3	Cluster4
Semiconductor	Coal	Beverage manufacturing	Whole vehicle
Computer equipment	Iron and steel	Food processing	Electric power equipment
Communication equipment	Metal material	Chinese medicine	Military project
Artificial Intelligence	Rare earth	Chemical pharmaceutical	Home appliance
Software	Gold	Biological medicine	Textile manufacturing
5G		Private hospital	Building materials
Internet of Things		Tourism	Chemical raw material
Security		Hotel	Natural gas
AR		Airport shipping	Environmental protection
Cultural media		Retail	Port

Where $change_t$ is the industry's price increase/decrease on day t, which can be calculated by Eq. (3), N is the number of constituent stocks within the industry, $s\_change_t^i$ is the increase/decrease percentage of the ith constituent stock on day t. K-means is used to analyze the price trend correlation between each industry in the range of 2016.01.04–2019.07.03. Because there are noises in the long price sequence, and these noises are amplified by superposition with the stock price. This will affect the performance of clustering algorithm based on Euclidean distance (Aghabozorgi et al., 2015). For clustering, we choose the series of price increase and decrease $\left[\text{change}_1, \text{change}_2, \ldots \text{change}_t\right]$ which reflect the price trend more intuitively. Table 2 shows 10 samples closest to the cluster center for each cluster. In Table 2, high-tech industries such as "semiconductor" and "computer equipment" are clustered into a together (Cluster 1). In Cluster2, Six raw material industries, including "coal", "steel" and "rare earth" are grouped. These industries are typically cyclical. Cluster 3 mainly consists of "pharmaceutical", "food", "tourism" and other industries closely related to daily life, which have been considered as typical "consumer" industries, usually with a low risk and stable profitability attribute. "whole vehicle", "Electric power equipment", etc. which are related to industry are classified into the fourth group.

In addition to explicit affiliation, stocks may have potential relationships with other industries, and certain closely related industries such as cement and real estate may also share similar attributes. In order to comprehensively consider these factors, this paper directly starts from the market performance. Since the price trends of industries with similar attributes tend to have high correlation, we use the Pearson correlation coefficient (Benesty et al., 2009) of the closing price series of industries and stocks to express the correlation between the two and further construct an industry-stock Pearson correlation matrix. The purpose is to extract a vector from the matrix that fully characterizes the industry attributes of stocks.

The constructed industry-stock Pearson correlation matrix $\boldsymbol{H} \in \mathbb{R}^{M \times N}$ is as follows:

$$\boldsymbol{H} = \begin{pmatrix} \gamma^{1,1} & \gamma^{1,2} & \cdots & \gamma^{1,\mathrm{N}} \\ \gamma^{2,1} & \gamma^{2,2} & \cdots & \gamma^{2,\mathrm{N}} \\ \cdots & \cdots & \cdots & \cdots \\ \gamma^{\mathrm{M},1} & \gamma^{\mathrm{M},2} & \cdots & \gamma^{\mathrm{M},\mathrm{N}} \end{pmatrix}$$

$$\gamma^{i,j} = 1 + \rho^{i,j} \tag{4}$$

$$\rho^{i,j} = \frac{\mathrm{Cov}\left(F^i, Y^j\right)}{\sigma\left(F^i\right)\sigma\left(Y^j\right)} \tag{5}$$

Each element in the matrix is calculated by Eq. (4), where $\rho^{i,j}$ represents the Pearson correlation coefficient between ith industry's price series F^i and jth stock's price series Y^j. The Pearson correlation coefficient is used to measure the degree of correlation between two series, which is defined as the ratio of the covariance of two series and the product of the standard deviations of two series, as shown in Eq. (5). $\rho^{i,j} \in [-1, 1]$, $\rho^{i,j} > 0$ indicate that the two series are positively correlated and $\rho^{i,j} < 0$ means the opposite. This article uses industry data provided by "Hithink Royalflush Information Network".

From the industry-stock Pearson correlation matrix, we extract the vectors representing the industry attributes of stocks. In this paper, the non-negative matrix factorization (Lee & Seung, 2000) (NMF) algorithm is applied to factorize the matrix $\boldsymbol{H}$ to obtain the representation vector of the stock's industry attributes. NMF algorithm is widely used in recommender systems (Koren et al., 2009), text mining (Pauca et al., 2004), and other fields. It can obtain the vector representation of the potential attributes of entities based on their interaction. As each element in matrix $\boldsymbol{H}$ represents the correlation between individual stock and a certain industry, the vector produced by factorizing the overall matrix integrates the correlation between individual stocks and different industries, as well as industries and industries, allowing for a more comprehensive understanding of the industry attributes of stocks. The experiments in Section 4 of this paper demonstrate the effectiveness of vectors extracted from matrix. The process of matrix factorizing is shown in Fig. 2. The matrix $\boldsymbol{H}$ is factorized into the form of multiplying the matrix $\boldsymbol{E}_1 \in \mathbb{R}^{M \times K}$ and $\boldsymbol{E}_2 \in \mathbb{R}^{K \times N}$ where the vector $\boldsymbol{p}^i \in \mathbb{R}^K$ in $\boldsymbol{E}_1$ represents the potential attributes of industry i, and the vector $\boldsymbol{q}^j \in \mathbb{R}^K$ in $\boldsymbol{E}_2$ represents the industry attributes of stock j.

The NMF algorithm estimates the vectors $\boldsymbol{p}^i$ and $\boldsymbol{q}^j$ by optimizing the loss function,

$$J = \sum_{i,j} \left(\gamma^{i,j} - \left(\boldsymbol{p}^i\right)^T \boldsymbol{q}^j + b_i + b_j\right)^2 \tag{6}$$

where b_i and b_j represent the bias terms corresponding to industry i and stock j.

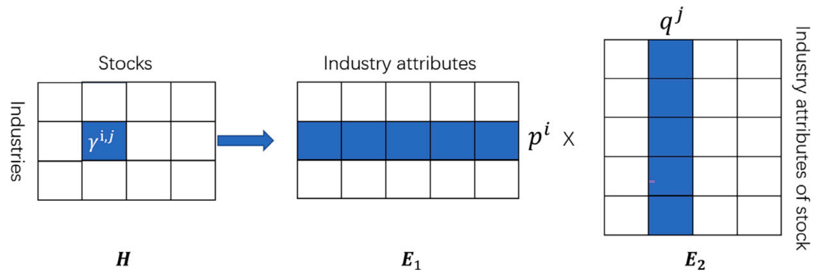


Fig. 2. The process of the NMF algorithm.

