Expert Systems With Applications 236 (2024) 121204

Contents lists available at ScienceDirect

Expert Systems With Applications

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

VGC-GAN: A multi-graph convolution adversarial network for stock price prediction

Dongbo Ma, Da Yuan [*], Maojun Huang, Ling Dong

School of Computer Science and Technology, Shandong Technology and Business University, Yantai 264005, China

ARTICLE INFO

Keywords:
Stock price prediction
Graph Convolutional Network
Adversarial training
Variational Mode Decomposition

ABSTRACT

Not only market signals but also disturbances of related companies influence the stock volatility of a company. Currently, most approaches that utilize inter-stock correlations rely on predetermined graphs, such as industry graphs, shareholder graphs, or event graphs, necessitating supplementary financial data. Consequently, this approach often leads to limited generalization ability of the model and incomplete representation of inter-stock relationships. Thus, this study introduces VGC-GAN, a multi-graph convolutional adversarial framework, for predicting stock prices. Initially, the proposed model generates multiple correlation graphs by analyzing historical stock data, providing a comprehensive depiction of inter-stock correlations from diverse perspectives. Subsequently, a Generative Adversarial Network (GAN) framework, augmented with Mean Square Error (MSE) loss, is constructed to enhance the predictive performance of the model. The framework combines Multi-Graph Convolutional Network (Multi-GCN) and Gated Recurrent Unit (GRU) as the generator. It undergoes supervised and adversarial training with Convolutional Neural Network (CNN), facilitating the in-depth exploration of hidden correlations between stocks and time dependence of stocks. To mitigate the impact of noise, VGC-GAN uses subsequences after Variational Mode Decomposition (VMD) with optimized parameters as input to the generator. The proposed model is evaluated on several real datasets, and the experimental results confirm its effectiveness in stock price prediction.

1. Introduction

Stock prediction is a fundamental problem at the intersection of finance and computer science that has consistently captured the interest of researchers. The Efficient Market Hypothesis (Fama, 1970) posits that stock prices promptly, precisely, and comprehensively reflect all relevant information related to the present and future value of a company. However, stock prices are influenced by intricate non-linear relationships among economic variables, corporate policies, investor psychology, and political events (Enke & Thawornwong, 2005), which makes stock market prediction a highly challenging task.

Statistical methods typically assume that the stock data under investigation are generated by a linear process (Kumar & Murugan, 2013). With the advancement of artificial intelligence technology, machine learning methods capable of capturing non-linear and non-additive relationships in complex financial market data have been introduced and demonstrated to achieve superior prediction results (De Haan, Mercadier, & Zhou, 2016). Deep learning models have also been introduced to capture data feature representations more effectively in stock prediction (Ozbayoglu, Gudelek, & Sezer, 2020). For example, (Li, Zhu, Shen, & Angelova, 2023) proposed a clustering-enhanced deep learning framework with a new similarity measure to improve stock price prediction accuracy. Furthermore, Several studies have utilized Fuzzy Set Theory to model the stock market (Syu, Lin, Wu, & Ho, 2023; Wu, Syu, Lin, & Ho, 2021).

In addition, hybrid models combining different techniques have shown promising predictive performance. For example, Jothimani, Shankar, and Yadav (2016) utilized a hybrid approach that combined Empirical Modal Decomposition (EMD) and Artificial Neural Network (ANN) for short-term prediction of stock indices. Another hybrid model proposed by Liu, Ma, Li, Li, and Zhang (2022) integrated Variational Mode Decomposition (VMD) and meta-learning to enhance the accuracy of stock price prediction. Yang, Chen, and Liu (2023) proposed an improved PSO algorithm combined with neural network for stock price prediction, featuring adaptive adjustment of inertial weight to improve global and local search ability. Yan (2023) proposed a hybrid model combining AdaBoost feature selection and LSTM for predicting stock index futures prices, outperforming other popular prediction models based on performance metrics. Moreover, Kumar et al. (2022) proposed

[*] Corresponding author.
E-mail addresses: 2021410057@sdtbu.edu.cn (D. Ma), yuanda@sdtbu.edu.cn (D. Yuan), 2467447255@qq.com (M. Huang), 2020410021@sdtbu.edu.cn (L. Dong).

https://doi.org/10.1016/j.eswa.2023.121204
Received 14 April 2023; Received in revised form 5 August 2023; Accepted 12 August 2023
Available online 23 August 2023

a generalized Generative Adversarial Network (GAN) framework that employed Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) for adversarial training to predict stock trends.

Most contemporary stock prediction methods treat stocks as independent and identically distributed, disregarding relevant stock market information such as inter-stock correlations. However, the stock fluctuations of a company are influenced by both its market signals and interferences from related companies. Hence, the stock market can be naturally viewed through a graph-based perspective, where each listed company can be represented as a node, interconnected through intricate dependencies. Graph Convolutional Networks (GCNs) have been demonstrated to leverage and explore inter-node correlations while maintaining temporal relationships effectively. Chen, Wei, and Huang (2018) constructed relationship graphs based on shareholder relationships to enhance prediction accuracy. Li et al. (2021) integrated news data using GCNs to predict overnight stock movements. Chen, Jiang, Zhang, and Chen (2021) fitted stock relationships by computing Spearman's rank correlation coefficients among historical stock data. Yin et al. (2022) generated relationship graphs based on stock events for stock trend prediction.

Stock relationship graphs are commonly constructed based on shareholder relationships, industry relationships, or relevant news data among companies. Obtaining and processing additional financial information required by these methods can be complex and time-consuming. Moreover, these methods may only reflect specific types of predefined relationships, potentially introducing noise and bias that could impact the accuracy of the prediction results. Another approach for constructing stock relationship graphs is by utilizing correlation coefficients among historical stock data. However, existing studies often only consider the global simultaneous correlations of stocks, disregarding the local correlations and lead–lag relationships among stocks.

To address the above issues, this paper presents a novel approach VGC-GAN, which is a multi-graph convolutional adversarial network. The proposed model utilizes both correlation coefficients and Fast Dynamic Time Warping (FastDTW) of stock historical data to construct stock relationship graphs, capturing both synchronous and asynchronous relationships among stocks and eliminating the need for additional financial information. Compared to studies that rely on only a single correlation coefficient between historical data, the proposed model offers a more comprehensive understanding of the relationships between stocks, resulting in a more nuanced analysis of the stock market. To improve the robustness of existing models based on graph neural networks, the proposed model constructs a Wasserstein Generative Adversarial Network (WGAN) framework combined with Mean Square Error (MSE) loss. The generator utilizes a Multi-Graph Convolutional Network (Multi-GCN) to combine both stock-related information and historical information, along with a Gated Recurrent Unit (GRU) to capture the temporal dependence of stocks and implement stock prediction. The discriminator employs a Convolutional Neural Network (CNN) to evaluate the authenticity of generated stock prices. The framework utilizes a combination of adversarial training and supervised training to improve the accuracy of generated prediction values, enhancing the robustness of the model. The generator in the model utilizes input from more stable sub-sequences decomposed by VMD to handle the non-stationarity of data. The parameters of VMD are optimized using a combinatorial optimization algorithm based on Genetic Algorithm (GA) to obtain a parameter combination more suitable for the stock sequence.

In summary, our main contributions are outlined below:

- This paper presents a multi-graph convolutional adversarial network (VGC-GAN). The proposed model is designed to capture inter-stock correlation information from multiple perspectives and utilizes the strengths of adversarial and supervised training within a WGAN framework to enhance predictive performance and robustness.
- The proposed model employs a novel GA-based combinatorial optimization approach to optimize the parameters of VMD. Experimental results demonstrate that this method effectively decomposes historical price sequences of stocks, leading to improved performance.
- The validity of the proposed model is demonstrated through a comparative analysis with existing methods on three real stock price datasets (one from a subscription source and two from the public domain).