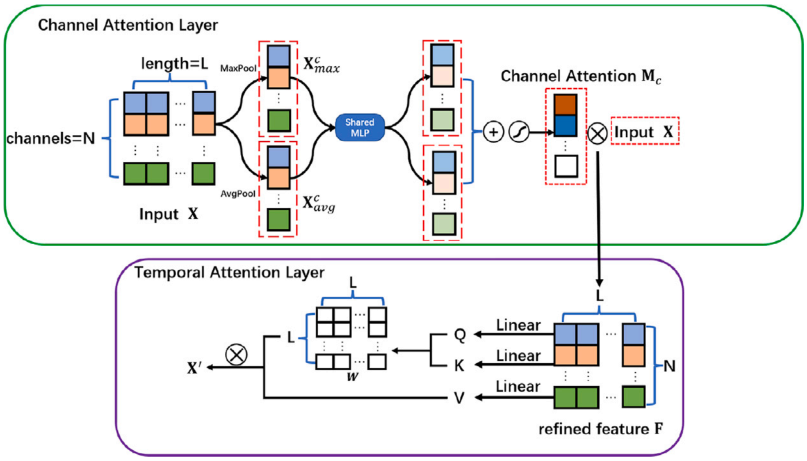


Fig. 3. The structure of CTAM.

3.3. Model structure

We propose a novel model for stock ranking prediction, as shown in Fig. 1. The model is mainly composed of three modules, a price feature extraction module, a market preference perception module, and an output mapping module. The price feature extraction module mines the features of the historical stock price data. The multiple price nature series of stock price data is first preprocessed by the input channel-temporal dual attention module (CTAM). As shown in Fig. 3, the CTAM module consists of two layers: channel attention layer and time attention layer. It is intended to calculate the attention weight of each nature series in the channel dimension to distinguish the importance of various nature and further focus on the dependencies between data at different times within the series in the time dimension. With the above preprocessing, a new series $\boldsymbol{X}'$ is created that the influence of other time data is added to the current time data in the form of weight. After that, the new series $\boldsymbol{X}'$ is input to TCN for feature extraction. The function of the market preference perception module is to dynamically capture the historical market preference according to the industry attributes of stocks and use LSTM to predict the future market preference, so as to generate the dynamic correlation between stocks and market preference. Among them, in order to represent the industry attributes of stocks, we decompose the industry-stock Pearson correlation matrix and extract the vector representing the industry attributes of stocks. The correlation matrix contains the correlations between stocks and different industries, as well as between different industries, so the vector obtained from it allows for a more comprehensive representation of the industry attributes of stocks. The historical price features of stocks and the correlation between stocks and future market preferences are then concatenated and fed into the output mapping module to generate the ranking scores of stocks.

Price Feature Extraction Module The function of the price feature extraction module is to extract features from the stock's historical price data (including opening price, closing price, highest price, lowest price, trading volume, and trading volume). As shown in Fig. 1, the input of this module is a time series $\boldsymbol{X} = \{\boldsymbol{x}_{t-L+1}, \dots, \boldsymbol{x}_t\}$ of length L, where $\boldsymbol{x}_t \in \mathbb{R}^N$ represents the historical price data of the stock at time t, N represents the number of nature such as closing price and volume. The module output $\boldsymbol{Y} = \{\boldsymbol{y}_{t-L+1}, \dots \boldsymbol{y}_t\}$ is a time series of the same length as the input. $\boldsymbol{y}_t \in \mathbb{R}^O$, where O is the number of output channels of the last convolutional layer of TCN. In the price feature extraction module, the multi-nature series X of stock price data is first input into CTAM for processing, and then TCN performs convolution operation to extract features.

The CTAM's structure is shown in Fig. 3, including two parts, the channel attention layer and the temporal attention layer, which are arranged in series order. These two parts calculate the attention matrix from the channel and time dimensions, respectively. The channel attention layer first calculates the channel attention matrix $\boldsymbol{M}_c \in \mathbb{R}^{N\times 1}$ according to the input series $\boldsymbol{X}$ to distinguish the importance of different nature, and the channel attention matrix is multiplied with the original input $\boldsymbol{X}$ to obtain the intermediate output $\boldsymbol{F} \in \mathbb{R}^{N\times L}$. The temporal attention layer models the dependencies of different temporal data within the series $\boldsymbol{F}$ based on the self-attention mechanism to obtain the final output $\boldsymbol{X}'$ of CTAM.

In the channel attention layer, average pooling and max pooling operations are used to compress the temporal dimension of the input data $\boldsymbol{X}$, generating two intermediate features $\boldsymbol{X}^c_{avg} \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{X}^c_{max} \in \mathbb{R}^{N\times 1}$. The two intermediate features are input into a shared network MLP and pass through the sigmoid activation function to output the channel attention matrix $\boldsymbol{M}_{\mathbf{c}} \in \mathbb{R}^{N\times 1}$, $\boldsymbol{M}_c$ and $\boldsymbol{X}$ are multiplied to obtain the intermediate output $\boldsymbol{F}$. The overall calculation process is as follows:

$$\begin{aligned}\boldsymbol{M}_c(\boldsymbol{X}) &= \sigma(\mathrm{MLP}(\mathrm{AvgPool}(\boldsymbol{X})) + MLP(\mathrm{MaxPool}(\boldsymbol{X}))) \\ &= \sigma\left(MLP\left(\boldsymbol{X}^c_{\mathrm{avg}}\right) + MLP\left(\boldsymbol{X}^c_{\mathrm{max}}\right)\right)\end{aligned} \tag{7}$$

$$\boldsymbol{F} = \boldsymbol{M}_c(\boldsymbol{X}) \otimes \boldsymbol{X} \tag{8}$$

The $\otimes$ represents element-wise multiplication, and during the multiplication process, $\boldsymbol{M}_c$ is broadcasted.

In the temporal attention layer, three linear operations are used to transform $\boldsymbol{F}$ in the channel dimension to obtain $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{d\times L}$ and then the weight matrix $\boldsymbol{W} \in \mathbb{R}^{L\times L}$ is obtained by the following calculation

$$\boldsymbol{W}_{i,j} = \mathrm{softmax}\left(\frac{\boldsymbol{k}_i^T \cdot \boldsymbol{q}_j}{\sqrt{d}}\right) \tag{9}$$

where $\boldsymbol{k}_i \in \mathbb{R}^d$ and $\boldsymbol{q}_j \in \mathbb{R}^d$ represent the data of $\boldsymbol{K}$ at time i and $\boldsymbol{Q}$ at time j, respectively. The output $\boldsymbol{X}'$ of the temporal attention module is calculated by Eq. (10).

$$\boldsymbol{X}' = \boldsymbol{V}\boldsymbol{W} \tag{10}$$

The output $\boldsymbol{X}'$ of CTAM is then input to TCN for feature extraction. TCN expands the range of information extraction through multiple layers of dilated causal convolution operations. In order to avoid the performance degradation caused by the increase of the number of layers, each layer of TCN is designed as a residual module (He et al., 2016), as shown in Fig. 1. The residual module performs two expansion causal convolution operations on the input data, and the result of the convolution operations is added to the input to obtain the output of the residual module. For the time series $\boldsymbol{h}' \in \mathbb{R}^L$ for each channel in $\boldsymbol{X}'$, the operation of dilated causal convolution is expressed as follows:

$$\boldsymbol{c}_t = \sum_{i=0}^{k-1} f_i \cdot \boldsymbol{h}'_{t-d\cdot i} \tag{11}$$

$\boldsymbol{f} = \left(f_0, f_1 \dots f_k\right)$ represents the convolution kernel, k represents the size of the convolution kernel, d is the expansion coefficient, and $\boldsymbol{h}'_{t-d\cdot i}$ represents the information before time t. For expanding the range for extracting information, d increases exponentially with the depth of the network (i.e. at the nth layer of the network, $d = O(2^n)$).

The final output series of TCN is $\boldsymbol{Y} \in \mathbb{R}^{O\times L}$ and we take the data $\boldsymbol{y}_t \in \mathbb{R}^O$ at the last moment of $\boldsymbol{Y}$ as the stock price feature extracted by the module.

Market preference perception module Affected by factors such as seasons and policies, investors in the market may prefer certain industries for a period of time. We have constructed a market preference perception module to capture the changing laws of market preferences in the past to predict future market preferences and then generate the correlation between stocks and future market preferences, as shown in Fig. 1. Historical market preferences are represented as a time series of length L: $\boldsymbol{S} = \left\{\boldsymbol{s}_{t-L+1}, \dots, \boldsymbol{s}_t\right\}$. Considering that stocks with high correlation with market preference usually have higher price increases, we choose the stocks with the highest return to represent the market preference. The market preferences $\boldsymbol{s}_t$ at time t can be expressed as:

$$\boldsymbol{s}_t = \frac{1}{G}\sum_{j=1}^{G} \boldsymbol{q}^j \tag{12}$$

G represents the number of stocks with higher return ranking, and $\boldsymbol{q}^j$ represents the industry attributes of stock j extracted by matrix factorization.

In order to obtain future market preferences, we choose long short-term memory network (LSTM) to model historical market preferences and predict future market preferences, as shown in Eq. (13). LSTM is a special RNN. For time series, LSTM processes the data at the corresponding time through a storage unit, and adds the gated structure to the unit to control the transmission of information, avoiding the gradient disappearance and explosion problems of traditional RNN.

$$\hat{\boldsymbol{s}}_{t+1} = \mathrm{LSTM}\left(\boldsymbol{s}_{\le t}\right) \tag{13}$$

We use the hidden layer vector of the last unit of the LSTM to represent future market preferences $\hat{\boldsymbol{s}}_{t+1}$ and calculate the correlation between stocks and future market preferences based on the stock industry attributes vector.

$$\widehat{D}^j_{t+1} = \hat{\boldsymbol{s}}_{t+1} \cdot \boldsymbol{q}^j \tag{14}$$

Output Mapping Module In the output mapping module, we concatenate the stock price feature $\boldsymbol{y}_t$ and the correlation $\widehat{D}^j_{t+1}$ of stock j and market preference and input to the fully connected layer to predict the stock's ranking score, as shown in Eq. (15).

$$\hat{r}^j_{t+1} = MLP\left(\left[\boldsymbol{y}_t, \widehat{D}^j_{t+1}\right]\right) \tag{15}$$

$\hat{r}_{t+1}^{j}$ represents the predicted ranking score of stock j in the future.

The overall process of the above several modules is as Algorithm 1

Algorithm 1 Model training process

Input: $\boldsymbol{X}$: $\left[\boldsymbol{x}_{t-L+1}, \dots, \boldsymbol{x}_t\right]$, represent historical price series, $\boldsymbol{x}_t \in \mathbb{R}^N$, represents the N price nature at day t.
M: The number of stocks for ranking.
$\boldsymbol{S}$: $\left[\boldsymbol{s}_{t-L+1}, \dots, \boldsymbol{s}_t\right]$, represent historical market preference, $\boldsymbol{s}_t \in \mathbb{R}^N$, represent the market preference at day t, which is calculated by eq(12).
Output: $\hat{\boldsymbol{r}}_{t+1} = \left[\hat{r}_{t+1}^1, \hat{r}_{t+1}^2, \dots, \hat{r}_{t+1}^M\right]$. $\hat{r}_{t+1}^j(\mathrm{j} = 1, 2, \dots, \mathrm{M})$, represents the ranking score of stock j at day t+1.

Initialize parameters of TCN and LSTM
$\hat{\boldsymbol{s}}_{t+1} = \mathrm{LSTM}\left(\left[\boldsymbol{s}_{t-L+1}, \dots, \boldsymbol{s}_t\right]\right)$, future market preferences predicted using LSTM.
for $j = 1$ to M **do**
  Get $\boldsymbol{X}^j$: The historical price series of the jth stock.
  Get $\boldsymbol{q}^j$: The industry attributes vector of the jth stock.
  Get $\boldsymbol{X}^{j'}$: The output from processing $\boldsymbol{X}^j$ using "CTAM".
  Get $\boldsymbol{y}_t$: The feature extracted by TCN through convolution operation on $\boldsymbol{X}^{j'}$.
  $\widehat{D}_{t+1}^{j} = \hat{\boldsymbol{s}}_{t+1} q^j$
  $\hat{r}_{t+1}^{j} = MLP([\boldsymbol{y}_t, \widehat{D}_{t+1}^{j}])$
end for
Get $\hat{\boldsymbol{r}}_{t+1} = \left[\hat{r}_{t+1}^1, \hat{r}_{t+1}^2, \dots, \hat{r}_{t+1}^M\right]$.
Calculate the loss function according to eq(1).
Update parameters of TCN and LSTM by gradient descent.

4. Experimental setup

In this section, we introduce the dataset in the experiment and the preprocessing of the data. Then the comparison methods and the indicators used to evaluate the effect of these models are given. Finally, we give the settings of some model-related parameters.