The rest of the paper is organized as follows. Section 2 briefly describes some related work on financial market prediction. Section 3 describes the technical details involved in our method. Section 4 describes the dataset and related settings used in the experiments. Section 5 shows the results of our experiments and discusses the results. Section 6 concludes and summarizes the work.

2. Related work

Stock market prediction has been extensively researched in the financial industry (Thakkar & Chaudhari, 2021). Previous approaches for stock prediction, such as Generalized Autoregressive Conditional Heteroskedasticity (GARCH) (Hyndman & Athanasopoulos, 2018) and Autoregressive Integrated Moving Average (ARIMA) (Bollerslev, 1986), relied on statistical linear prediction models. However, the inherently chaotic nature and instability of stock data limited their predictive accuracy. Consequently, machine learning methods have been introduced for stock market prediction, such as Support Vector Machines (SVM), decision trees, Artificial Neural Networks (ANN), and Random Forests (RF). For instance, Lee (2009) combined SVM and hybrid feature selection for trend prediction. Basak, Kar, Saha, Khaidem, and Dey (2019) used decision trees for stock trend prediction. Picasso, Merello, Ma, Oneto, and Cambria (2019) proposed a combination of RF, SVM, and ANN for stock trend prediction. And Parray, Khurana, Kumar, and Altalbe (2020) utilized SVM, ANN, and logistic regression for predicting stock trends on the second day.

In recent years, deep learning methods, such as CNN, LSTM, GRU, and attention mechanisms, have gained popularity in stock market prediction due to their ability to capture nonlinear and complex spatiotemporal dependencies. CNN is a deep learning model that automatically extracts temporal features from stock data, improving accuracy, stability, and mitigating overfitting (Hoseinzade & Haratizadeh, 2019). LSTM, a variant of RNN, addresses the gradient vanishing problem and is suitable for long time series data. Baek and Kim (2018) combined two LSTMs to prevent overfitting for stock prediction. Ding and Qin (2020) proposed a multi-input multi-output LSTM model for stock trend prediction. Chen, Wu, and Wu (2022) proposed a new hybrid method combining LSTM with K-Means for dynamic and static stock price prediction. GRU, a simplified alternative to LSTM, has a simpler structure with fewer parameters and faster training, and shows promising results on some datasets. Xu, Chai, Luo, and Li (2022) proposed a novel GRU model based on reinforcement learning to improve predictive performance. Ma, Karkus, Hsu, and Lee (2020) proposed PF-LSTM and PF-GRU, which combined the advantages of RNNs and particle filtering to have higher stock prediction accuracy than the original network. Attention mechanisms have also been combined with RNN, LSTM, and GRU to enhance stock trend prediction (Zhao, Zeng, Liang, Kang, & Liu, 2021).

Hybrid models that combine decomposition algorithms with deep learning techniques have been proposed to achieve better prediction results in stock market prediction. These models aim to reduce the modeling difficulty by decomposing the original sequence into different frequencies or more stable subsequences and then applying deep learning methods to these decomposed components. For example, Liang, Ge, Sun, He, and Chen (2019) proposed a stock price prediction model that combines wavelet transform with LSTM. Niu, Xu, and Wang (2020)

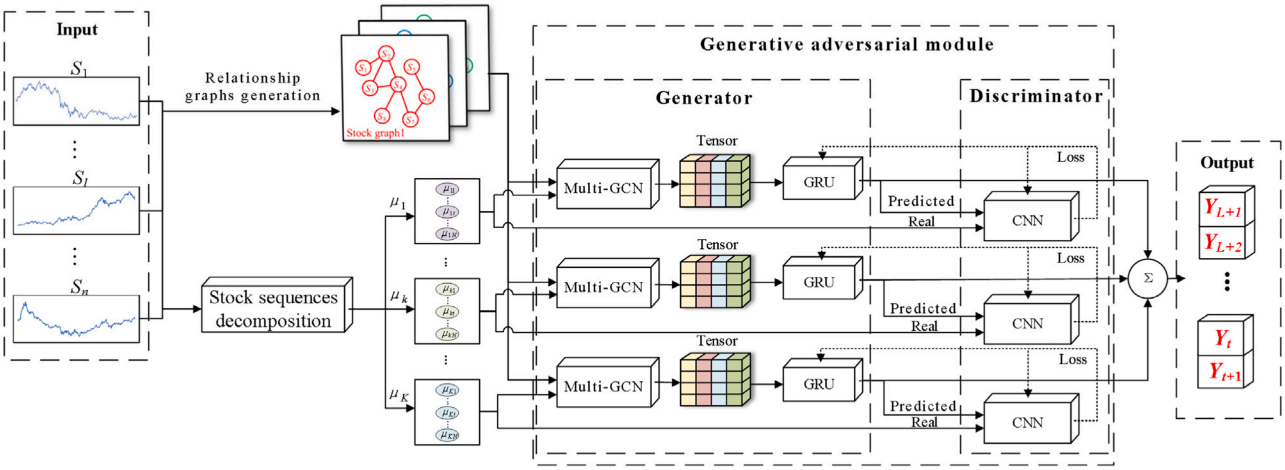


Fig. 1. The frame of the VGC-GAN model.

proposed a hybrid model that combines VMD with LSTM to improve the accuracy of stock market index prediction. Liu, Ma et al. (2022) combined VMD with meta-learning and changed the decomposition object to a window series to eliminate the look-ahead bias.

Generative Adversarial Networks (GANs), a type of deep learning model that consists of a generator and a discriminator trained in an adversarial manner, have also been utilized in stock market prediction to improve the training of generators and generate more accurate and realistic data. For instance, Kumar et al. (2022) proposed a general GAN framework that uses LSTM and CNN for adversarial training to predict stock trends. Yin, Han et al. (2021) proposed a multi-variate time series prediction method based on multi-attentive GANs, which combines GANs with attention mechanisms for improved prediction performance.

According to Wang et al. (2023), stock price movements are influenced not only by historical prices but also by related stocks. Multivariate time series prediction methods have been used to make predictions using inter-variate relationships by modeling individual variables separately (Madan & Mangipudi, 2018; Zhang, Zhang, Wang, Qin, & Wang, 2017). However, these methods may only partially leverage the potential dependency relationships among variables. Graph Neural Networks (GNNs) typically assume that the state of a node depends on the state of its neighbors, allowing them to capture potential spatial dependencies among nodes while preserving temporal relationships.

GNNs have been widely used to capture spatial dependencies in various fields, but their application to stock prediction has been relatively limited. For instance, Matsunaga, Suzumura, and Takahashi (2019) introduced GNNs into the stock market context, while (Chen et al., 2018) constructed a graph containing related company relationships to improve prediction accuracy. Chen et al. (2021) utilized GNNs to capture temporal features of stock data and proposed a CNN based on graph convolutional features for stock prediction. Additionally, Ye, Zhao, Ye, and Xu (2021) proposed a multi-graph convolution model called Multi-GCGRU, which extracts cross effects from predefined graphs. Wang, Li, Wang, and Zheng (2021) suggested the Hierarchical Adaptive Temporal-Relational (HTAR) model for stock prediction, and Xiang, Cheng, Shang, Zhang, and Liang (2022) proposed the Temporal and Heterogeneous Graph Neural Network (THGNN) to learn the dynamic relationships among stock price movements.