4.1. Data preparation and preprocessing

We select the constituent stock's data of the Shanghai Stock Exchange ("SHSE") Index, the Shenzhen 1000 ("SZSE1000") Index and the S&P500 index as the three datasets of the experiment. The two datasets of Chinese stock market come from the "Tushare", and each stock uses six nature of opening price, closing price, highest price, lowest price, trading volume and turnover as its daily price data. The data source for S&P500 constituent stocks is "Yahoo finance", and each stock uses five nature: opening price, closing price, highest price, lowest price and trading volume. We select the daily price data from 2016.01.04 to 2020.12.31, the time series's length of SHSE and SZSE1000 is 1218 and the length of S&P500 is 1259. In order to avoid too much missing data affecting the accuracy of the experimental results, the stocks that were not listed before 2016.01.04 and stocks with more than 20 suspensions in the selected time interval are removed. In the end, the datasets are 950 stocks in SHSE, 750 stocks in SZSE1000 and 486 stocks in S&P500.

In terms of industry data, for SHSE and SZSE1000, we select the closing price data of the industries from 2016.01.04 to 2020.12.31, and the data source is "Tushare". This data is provided by "Hithink Royalflush Information Network", and the calculation method is explained in Section 3.2. For S&P500, we obtain the daily price of 11 industry ETFs from 2016.01.04 to 2020.12.31 as industry data, and these ETF codes are "XLB", "XLRE", "XLP", "XLY", "XLU", "XLE", "XLI", "XLC", "XLF" and "XLK" respectively. The datasource is "Yahoo finance".

The series corresponding to each nature in the stock price data is normalized (Ioffe & Szegedy, 2015) before entering the model, as shown below:

$$\bar{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{16}$$

where x represents the original value of the series, and $x_{\max}$ and $x_{\min}$ represent the minimum and maximum values of the series in the time interval, respectively. In this way, each value in the series is limited to the range [0,1].

In this experiment, the data of the previous 70% time are used as the training set, while the remaining 10% and 20% are used as the verification set and test set respectively. The input series length of the model is 10, that is, the price data of the past 10 days is used to predict the next day. The sample division is shown in Table 3. In addition, to avoid data leakage, we use the stock and industry price series in the training set time interval to build Pearson correlation matrix.

4.2. Evaluation indicators

Our goal is to predict the return ranking of stocks in dataset, so this paper uses three indicators: Precision, Sharpe ratio (SR) and cumulative investment return (IRR) to evaluate the performance of the model.

Table 3
The sample division.

		Time interval	Length	Number of samples
SHSE, SZSE1000	Train set	2016.01.04–2019.07.03	852	842
	Valid set	2019.07.04–2019.12.31	123	113
	Test set	2020.01.02–2020.12.31	243	233
S&P500	Train set	2016.01.04–2019.06.06	882	872
	Valid set	2019.06.07–2019.12.18	125	115
	Test set	2019.12.19–2020.12.31	252	242

Precision Precision is an indicator for evaluating the effect of the model on the prediction of stock ranking. It refers to the proportion of the stocks in the top k of the output list whose actual returns are also in the top k. It is calculated as follows:

$$\text{Precison}@k = \frac{L@k(\hat{r}) \cap L@k(r)}{k} \tag{17}$$

$L@k(\hat{r})$ and $L@k(r)$ respectively represent the top k stock set according to the ranking score predicted by the model and the top k stock set according to the real ranking score. The larger the value of Precison, the better the ranking effect of the model, we choose $Precison@20$ and $Precison@50$ as the ranking evaluation indicator.

Investment return ratio (IRR) In the time interval of the test set, the test samples (that is, the price data of the past ten days) are input into the model and predicted to get the ranking list of stock returns of the next day. Select top k stocks in the list and buy them at the closing price of the day, then sell them at the closing price of the next day. Because the trading mode is relatively simple, and it is difficult to accurately buy and sell at the closing price in the actual trading, that is, the slippage effect may occur. In order to reduce the impact of slippage on the return rate, it is necessary to buy more stocks in trading. In the experiment, k=20.

Sharpe ratio The Sharpe Ratio (SR), which quantifies the average return per unit of volatility over the risk-free rate, is calculated as:

$$R_t = r_t - \frac{Q}{252} \tag{18}$$

$$SR = \frac{\sqrt{252}m}{S} \tag{19}$$

r_t represents the return on the t day, Q represents the risk-free interest rate, this paper uses the average yield of China's one-year government bond in 2019 as the risk-free rate, and R_t represents the rate of return over the risk-free rate on the t day, m is the mean of R_t, and s is the standard deviation of R_t. We select the average return of the top 20 stocks in the ranking list given by the model on day t as the return.

4.3. Comparison method

To evaluate the performance of our proposed model on the stock rank prediction task, we compare it with several deep learning models. All these methods use open, close, high, low, volume, and turnover six attributes as price data for prediction. Specifically, we compare the following models:

LSTM Input the time series of historical stock price data into the LSTM model, and input the hidden layer state of the storage unit corresponding to the last time step into the fully connected layer to predict the future ranking scores of stocks.

SFM On the basis of LSTM, the state frequency memory model (SFM) decomposes the memory cell state into signals of multiple frequencies using discrete Fourier transform, and learns feature embeddings of different frequencies to capture different transaction patterns.

PF-RNN In order to better deal with highly variable stock data, PF-RNN (Ma et al., 2020) approximates the hidden state of traditional RNN as a set of particles, and updates the hidden state according to Bayesian rule.

VML VML (Liu, Ma et al., 2022) decomposes the original stock price series by VMD, processes each subsequence through LSTM, and introduces the meta-learning method to train the model.

SCINet SCINet (Liu et al., 2022d) is a latest time series prediction model based on convolution structure. Recursively decomposes a sequence into even and odd subsequences, and convolutes each subsequence using a set of filters to extract different but valuable time features from each part.

4.4. Parameter settings

In our work, both the contrasting method and our model use historical 10-day data to perform one-step forward predictions. Grid search is used to find the optimal parameter combination, so that all methods perform best on the validation set. In our method, we look for the size of the vector dimension representing the industry attributes of the stocks in [8,16,32,64,128], which is also the vector dimension of the hidden layer of the LSTM in the market preference perception module. We tune the number of the TCN's channels within [32,64,128,256,512]. The kernel size of TCN is set to 3, the stride of the convolution operation is 1, and the number of layers of TCN is 4. We use the Adam optimizer (Kingma & Ba, 2014) to train the model and set the learning rate to 0.001.