3. Model

3.1. Problem formulation

The stock prediction task can be formulated as a multivariate time series forecasting problem with auxiliary prior knowledge. This prior knowledge is encapsulated within a pre-defined weighted graph denoted as $\mathcal{G} = (V, E)$. Here, V is a set of $n = |V|$ vertices representing different stocks, while E comprises a set of edges that represent the strength of the relationship between vertices(*e.g.,* Pearson correlation coefficient and DTW between historical data).

The stock price time series is denoted as a feature tensor $X \in \mathbb{R}^{T\times N}$, where T denotes the length of sequence. At each time step t, the time series is denoted as $X_t \in \mathbb{R}^N$. Consequently, the stock prediction problem can be formulated as follows:

Given the graph $\mathcal{G} = (V, E)$ and observed stock closing price time series over a time window of length L, represented as $X_{(t-L):(t+1)}$, we aim to learn a function f that effectively maps $X_{(t-L):(t+1)}$ and the graph $\mathcal{G}$ to predict the next step closing price $\hat{X}^i_{t+1}$ of the target stock i. Specifically, $\hat{X}^i_{t+1}$ can be represented as follows:

$$\hat{X}^i_{t+1} = f(X_{(t-L):(t+1)}, \mathcal{G}) \tag{1}$$

where $X_{(t-L):(t+1)} = (X_{t-L}, X_{t-L+1}, \dots, X_t) \in \mathbb{R}^{L\times N}$ and $\hat{X}^i_{t+1} \in \mathbb{R}^1$.

3.2. Framework overview

The proposed model consists of three main parts: relationship graphs generation, stock sequence decomposition, and generative adversarial module, as illustrated in Fig. 1.

To evaluate the correlation between stocks, the relationship graphs generation part employs several methods, including global and local Pearson correlation coefficients, Spearman rank correlation coefficients, and FastDTW. By constructing synchronous and asynchronous correlation graphs, the relationships between stocks are effectively represented. Multiple correlation graphs constructed from different perspectives can provide a more comprehensive representation of the relationships between stocks.

In the stock sequences decomposition part, the VMD algorithm is utilized to decompose the historical data sequences of equities into multiple sub-sequences. This approach reduces the instability and complexity of stock price sequence. The parameters of VMD are optimized using a combinatorial optimization method based on genetic algorithm to achieve better decomposition effect.

The Generative adversarial module part is constructed using the WGAN framework combined with MSE loss. The generator includes Multi-GCN and GRU. The relationship graph created from the original sequence and the sub-sequences serving as node features serve as the input to Multi-GCN, which can achieve graph embedding and capture hidden correlation information between stocks. GRU is used to build temporal relationships within each node embedding. The discriminator employs CNN to determine the authenticity of the generated data.

The generator learns data representation and generates more realistic predictions through adversarial and supervised training. Finally, the predicted values of each sub-sequence generator are added to obtain the stock prediction value.

3.3. Stock sequences decomposition

The decomposition algorithm is effective in decomposing the complex stock historical data into multiple sub-series with distinct frequencies and trends. This process helps to reduce the instability and complexity of the sequences, leading to improved accuracy and stability in predictions. In our approach, each decomposed subsequence is modeled and predicted individually, and the predicted values are then aggregated to obtain the predicted value of the original stock sequences. Commonly used decomposition algorithms include wavelet transform, EMD, and VMD. VMD stands out due to its ability to adaptively adjust regularization parameters during the decomposition process, allowing it to better align with the sequence's features. As a result, VMD demonstrates superior performance and broader applicability when dealing with non-stationary sequences. Considering the high noise, dynamic, nonlinear, non-parametric, and chaotic nature of stock data, VMD algorithm is more suitable for the proposed model. And recent research (Jothimani & Başar, 2019) has demonstrated that the hybrid model based on VMD yields higher prediction accuracy compared to other decomposition methods.

3.3.1. Variational Mode Decomposition (VMD)

VMD, developed by Dragomiretskiy and Zosso (2013), is a fully non-recursive model that is capable of decomposing non-stationary sequences into several subsequences with different frequency scales and relatively stable characteristics. This decomposition process reduces the complexity of non-stationary sequences, allowing for more effective modeling. VMD decomposes the original signal or sequence data into a specified number of Intrinsic Mode Functions (IMFs) with specific bandwidths, achieved by constructing and solving a constrained variational problem. The VMD algorithm can be represented as a function $f_{vmd}()$, as outlined in the literature (Dragomiretskiy & Zosso, 2013).

$$\{u_k[t]|k=1,2,\dots,K;t=1,2,\dots,n\} = f_{vmd}\left(S[t],K,\alpha,\tau,\epsilon\right) \tag{2}$$

$$\sum_{k=1}^{K} u_k[t] = S'[t] \tag{3}$$

where $u_k[t]$ represents the kth mode of the signal at time script t, and n represents the length of the historical stock price sequences. $S[t]$, $S'[t]$, K, α, τ, and ϵ represent the original time sequence, the reconstructed signal, the number of modes, the penalty factor, the Lagrange multiplier, and the convergence tolerance level, respectively. Among these, K and α are positive integers, and τ is set to 0 or 1, as stated in the literature (Dragomiretskiy & Zosso, 2013).

Fig. 2 illustrates the decomposition results of VMD, where (a) represents the original stock historical price sequence, and (b)–(e) depict the four subsequences obtained after decomposition. It is evident from the figure that the subsequences exhibit higher stability and regularity compared to the original sequence.

3.3.2. The combinatorial optimization of VMD parameters

The optimal values of the mode number K and penalty factor α in VMD depend on the specific characteristics of the dataset and significantly impact the decomposition process. If K is set too large, it may result in increased computation time and over-decomposition, leading to higher complexity in prediction. Conversely, if K is set too small, the decomposed series may exhibit significant instability, failing to reduce the original sequence's instability effectively. Similarly, if α is set too large, some signals in the decomposed signal may be lost. On the other hand, if α is set too small, some components may contain signals from other components, leading to inaccurate decomposition results. Proper tuning of K and α is crucial to ensure proper and stable decomposition using VMD.

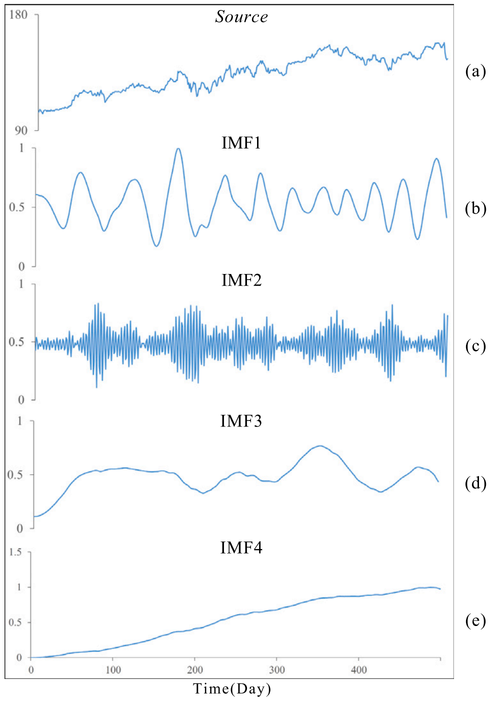


Fig. 2. Time series decomposition results.

GA (Holland, 1992) is an intelligent optimization algorithm known for its strong global nonlinear optimization capabilities. It can effectively consider the interaction between K and α and avoid falling into local optima. The objective function in GA is typically designed to minimize either the average envelope entropy E_p or the average residual E_r, which are calculated as follows, respectively:

$$p_s^k[t] = \frac{a_s^k[t]}{\sum_{t=1}^{n} a_s^k[t]} \tag{4}$$

$$E_p = \frac{1}{N}\sum_{s=1}^{N} -\frac{1}{K}\sum_{k=1}^{K}\sum_{t=1}^{n} p_s^k[t]\, log p_s^k[t] \tag{5}$$

$$E_r = \frac{1}{N}\sum_{s=1}^{N} \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(S_s[t]-S_s'[t]\right)^2} \tag{6}$$

where n represents the length of the sequences, N denotes the number of stocks in the dataset, K signifies the number of IMFs obtained through VMD decomposition, and $a_i^j[t]$ represents the envelope signal of the ith stock in the jth IMF at time stamp t, demodulated by Hilbert.