Table 4
Experimental results of different methods on three datasets.

	SHSE				SZSE1000				S&P500			
	Pre20	Pre50	SR	IRR	Pre20	Pre50	SR	IRR	Pre20	Pre50	SR	IRR
LSTM	0.048	0.063	1.378	0.409	0.036	0.072	1.517	0.554	0.065	0.098	1.417	0.608
SFM	0.058	0.075	1.616	0.660	0.038	0.074	1.594	0.631	0.072	0.125	1.461	0.843
PF-RNN	0.056	0.079	1.780	0.818	0.062	0.086	1.979	1.027	0.089	0.136	1.777	1.247
VML	0.061	0.085	2.120	1.173	0.069	0.095	2.209	1.286	0.075	0.132	1.785	1.137
SCINet	0.059	0.090	1.977	1.050	0.072	0.106	2.303	1.392	0.083	0.159	1.754	1.110
OURS	**0.071**	**0.098**	**2.346**	**1.416**	**0.078**	**0.129**	**2.317**	**1.434**	**0.094**	**0.178**	**2.031**	**1.491**

5. Experimental results and analysis

We first conduct comparative experiments on three datasets to verify the effects of various methods, and then conduct ablation experiments on three data sets to verify the effectiveness of each module in the model. In addition, similar to most existing stock predicting work based on daily data, our experiments are based on two assumptions: 1. We can buy or sell stocks at the closing price of the day. 2. Buying and selling operations do not affect stock price changes.

5.1. Comparative experiment

Our model is compared with several models in 4.3, Table 4 shows the performance of different methods on three data sets. The Precision of our model is the highest. On three datasets, the Pre20 reaches 0.071, 0.078, and 0.094, while the Pre50 reaches 0.098, 0.129, and 0.178. These proves the effectiveness of the proposed method for the stock ranking prediction task. From an IRR and SR perspective, using our model to guide transactions yields the highest return with the lowest risk. In the SHSE, the SR and IRR reach 2.346 and 1.416, respectively. In the SZSE1000, the two indicators reach 2.317 and 1.434. According to the S&P500 dataset, the IRR also reached 1.491, but the SR decreased to 2.031 as compared to the previous two datasets due to the volatility of U.S. stock market.

In order to show the experimental results more intuitively, we also provide the IRR curves of the six methods on the three datasets, as shown in Fig. 4, where the "Market" is the IRR obtained by buying and selling all the stocks in the index, reflecting the return rate obtained by investing the whole index. The figure shows that using our method to guide the exchange yields the highest IRR. In the comparison methods, VML has the best performance on SHSE, with an IRR of 1.173, while SCINet has an IRR of 1.392 on SZSE1000. The IRR of PF-RNN on the S&P500 dataset is 1.247. Compared with the two baseline models LSTM and SFM, the above three methods have shown good improvement. Due to the impact of COVID-19 on the market, several methods have a certain retracement of returns in the first 50 trading days of the test set, while the retracement of the method in this paper and the VML method is minimal. It is worth noting that in the Chinese stock market, using our model to guide the exchange produced a significantly better return than other methods during the 70–150 trading day period. We analyze this is because during this period, with the stabilization of COVID-19 and investors' confidence having increased, the overall liquidity of the market has been good and there is a relatively obvious rotation phenomenon of hot industries.

5.2. Ablation experiment

To evaluate the effectiveness of the market preference perception module and the CTAM, we conduct the following experiments on the two datasets, and the specific experimental results are shown in Table 5.

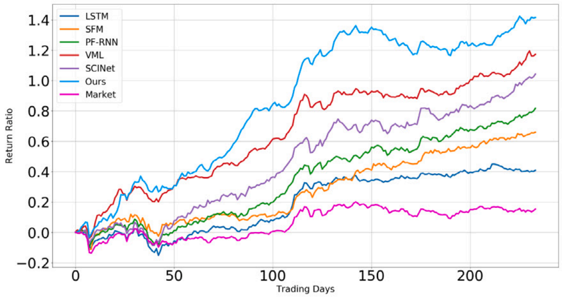
TCN+CTAM On the basis of the original model, the market preference perception module is removed.

TCN+MP On the basis of the original model, CTAM is removed.

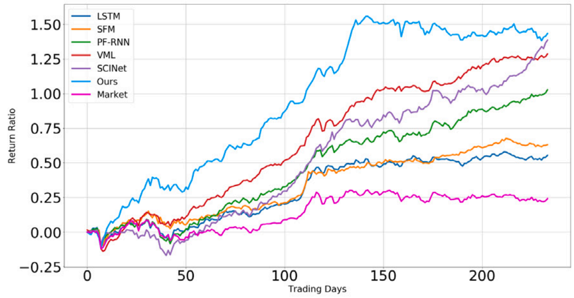
TCN Remove both the market preference module and the CTAM, that is, use TCN alone.

According to Table 5, using the market preference perception module or the CTAM on the basis of TCN produces better results than using TCN alone, and using both modules together has a greater impact than using either module alone. This is because the market preference perception module can assign higher ranking scores to stocks with a particular industry nature according to the current market preference, and the price feature extraction module with CTAM can analyze historical stock price data to extract features.

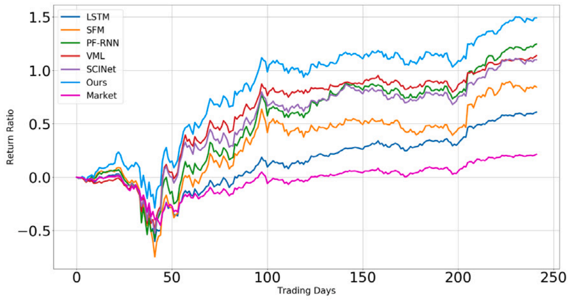
The IRR curves of these methods on the three data sets are shown in Fig. 5. According to Fig. 5, the IRR using TCN for SHSE, SZSE1000, and S&P500 datasets is 0.759, 0.732, and 0.639, respectively. The IRR of TCN+CTAM is 1.191, 0.994 and 0.861, respectively, which was 43.2%, 26.2%, and 22.2% higher than that of TCN. This indicates that the CTAM can learn the dependencies between multiple nature series and different time data within the series, improving the feature extraction ability. TCN+MP's IRR on the three datasets is 0.988, 0.822, 1.217, which is 22.9%, 9.0%, 57.8% higher than TCN's. This indicates that the market preference perception module can better detect the dynamic correlation between stocks and market preference based on the industry attributes of stocks, which can enhance the model prediction performance. Our model has the highest IRR among three datasets, indicating that the combination of CTAM and market preference perception module can work better.