To achieve a more comprehensive decomposition of stock historical data sequences, the residual correlation theory was proposed by Jia, Zhang, Liu, and Gong (2021). According to this theory, when the original sequence is fully decomposed using VMD, the Pearson correlation coefficient $\rho_P\left(S[t], Res(t)\right)$ between the residuals $Res(t)$ and the original sequence $S[t]$ should be minimized, corresponding to the optimal parameter combination (K, α). Therefore, a combination of Genetic Algorithm (GA) and residual correlation theory is utilized, where the fitness function for GA is set as $\left\{\rho_P\left(S_{train}^{sn}[t], Res(t)\right) | sn = 1,2,\dots,N\right\}$, with $S_{train}^{sn}[t]$ denoting the training set of the snth original sequence, $size\left(S_{train}^{sn}[t]\right)$ denoting the length of the training set of the original sequence, and N representing the number of stocks in the dataset. The parameter settings for GA are $K \in \{2,3,\dots,10\}$ and $\alpha \in \left[\frac{size\left(S_{train}^{sn}[t]\right)}{2}, 3 \times size\left(S_{train}^{sn}[t]\right)\right]$. The model can be expressed as follows:

$$minLoss_{VMD} = \frac{\sum_{sn=1}^{N} \rho_P \left(S_{train}^{sn}[t], Res(t)\right)}{N}$$

$$s.t. \begin{cases} K \in \{2, 3, \dots, 10\} \\ \alpha \in \left[\frac{Size(S_{train}^{sn}[t])}{2}, \ 3 \times size\left(S_{train}^{sn}[t]\right)\right] \end{cases}$$

3.4. Relationship graphs generation

When considering stock prediction using only individual stock information, the potential impact of correlations between stocks on the target stock may be overlooked. However, a more comprehensive and intuitive representation of the relationships among stocks can be obtained by treating the stock market as a graph, where each listed company is represented as a node, and the nodes are interconnected by hidden dependencies. This approach allows for a higher degree of interpretability in the model, as the stock relationship graphs can visually depict the interconnectedness and interdependencies among stocks, providing valuable insights for stock prediction tasks.

Studies have revealed that companies can have multiple types of relationships, such as being in the same industry, having common shareholders, or having relationships between suppliers and customers, as well as between shareholders and investee companies (Chen et al., 2018; Yin et al., 2022). Constructing relationship graphs through human tagging or NLP techniques may face challenges such as difficult access to information or the need for additional knowledge from potentially ambiguous textual news sources. On the other hand, generating relationships directly based on market signals can be easier to implement and has shown to be effective in practical applications (Chen et al., 2021; Xiang et al., 2022).

Nevertheless, existing studies on stock prediction based on market signals often focus on mining simultaneous correlations in historical stock data while overlooking the lead–lag relationships among stocks. This limitation prevents the model from fully capturing the impact of stock market information on stock prices, as the temporal dependencies and dynamics among stocks are not fully accounted for. Incorporating lead–lag relationships among stocks in constructing relationship graphs can provide a more comprehensive and accurate representation of stock market dynamics, leading to improved stock prediction performance.

Hence, in this paper, both synchronous and asynchronous correlation matrices are calculated based on historical market signals of the stocks. Pearson correlation coefficient, Spearman's Rank correlation coefficient, and FastDTW are employed to obtain these correlation matrices. The value of each element in the correlation matrix is used to determine the strength of the inter-company relationship. Specifically, for each type of relationship $r \in \mathcal{R}$, an undirected graph $\mathcal{G}_r = (V, E)$ is constructed, where each node $v \in V$ corresponds to a stock. Graph $\mathcal{G}_r$ is constructed using the adjacency matrix $A_r = (a_{ij})_{\mathcal{V}\times\mathcal{V}}$, where the element a_{ij} in A_r represents the strength of the relationship between stock i and stock j.

To investigate synchronous correlations among stocks, Pearson correlation coefficient graph $\mathcal{G}_P$ and Spearman's Rank correlation coefficient graph $\mathcal{G}_{SP}$ are constructed. The Pearson correlation coefficient is a statistical measure that quantifies the linear relationship between two sequences of historical stock prices. On the other hand, Spearman's Rank correlation coefficient assesses the strength of the monotonic relationship among the stock price series. To calculate Spearman's Rank correlation coefficient, each historical price sequence of stocks is sorted in ascending order and assigned a rank. For a sample size of n, the Pearson correlation coefficient and Spearman's Rank correlation coefficient are calculated as follows:

$$\rho_P(X, Y) = \frac{E\left[\left(X - \mu_X\right)\left(Y - \mu_Y\right)\right]}{\sqrt{\sum_{i=1}^{n}\left(X_i - \mu_X\right)^2 \sum_{i=1}^{n}\left(Y_i - \mu_Y\right)^2}} \tag{7}$$

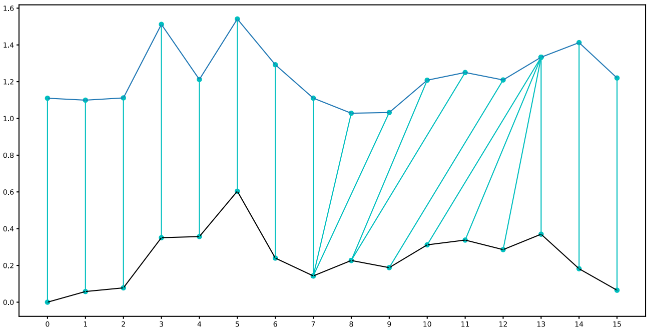


Fig. 3. Time series decomposition results.

$$\rho_{Sp}(X, Y) = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2 - 1)} \tag{8}$$

where μ represents the mean value and d_i denotes the rank difference between X_i and Y_i.

The FastDTW graph $\mathcal{G}_{DTW}$ is constructed to integrate the lead–lag relationship among stocks into the stock relationship. DTW is an algorithm that calculates the similarity of two time series by warping the time sequences to find the optimal alignment between them. It uses a dynamic programming approach to perform the calculation of time regularization. As shown in Fig. 3, each point in one time series is not directly aligned with the corresponding point in the other time series but is warped to a point in the other time series. The FastDTW algorithm captures the asynchronous correlation of stocks due to the lead–lag relationship, thus refining the correlation among stocks.

As per the literature (Salvador & Chan, 2007), the FastDTW algorithm is used as an approximate algorithm of DTW due to the time complexity of DTW being $O(n^2)$, which is not suitable for large-scale stock data. FastDTW offers a reduced time complexity of $O(n)$, making it more feasible for analyzing large-scale stock data. The three key operations of the FastDTW algorithm, as mentioned in the literature (Salvador & Chan, 2007), are:

(1) Coarse Granularity: The size of a time sequence is reduced by averaging adjacent point pairs to represent the same curve as accurately as possible with fewer data points.

(2) Projection: The DTW algorithm is run on time sequences at a coarser granularity.

(3) Fine granularity: The aggregation path obtained at a coarser granularity is further refined on a finer-grained time series.

To combine with the rest of the correlation graphs, the results are scaled between 0 and 1. Considering the entire FastDTW algorithm as a function $f_{DTW}()$, the formula for calculating the edge of the graph $\mathcal{G}_{DTW}$ for a sample size of n is as follows:

$$\rho_{DTW}(X, Y) = 1 - \frac{f_{DTW}(X, Y)}{n} \tag{9}$$

Considering that some relationships among stocks are unstable and time-sensitive, the global and local correlation of stocks are combined. In addition to constructing the global correlation coefficient graph $\mathcal{G}_P$, $\mathcal{G}_{SP}$, and $\mathcal{G}_{DTW}$, the training is also divided into several windows of length T_w by sliding windows. The average correlation coefficients $\overline{\rho_P}$, $\overline{\rho_{SP}}$, and $\overline{\rho_{DTW}}$ of all windows are then calculated to construct local correlation coefficient graphs $\mathcal{G}_{wp}$, $\mathcal{G}_{wSP}$, and $\mathcal{G}_{wDTW}$.

3.5. Generative adversarial module

Generative Adversarial Networks (GANs) are utilized in our model, employing the concept of adversarial training with two competing neural networks: a generator and a discriminator. The generator is responsible for capturing relevant information and temporal dependence on the stocks and uses it to predict future stock prices. The

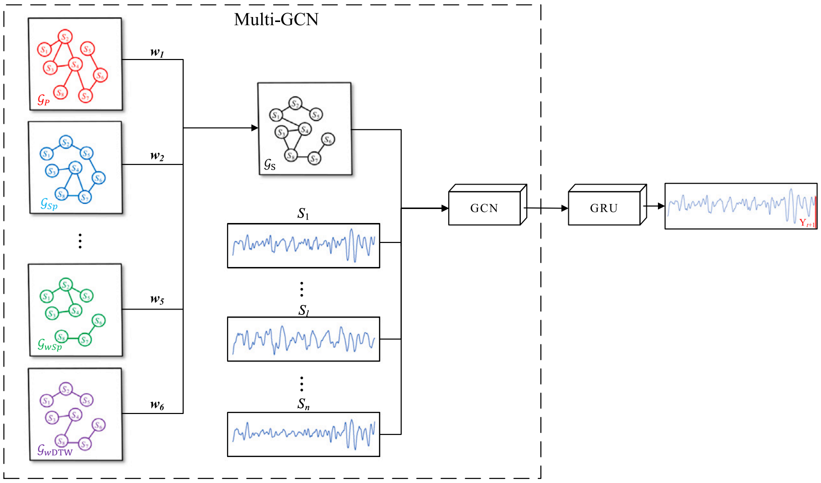


Fig. 4. Schematic diagram of the generator.

predicted stock prices are combined with the input data to construct fake samples, while the input data is combined with the true stock prices to construct real samples. These samples are then fed into the discriminator, which aims to distinguish between real and fake samples, and updates the generator's gradient through backpropagation. The generator and discriminator are trained alternately and iteratively, engaging in the process of mutual competition until they reach Nash equilibrium, where the generator produces realistic and high-quality stock price predictions. This adversarial training approach allows the generator to learn and improve its predictions over time, resulting in more accurate and realistic future stock price predictions.

3.5.1. Generators

To capture potential correlations and dependencies among stocks, the generator of the proposed model is designed to capture the complex local and global temporal dependencies and predict future stock prices, as shown in Fig. 4. The generator includes a Multi-Graph Convolutional Network (Multi-GCN) for capturing hidden correlation information among stocks and a GRU for establishing temporal relationships of stock data. The combination of Multi-GCN and GRU allows the generator to effectively capture both the local and global correlations among stocks, as well as the temporal dependencies in the stock data.

(1) Multi-graph convolutional network

To capture the intricate interconnections among stocks, the locally and globally generated relationship graphs are combined into a synthesis graph denoted as $\mathcal{G}_S$. In this figure, the adjacency matrix is constructed using the following approach:

$$A_S = \sum_{kn=1}^{K} w_{kn} A_{kn} \tag{10}$$

where the adjacency matrices of K generated relationship graphs are denoted by $\left\{A_{kn} | kn = 1, 2, \ldots, K\right\}$, $w_{kn} \in \mathbb{R}^{n\times n}$ represent the trainable parameters. In the proposed model, the number of generated relationship graphs is 6, *i.e.* $K = 6$.

Since the impact of different relationships on stock prediction varies and is challenging to be determined manually, the weights of each relationship are learned from the data.

GCNs exhibit robust learning capabilities for graph representation, making them suitable for graph embedding by integrating node topology and attributes. In this subsection, GCNs are employed to fuse stock-related data with historical stock information, capturing intricate interactions among related stocks and generating novel features for each stock that incorporate cross-influence information from other stocks. To streamline the graph convolution, a local first-order approximation is employed, and each layer is computed as follows:

$$H^{l+1} = ReLU\left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{l}W^{l}\right) \tag{11}$$

where $\widetilde{A} = A + I_n$, and the diagonal array $\widetilde{D}$ is the degree matrix with $\widetilde{D}_{ii} = \sum_j \widetilde{A}_{ij}$. H^l denotes the node vector of the lth layer, where H^0 represents the input matrix, and W^l is the trainable weight matrix of the corresponding layer.

$\hat{A}$ is defined as $\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}$. Accordingly, each layer of the graph convolution is given by:

$$H^{l+1} = ReLU\left(\hat{A}H^{l}W^{l}\right) \tag{12}$$

After graph embedding, each node will have its unique node embeddings. These node embeddings can be utilized to establish temporal relationships among stocks for stock price prediction. By leveraging the learned node embeddings, the model can capture the dynamics and interactions among stocks over time, allowing for more accurate stock price prediction based on temporal dependencies.

(2) Gated Recurrent Unit

The consideration of temporal dependence in historical stock data is crucial for accurate stock price prediction. Gated Recurrent Units (GRUs) (Chung, Gulcehre, Cho, & Bengio, 2014) are an advanced type of recurrent neural networks (RNNs) that possess the memory property of RNNs and are effective in mitigating the issue of gradient explosions. Therefore, after obtaining node embeddings that incorporate stock-related information, GRUs are utilized to enhance the model's ability to capture local and global temporal correlations among nodes. The computation of a GRU unit is outlined as follows:

$$r_t = \sigma\left(W_r \cdot x_t + U_r \cdot h_{t-1} + b_r\right) \tag{13}$$

$$z_t = \sigma\left(W_z \cdot x_t + U_z \cdot h_{t-1} + b_z\right) \tag{14}$$

$$\widetilde{h}_t = tanh\left(W_h \cdot x_t + r_t \circ (U_h \cdot h_{t-1}) + b_h\right) \tag{15}$$

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \widetilde{h}_t \tag{16}$$

where σ represents the logistic sigmoid function, while r_t and z_t refer to the reset gate and update gate, respectively. h_{t-1} and x_t denote the hidden layer state and the node embedding of the input at time t, respectively. $\widetilde{h}_t$ corresponds to the candidate values for storing the input values, and $\circ$ denotes the element-wise (Hadamard) product. W and U represent the weight matrices, while b denotes the bias vector.