(a) SHSE

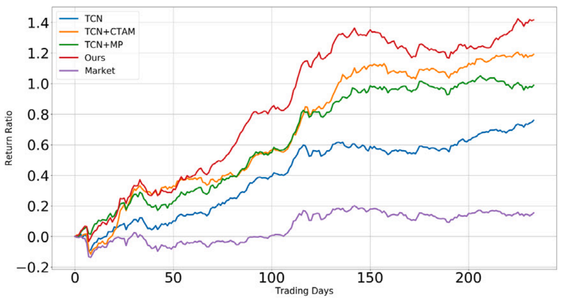


(b) SZSE1000

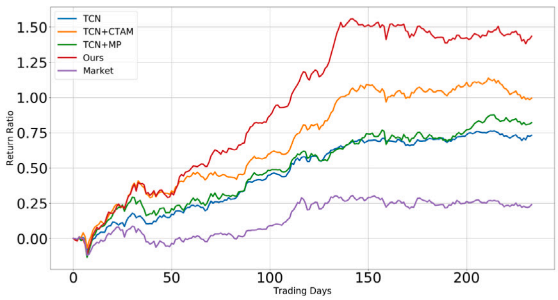


(c) S&P500

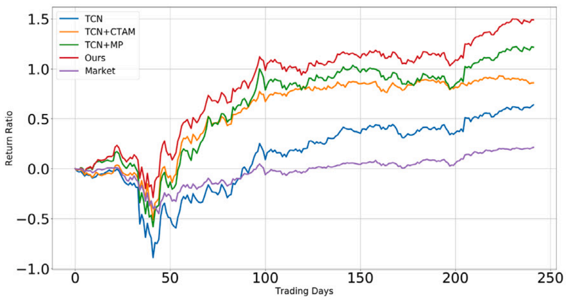
Fig. 4. Back-testing IRR indicator of different methods on three datasets.



(a) SHSE



(b) SZSE1000



(c) S&P500

Fig. 5. Back-testing IRR indicator of different methods on three datasets.

Table 5
Experimental results of different methods on three datasets.

	SHSE				SZSE1000				S&P500			
	Pre20	Pre50	SR	IRR	Pre20	Pre50	SR	IRR	Pre20	Pre50	SR	IRR
TCN	0.050	0.071	1.729	0.759	0.038	0.074	1.705	0.732	0.071	0.095	1.293	0.639
TCN+MP	0.056	0.079	1.946	0.988	0.053	0.093	1.756	0.822	0.085	0.129	1.737	1.217
TCN+CTAM	0.059	0.083	2.136	1.191	0.058	0.081	1.928	0.994	0.073	0.135	1.578	0.861
OURS	**0.071**	**0.098**	**2.346**	**1.416**	**0.078**	**0.129**	**2.317**	**1.434**	**0.094**	**0.178**	**2.031**	**1.491**

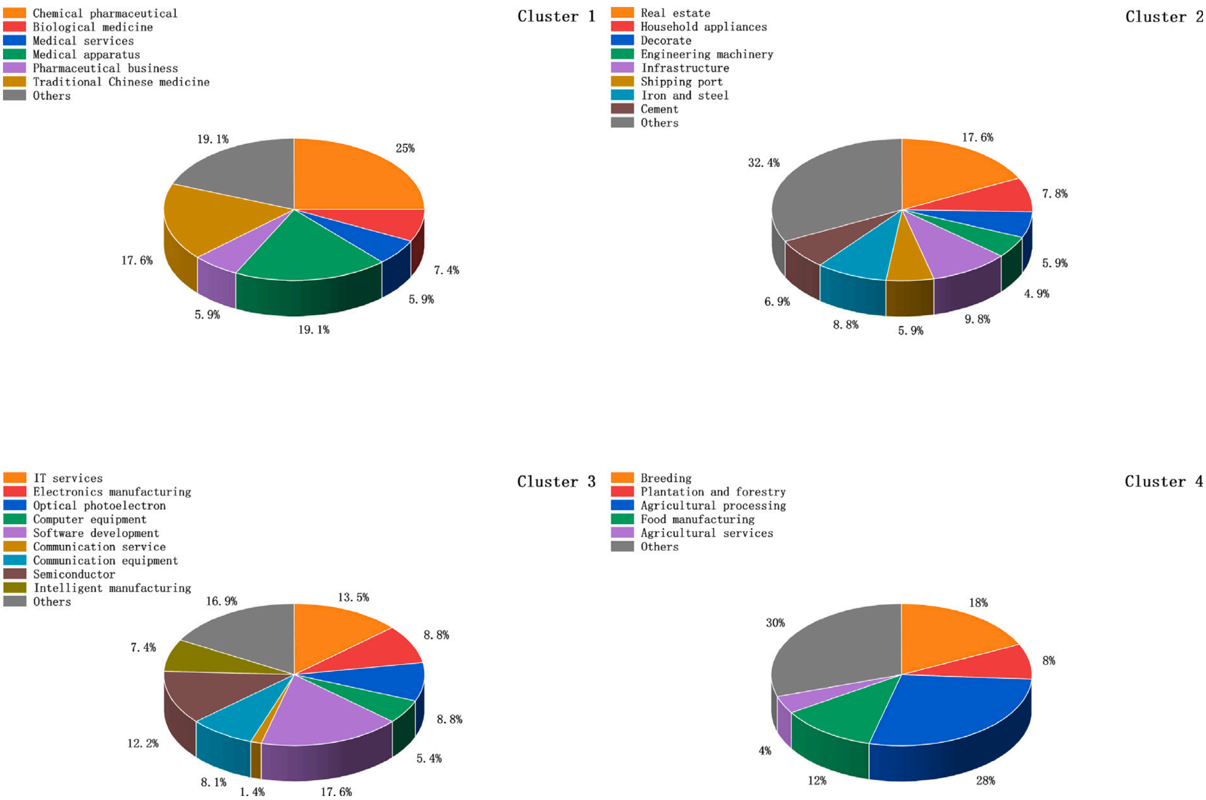
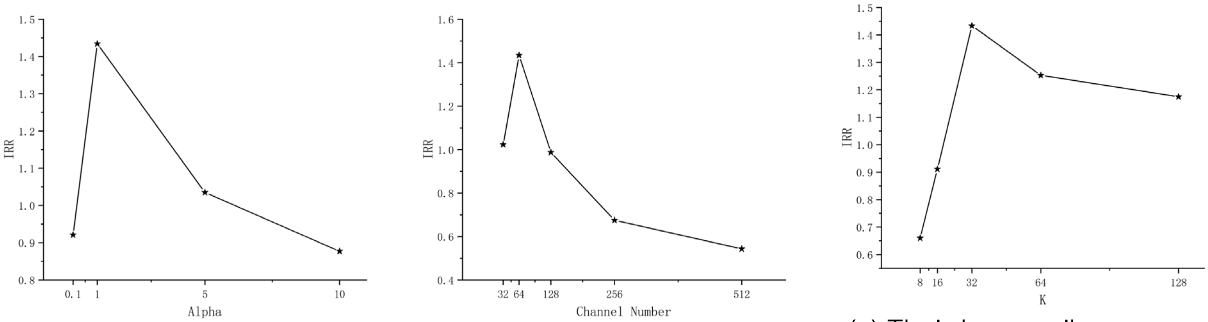