3.5.2. Discriminator

The discriminator in the proposed model takes two sets of samples as input: real samples and fake samples. Its main objective is to distinguish between real and fake samples. The real sample consists of L data used as input, along with target data. On the other hand, the fake

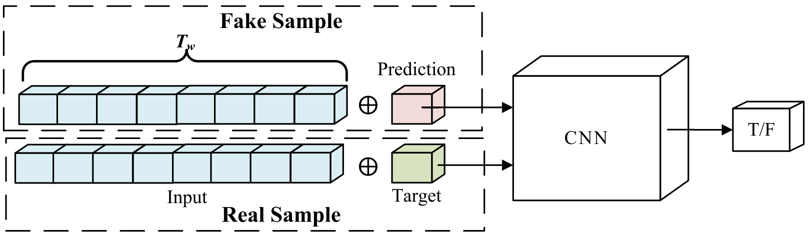


Fig. 5. Discriminator architecture diagram.

sample is a combination of L data used as input and the predicted value generated by the generator. The architecture of the discriminator, as illustrated in Fig. 5, includes a Convolutional Neural Network (CNN) that models latent relationships and classifies the inputs as true or false. The output of the discriminator represents the degree of falsity, with smaller values indicating a more "true" input. The gradient of the discriminator is then backpropagated to the generator to improve the accuracy of the generated predictions.

3.5.3. *Adversarial training*

GANs have gained significant attention since their introduction. By employing iterative training of the generator and discriminator, GANs can enhance the training process and improve the performance of the generator. However, GANs often suffer from challenges such as unstable training and mode collapse, where the generator fails to generate diverse samples. Wasserstein GANs (WGANs) utilize the Wasserstein distance as the loss function, resulting in more stable training. Nonetheless, WGANs may still encounter convergence difficulties. Therefore, WGAN-GP, proposed by Gulrajani, Ahmed, Arjovsky, Dumoulin, and Courville (2017), presents a modification to WGANs that replaces weight clipping with gradient penalty to achieve improved training stability and generate higher-quality samples. The objective function of WGAN-GP is expressed as follows:

$$\begin{aligned} L =& \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}\left[D\left(\tilde{x}\right)\right] - \mathbb{E}_{x\sim\mathbb{P}_r}\left[D\left(x\right)\right] \\ &+ \lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[\left(\|\nabla_{\hat{x}}D\left(\hat{x}\right)\|_2 - 1\right)^2\right] \end{aligned} \tag{17}$$

where $\mathbb{P}_g$ represents the distribution of generated stock prices, $\mathbb{P}_r$ denotes the distribution of true stock prices, and $\mathbb{P}_{\hat{x}}$ signifies the distribution of added noise. The value of $\hat{x}$ is computed as $\hat{x} = \epsilon\tilde{x} + (1-\epsilon)\,x, \epsilon \sim U\,[0,1]$, where ϵ is drawn from a uniform distribution on the interval [0,1]. The parameter λ is used to adjust the significance of the gradient penalty, with a default value of 10.

The non-linear and chaotic characteristics of stock prices necessitate the minimization of specific loss functions without compromising performance. In the literature (Khaled, Elsir, & Shen, 2022), an adversarial training approach combining adversarial loss and Mean Squared Error (MSE) loss for traffic flow prediction was proposed, which improved the learning efficiency of the model. However, this approach did not address the issue of unstable GAN training. To mitigate this, we combine WGAN-GP with MSE loss to enhance the learning efficiency and prediction accuracy of the model through adversarial and supervised training. The MSE loss (Eq. (18)) is added to the generator's loss (Eq. (19)) to encourage the generated stock prices to be closer to real stock prices. The formulated loss function is as follows:

$$Loss_M = \sqrt{\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g, x\sim\mathbb{P}_r}\left[\|x-\tilde{x}\|^2\right]} \tag{18}$$

$$Loss_G = -\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}\left[D\left(\tilde{x}\right)\right] + k\,Loss_M \tag{19}$$

$$\begin{aligned} Loss_D =& \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}\left[D\left(\tilde{x}\right)\right] - \mathbb{E}_{x\sim\mathbb{P}_r}\left[D\left(x\right)\right] \\ &+ \lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[\left(\|\nabla_{\hat{x}}D\left(\hat{x}\right)\|_2 - 1\right)^2\right] \end{aligned} \tag{20}$$

where k represents a parameter that adjusts the significance of the MSE loss and adversarial loss.

4. Experiments

This section presents the experimental settings for the real dataset experiments. In particular, Section 4.1 provides detailed information on the three datasets used, while Section 4.2 describes the experimental setup in depth. Furthermore, Section 4.3 presents relevant information on several competing models.

4.1. *Dataset description*

The predictive performance of the model has been validated in various markets, as demonstrated in the datasets of Liu, Ma et al. (2022), Yin, Yan, Almudaifer, Yan, and Zhou (2021). These datasets include:

(a) The Exchange-Traded Fund(ETF) dataset includes the daily stock price averages of the top 50 exchange-traded funds from November 22nd, 2010 to June 04th, 2020, and it can be subscribed from Kibot.

(b) The Shanghai Stock Exchange (SSE) dataset Includes daily closing prices of SSE 50 stocks in the China market from January 2nd, 2018, to December 31st, 2020. It can be obtained through Tushare.

(c) The Dow Jones Industrial Average (DJIA) dataset includes daily closing prices of dow jones industrial average stocks in the U.S. market from January 2nd, 2018, to December 31st, 2020. It can be obtained from Yahoo Finance.

The daily closing prices of the three datasets (ETF, SSE, DJIA) are used for the experiment. Stocks with incomplete price values and those with outliers are not included in the analysis to ensure data integrity and accuracy of the results. All the datasets are divided into training set, validation set, and test set according to the ratio of 7:1:2. The specific datasets are divided as shown in Table 1.

4.2. *Experimental setup*

4.2.1. *Data preprocessing*

The Min–Max normalization is applied to the datasets to reduce the impact of noise. The normalization formula is as follows:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \times (D - C) + C \tag{21}$$

where C and D are the lower and upper bounds of the predefined range, respectively, with C set to 0 and D set to 1.

Existing studies often decompose or normalize the training and test sets as a whole, which can improve the model's performance but may also reduce its credibility due to the risk of data leakage. To address this issue, we adopt a different approach. We first decompose and normalize the training set data and then apply the same parameters to decompose and normalize the test set data, ensuring the normalization is consistent with the training set. The results are also back-normalized to ensure the reliability of the resulting evaluation metrics.

Table 1
The characteristics of the datasets.

Dataset	Total	Train	Vaild	Test	Stock numbers
ETF	2854	1998	285	571	45
SSE	730	511	73	146	41
DJIA	756	529	76	151	29

4.2.2. Parameter setting

After conducting several trials, the generator and discriminator are trained using the RMS optimizer with the following parameters: learning rate of 0.008, epoch of 100, batch size of 128, sliding window length (L) of 15, local correlation window length (T_w) of 30, and a threshold value of the generative graph (ε) set to 0.9 (where nodes i and j are connected if $A(i,j) > \varepsilon$). The MSE loss factor (k) is set to 3. The study of parameters k and T_w is presented in Section 5.3.

The parameters K and α of the Variational Mode Decomposition (VMD) are calculated using Genetic Algorithm (GA), while the Lagrange multiplier (τ) is set to 0, and the convergence tolerance level (ε) is set to 1e-7. The initial distribution of the proportion parameters $\{w_1, w_2, \dots, w_6\}$ of each relationship graph is set to uniform. The parameters of K and α are studied in Section 5.2.1.

4.2.3. Evaluation metrics

The popular evaluation metrics, including Accuracy, Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE), are utilized to assess the predictive performance of the model. The formulas for these metrics are as follows:

$$Accuracy = \frac{n_{correct}}{n} \tag{22}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{23}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{24}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{25}$$

where y_i represents the real stock price for the ith stock, $\hat{y}_i$ represents the predicted stock price for the ith stock, n represents the total number of samples, and $n_{correct}$ represents the number of samples with the correct predicted trend.

Higher accuracy and smaller RMSE, MAE, and MAPE values indicate higher prediction effectiveness.

4.3. Baseline methods

To assess the predictive performance of the proposed method, a comparison is made with other existing methods, including GRU, GCGRU, SCINet, StemGNN, and PF-LSTM.