Fig. 6. Four examples of clusters generated based on stock's industry attributes.

5.3. Analysis of the industry attributes of stocks

To further verify the validity of the industry attributes of stocks extracted from the industry-stock Pearson correlation matrix, we conducted some analyses. We hope that the vector representing the industry attributes of stocks obtained from the matrix can effectively reflect the correlation information between individual stocks and the industry as well as between industries. Therefore, Shenzhen Stock Exchange 1000 Index constituent stocks are clustered according to the vector corresponding to their industry attributes. The clustering method used is K-means, and then the industry of the companies in each cluster is analyzed. Fig. 6 shows four clusters out of all the clustering results. In the first cluster, the industries to which companies belong are mainly related to medical care, such as chemical pharmaceuticals, medical devices, traditional Chinese medicine, etc., and medical-related companies account for 80.9%. The companies in the second cluster mainly belong to the real estate, decoration, infrastructure, cement, steel and other industries. These companies account for 67.6%. Although they belong to different industries, they are closely related. In the third cluster, 84.1% of the company's main business is related to technology, such as IT services, semiconductors, software development, etc., and 70% of the company's business in the last category is related to agriculture, such as breeding, agricultural processing, planting and forestry, etc. Clustering results like these indicate that the industry attribute vectors of stocks extracted in this study have sufficient representational significance.



(a) The α in the loss function (b) The number of channels of TCN (c) The industry attributes vector dimension K

Fig. 7. The effect of different parameters on model performance.

5.4. Parameter sensitivity analysis

In this section, we conduct experiments on the Shenzhen Stock Exchange 1000 Index data set, and study the impact of three important hyperparameters on the prediction performance of the model, including the α value of balancing the two loss terms in the loss function, the number of channels of TCN, and the dimension of the stock's industry attributes vector.

The α in the loss function In order to analyze the influence of the hyperparameter α of the model loss function in Eq. (1) on the model ranking performance, we selected 0.1, 1, 5, and 10 as α for experiments. The experimental results are shown in Fig. 7(a). When the α is 0.1, the final IRR of trading according to the model ranking is 0.921; when the value of α increases to 1, the value of IRR increases to 1.434; when the value of α continues to increase to 5 and 10, the performance of the model begins to decline. The reason for this may be: when α is too small, the model loss function has a smaller penalty for the second term of Eq. (1), so that the model is insensitive to stock ranking order errors. When α is too large, the penalty for the first item by the loss function becomes relatively small, which leads to a decrease in the accuracy of the stock returns predicted by the model, which in turn affects the final stock ranking.

The number of channels of TCN The number of channels in the middle layer of TCN represents the dimension size of the price intermediate feature extracted by TCN. We set the number of intermediate layer channels to 32, 64, 128, 256, and 512, respectively. The experimental results are shown in Fig. 7(b). It can be seen that the prediction effect of the model is the best when the number of channels is 64. The reason may be that a small number of channels will limit the intermediate feature representation capability of the model, and an excessive number of channels will easily cause the model overfitting.

The industry attributes vector dimension K We set the dimensions of the vectors to 8, 16, 32, 64, and 128, respectively, and the experimental results are shown in Fig. 7(c). When $K = 8$, the prediction performance of the model is the worst, and when K increases to 32, the performance of the model reaches the best. The reason may be that the vector obtained by matrix factorization contains the correlation information between stocks and industries, and between industries and industries, and the smaller vector dimension cannot fully characterize this information. When K continues to increase, the prediction performance of the model declines. This may be because the dimension of the vector representing market preference is the same as that of the vector representing the industry attributes. Larger K makes the market preference perception module unable to effectively model historical market preference, which affects the model prediction performance.

6. Conclusion

For ranking prediction, we propose a novel method that combines the historical price data of stocks and the industry attributes of stocks. The channel-Time Dual attention module (CTAM) we designed can effectively improve the ability of TCN to model the internal dependencies of multi-natures series of stock price, thereby enhancing the feature extraction ability of the module. By constructing the Pearson correlation matrix of industries and stocks and performing matrix factorization, the industry attributes vector of stocks can be constructed by combining the information of the correlation between industries and industries and the correlation between stocks and industries, so as to better reflect commonalities and differences between stocks. Further, we construct a market preference perception module that dynamically perceives changes in market preference from an industry perspective by modeling historical market preferences and predicting future market preferences to enhance the predictive performance of the model. Experimental results on two data sets of the Shanghai Stock Exchange Index and the Shenzhen Stock Exchange 1000 index shows the effectiveness of the method in this paper for the stock ranking prediction, especially in the "bull market" where the phenomenon of industry rotation is relatively obvious. During this period, using our method to guide trading yields significantly better cumulative returns than other comparative methods.

This paper uses the correlation between industry and stock price trends to mine industry attributes of stocks and applies them to predicting. We can consider mining other potential attributes of stocks from more data perspectives in the future, such as company financial factor data, social media text information, etc. Meanwhile, when compared to daily frequency data, tick data can accurately reflect the most basic information about market transactions, and more directly reflect the power balance between long and short sides of trading, which is conducive to our analysis and investment. Ticks data will be used in the future for our work.

CRediT authorship contribution statement

Huajin Liu: Mainly responsible for method research, Data experiment, Draft collaboration. **Tianlong Zhao:** Methodological studies, Data analysis, etc. **Suwei Wang:** Methodological studies, Data analysis, etc. **Xuemei Li:** Propose concepts, Revise manuscripts, Manage projects.