- **GRU** (Chung et al., 2014): GRUs are a variant of RNNs that employ a gating mechanism, enabling effective handling of long-term dependencies and mitigating the issue of gradient vanishing or exploding. GRUs have been widely used as a common approach for stock prediction.
- **GCGRU** (Yin, Yan et al., 2021): GCGRU is a hybrid approach that combines GCNs and GRUs. It constructs a stock relationship graph by calculating Pearson correlation coefficients from historical stock data and then extracts correlation features of stocks using GCNs. These correlation features are then integrated into the stock prediction model based on GRU.
- **SCINet** (Liu, Zeng et al., 2022): SCINet is a hierarchical downsampling-convolution-interaction structure that utilizes a diverse set of convolutional filters. It iteratively extracts and exchanges information at multiple temporal resolutions, enabling the model to learn efficient representations with enhanced predictability.
- **StemGNN** (Cao et al., 2020): StemGNN is an end-to-end framework that combines the Graph Fourier Transform (GFT) for modeling inter-series correlations and the Discrete Fourier Transform (DFT) for modeling temporal dependence. It can automatically learn inter-series correlations from the data without relying on predefined priors.
- **PF-LSTM** (Ma et al., 2020): PF-RNNs combine the strengths of both RNNs and particle filtering techniques. It learns potential particle belief representations and updates beliefs using particle filters, which can enhance the model's prediction accuracy. In this study, PF-LSTM is used as a natural alternative to PF-RNN to evaluate its performance in stock prediction.
- **DLinear** (Zeng, Chen, Zhang, & Xu, 2023): DLinear decomposes the time series into a trend series and a remainder series, and then uses two single-layer linear networks to model these two series for prediction tasks.
- **FEDformer** (Zhou et al., 2022):FEDformer combines Transformer and seasonal trend decomposition methods for time series analysis with a linear time complexity, making it faster and more efficient than the Transformer.
- **N-hits** (Challu et al., 2022): N-HiTS is an extension of the N-BEATS model that enhances both prediction accuracy and computational efficiency. This is achieved by introducing novel techniques such as layered interpolation and multi-rate data sampling.
- **WaveForM** (Yang et al., 2022): WaveForM is an end-to-end framework for multivariable time series forecasting that combines Discrete Wavelet Transform with graph construction. It uses a graph-enhanced prediction module that utilizes dilation convolution and graph convolution to capture correlations and predict wavelet coefficients at different levels.
- **PatchTST** (Nie, H. Nguyen, Sinthong, & Kalagnanam, 2023): PatchTST is an efficient Transformer-based model for multivariate time series forecasting and self-supervised representation learning. It offers three key benefits: embedding retains local semantic information, attention maps have reduced computation and memory usage, and the model can attend to longer history.

5. Results and discussion

In this section, the effectiveness of the proposed method is verified by comparing it with various benchmark models on real stock datasets. Additionally, ablation studies are conducted to analyze the impact of each component of the model on the prediction performance. Furthermore, an analysis of the hyperparameters is performed to understand their influence on the model's performance.

5.1. Computational performance evaluation

To evaluate the computational performance of the proposed VGC-GAN model, its performance was compared against baseline methods using real datasets. Table 2 presents the average prediction performance of all models on these datasets, with the best performance in each column indicated in bold. Table 3 also reports the number of trainable parameters, average training time, and testing time of our model with baseline methods to analyze its complexity.

All methods, except for DLinear, FEDformer, N-HiTS, and PatchTST, have a sliding window length (L) of 15, predict a length of 1, the epoch size is set to 100, and the learning rate is 8e-4. For DLinear and Former, the epoch size and learning rate are set to 10 and 0.01, respectively.

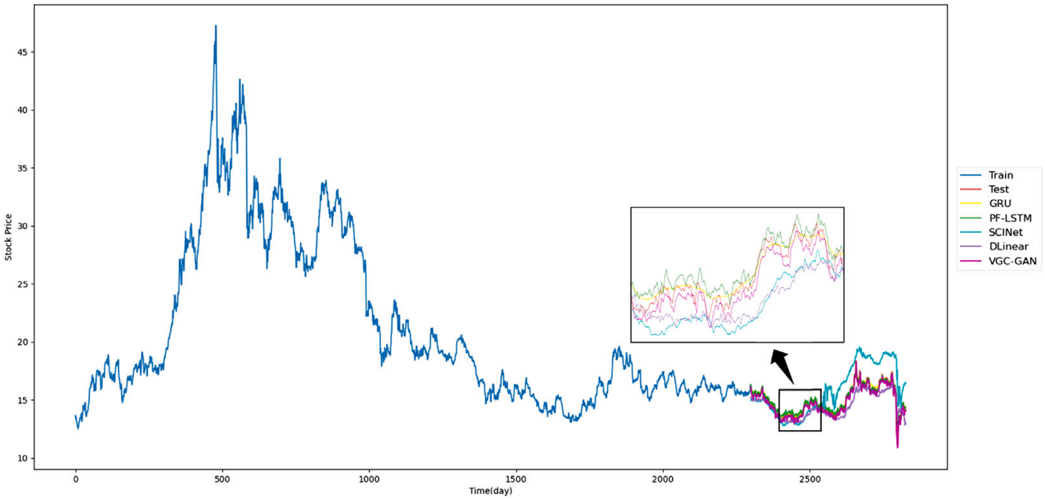


Fig. 6. Predicted and real values of different methods on stock CGO.

Table 2
The experimental results compared with baseline methods. Results with **bold** are the best performance.

Method	ETF				SSE				DJIA			
	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE
GRU	0.4838	9.1341	7.1865	6.085%	0.4876	0.8268	0.6643	4.049%	0.5029	22.3577	20.7333	9.600%
PF-LSTM	0.5206	8.0971	6.6303	**5.749%**	0.5646	0.8260	0.6996	4.981%	0.5323	9.8449	8.6189	5.342%
GCGRU	0.4837	25.3827	23.1745	21.889%	0.5218	4.4529	4.2846	14.571%	0.4802	92.3730	91.4799	52.792%
SCINet	0.4890	17.9661	16.7680	17.340%	0.4850	1.0059	0.8296	6.273%	0.4982	39.6562	38.3875	26.437%
StemGNN	0.4697	15.4722	13.5813	14.397%	0.5164	1.9070	1.6225	9.815%	0.4989	22.7978	20.9550	13.824%
DLinear	0.4879	7.6692	6.7577	6.652%	0.4910	0.6935	0.5459	3.811%	0.4938	6.2948	5.1648	3.564%
FEDformer	0.4980	8.0715	5.5172	9.435%	0.4996	0.8375	0.6704	5.651%	0.5170	6.0464	4.6955	3.643%
N-HiTS	0.5039	15.3731	11.9914	14.198%	0.5145	2.6274	2.1332	14.429%	0.5129	22.5090	18.2472	12.367%
WaveForm	0.4911	17.1118	15.1484	19.304%	0.5076	1.4570	1.2439	8.618%	0.5073	46.7579	45.2509	53.601%
PatchTST	0.5019	15.0374	11.4036	12.186%	0.5245	2.7224	2.3130	16.358%	0.4977	39.6321	34.1735	22.835%
VGC-GAN	**0.5285**	**5.9776**	**4.9362**	5.865%	**0.5673**	**0.4380**	**0.3601**	**2.415%**	**0.5712**	**4.0661**	**3.3558**	**2.285%**

Table 3
Complexity analysis of proposed method versus baseline methods.