Declaration of competing interest

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

Data availability

Data will be made available on request.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. This work was supported in part by the National Natural Science Foundation of China (Grant No. 62072281)

References

Abu-Mostafa, Y. S., & Atiya, A. F. (1996). Introduction to financial forecasting. *Applied Intelligence*, *6*(3), 205–213.
Aghabozorgi, S., Shirkhorshidi, A. S., & Wah, T. Y. (2015). Time-series clustering–a decade review. *Information Systems*, *53*, 16–38.
Akita, R., Yoshihara, A., Matsubara, T., & Uehara, K. (2016). Deep learning for stock prediction using numerical and textual information. In *2016 IEEE/ACIS 15th international conference on computer and information science* (pp. 1–6). IEEE.
Ang, J.-S., Ng, K.-W., & Chua, F.-F. (2020). Modeling time series data with deep learning: A review, analysis, evaluation and future trend. In *2020 8th international conference on information technology and multimedia* (pp. 32–37). IEEE.
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* (pp. 106–112). IEEE.
Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271.
Benesty, J., Chen, J., Huang, Y., & Cohen, I. (2009). Pearson correlation coefficient. In *Noise reduction in speech processing* (pp. 1–4). Springer.
Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.
Chen, X., Ma, X., Wang, H., Li, X., & Zhang, C. (2022). A hierarchical attention network for stock prediction based on attentive multi-view news learning. *Neurocomputing*, *504*, 1–15.
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, no. 1* (pp. 55–62).
De Fortuny, E. J., De Smedt, T., Martens, D., & Daelemans, W. (2014). Evaluating and understanding text-based stock price prediction models. *Information Processing & Management*, *50*(2), 426–441.
Deng, S., Zhang, N., Zhang, W., Chen, J., Pan, J. Z., & Chen, H. (2019). Knowledge-driven stock trend prediction and explanation via temporal convolutional network. In *Companion proceedings of the 2019 world wide web conference* (pp. 678–685).
Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, *25*(2), 383–417.
Farrell, J. L. (1974). Analyzing covariation of returns to determine homogeneous stock groupings. *Journal of Business*, *47*(2), 186–207.
Feng, F., He, X., Wang, X., Luo, C., Liu, Y., & Chua, T.-S. (2019). Temporal relational ranking for stock prediction. *ACM Transactions on Information Systems (TOIS)*, *37*(2), 1–30.
Gao, Q. (2016). *Stock market forecasting using recurrent neural network* (Ph.D. thesis), University of Missouri–Columbia.
He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.
Huang, Q., & Hain, T. (2021). Improving audio anomalies recognition using temporal convolutional attention networks. In *ICASSP 2021-2021 IEEE international conference on acoustics, speech and signal processing* (pp. 6473–6477). IEEE.
Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456). PMLR.
Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37.
Lara-Benítez, P., Carranza-García, M., & Riquelme, J. C. (2021). An experimental review on deep learning architectures for time series forecasting. *International Journal of Neural Systems*, *31*(03), Article 2130001.
Lee, D., & Seung, H. S. (2000). Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, *13*.
Li, Q., Chen, Y., Jiang, L. L., Li, P., & Chen, H. (2016). A tensor-based information framework for predicting the stock market. *ACM Transactions on Information Systems (TOIS)*, *34*(2), 1–30.
Li, L., Leng, S., Yang, J., & Yu, M. (2016). Stock market autoregressive dynamics: A multinational comparative study with quantile regression. *Mathematical Problems in Engineering*, *2016*.
Li, X., Wu, P., & Wang, W. (2020). Incorporating stock prices and news sentiments for stock market prediction: A case of Hong Kong. *Information Processing & Management*, *57*(5), Article 102212.
Liu, X., Guo, J., Wang, H., & Zhang, F. (2022). Prediction of stock market index based on ISSA-BP neural network. *Expert Systems with Applications*, Article 117604.
Liu, T., Ma, X., Li, S., Li, X., & Zhang, C. (2022). A stock price prediction method based on meta-learning and variational mode decomposition. *Knowledge-Based Systems*, *252*, Article 109324.

Liu, J., Yan, J., Wang, L., Huang, L., He, H., & Liu, H. (2021). Remote sensing time series classification based on self-attention mechanism and time sequence enhancement. *Remote Sensing*, *13*(9), 1804.

Liu, W., Yang, Z., Cao, Y., & Huo, J. (2022). Discovering the influences of the patent innovations on the stock market. *Information Processing & Management*, *59*(3), Article 102908.

Liu, M., Zeng, A., Xu, Z., Lai, Q., & Xu, Q. (2022). SCINet: time series modeling and forecasting with sample convolution and interaction. In *36th Conference on Neural Information Processing Systems (NeurIPS)*.

Ma, X., Karkus, P., Hsu, D., & Lee, W. S. (2020). Particle filter recurrent neural networks. In *Proceedings of the AAAI conference on artificial intelligence, Vol. 34, no. 04* (pp. 5101–5108).

Ma, X., Li, X., Zhou, Y., & Zhang, C. (2021). Image smoothing based on global sparsity decomposition and a variable parameter. *Computational Visual Media*, *7*(4), 483–497.

Pauca, V. P., Shahnaz, F., Berry, M. W., & Plemmons, R. J. (2004). Text mining using non-negative matrix factorizations. In *Proceedings of the 2004 SIAM international conference on data mining* (pp. 452–456). SIAM.

Qin, Y., Song, D., Cheng, H., Cheng, W., Jiang, G., & Cottrell, G. W. (2017). A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the 26th international joint conference on artificial intelligence* (pp. 2627–2633).

Rather, A. M., Agarwal, A., & Sastry, V. (2015). Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Systems with Applications*, *42*(6), 3234–3241.

Rathore, R. K., Mishra, D., Mehra, P. S., Pal, O., Hashim, A. S., Shapi'i, A., Ciano, T., & Shutaywi, M. (2022). Real-world model for bitcoin price prediction. *Information Processing & Management*, *59*(4), Article 102968.

Schumaker, R. P., & Chen, H. (2009). A quantitative stock prediction system based on financial news. *Information Processing & Management*, *45*(5), 571–583.

Singh, P., Kumar, B., & Pandey, A. (2010). Price and volatility spillovers across North American, European and Asian stock markets. *International Review of Financial Analysis*, *19*(1), 55–64.

Tsay, R. S. (2005). *Analysis of financial time series*. John wiley & sons.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.

Wang, J., Hu, Y., Jiang, T.-X., Tan, J., & Li, Q. (2023). Essential tensor learning for multimodal information-driven stock movement prediction. *Knowledge-Based Systems*, Article 110262.

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).