Method	Complexity Analysis		
	Params	Training(s)	Testing(s)
GRU	74081	58.04	0.20
PF-LSTM	135891	511.19	0.35
GCGRU	98945	22.80	0.08
SCINet	518800	1816.34	2.23
StemGNN	1661212	1518.97	2.39
DLinear	1516721	1.11	0.27
FEDformer	11761728	336.83	2.28
N-HiTS	3244651	1.79	0.06
WaveForm	1553774	2495.33	4.62
PatchTST	1077138	9.54	0.14
VGC-GAN	235366	74.46	0.11

N-HiTS and PatchTST have an epoch size of 50, and the learning rate is set to its default. Based on the findings from Tables 2 and 3, the following observations and conclusions can be drawn:
(1)GRU, GCGRU, DLinear, N-HiTS, PatchTST, and the proposed VGC-GAN demonstrate high computational efficiency, while PF-LSTM, SCINet, StemGNN, FEDformer, and WaveForm have low computational efficiency.
(2)The proposed model demonstrates superior prediction performance compared to the baseline methods across all three datasets, indicating its robustness. Specifically, the proposed model achieves optimal values for Accuracy, RMSE, and MAE on all three datasets, with the MAPE on the ETF dataset being slightly higher but not significantly deviating from the optimal value.

Figs. 6, 7, and 8 depict the prediction graphs of individual stocks on different datasets for some of the better-performing methods. Specifically, Fig. 6 shows the ETF dataset, Fig. 7 shows the SSE dataset, and Fig. 8 shows the DJIA dataset. These figures demonstrate that the proposed model can accurately and efficiently predict stock prices, with its predicted value lines closely aligning with the lines of the original values.

5.2. Ablation studies

To analyze the impact of different modules on prediction performance, extensive validations were conducted on all datasets using various variants of the proposed model.

5.2.1. Impact of different VMD parameter optimization methods

In Table 4, Envelope entropy*, Residual*, Pearson*, and GA-Envelope entropy ,GA-Residual, and GA-Pearson represent different VMD parameter optimization methods. And in Tables 4 and 5, the asterisk (*) denotes that the parameters of the model were selected using the same approach as described in the literature (Jia et al., 2021; Liu, Ma et al., 2022). Specifically, α was set to 2000, and the optimal value of K was determined using an exhaustive method from the range $K \in \{2, 3, \dots, 10\}$.

Fig. 9 displays the values of the objective function for different values of K with $\alpha = 2000$ on the SSE dataset. The red dots represent the optimal values obtained, while the dashed lines represent the optimal values calculated using GA. Based on the results presented in Tables 4, 5, and Fig. 9, the following observations and conclusions can be drawn:

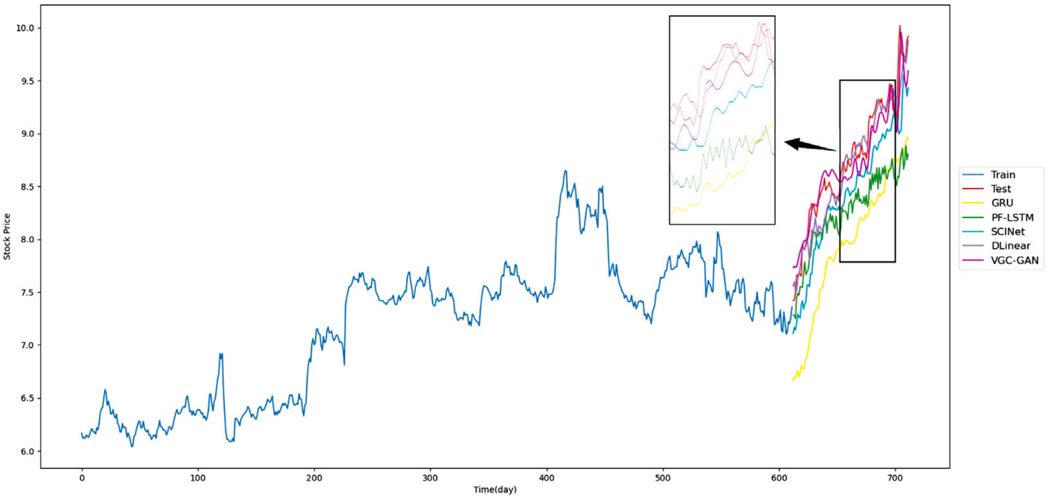


Fig. 7. Predicted and real values of different methods on stock 600,015.

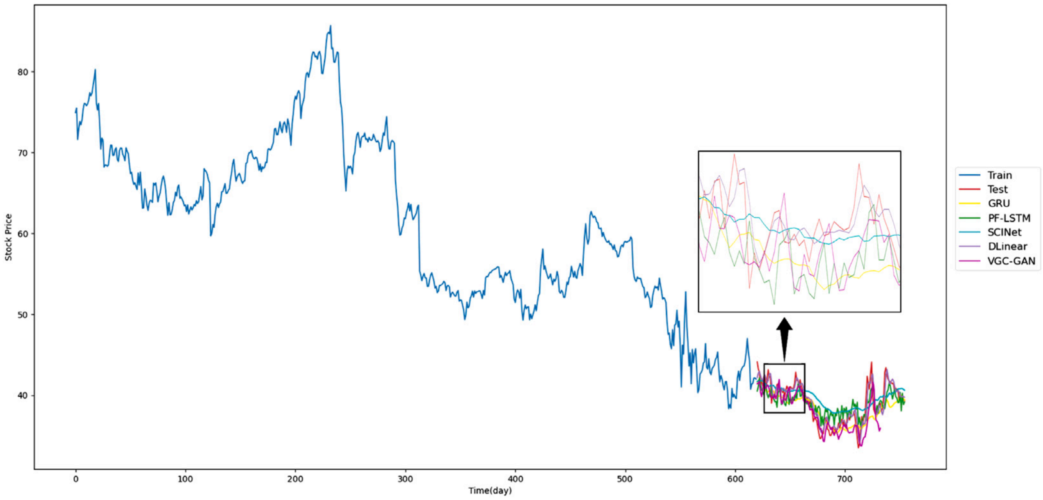


Fig. 8. Predicted and real values of different methods on stock WBA.

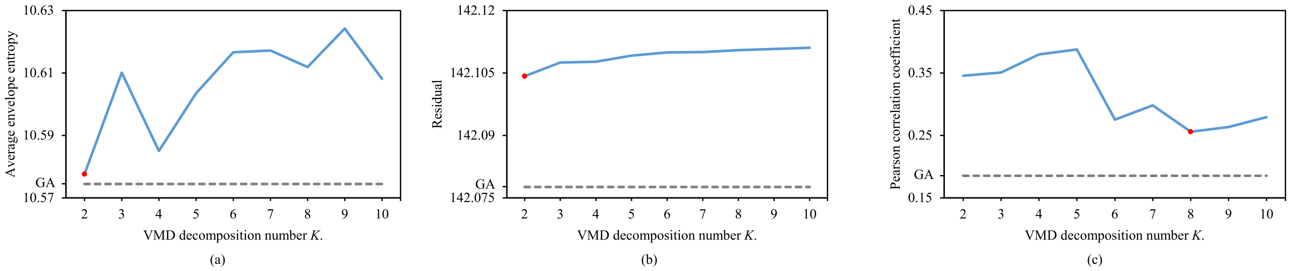
Table 4
The prediction performance of different VMD parameter optimization methods and other decomposition algorithms.

Optimization method	ETF				SSE				DJIA			
	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE
EMD	0.5143	7.0029	6.2316	7.106%	0.5448	0.5732	0.4990	3.298%	0.5510	4.2960	3.7113	2.520%
EEMD	0.5188	7.1680	6.3849	7.864%	0.5352	**0.3796**	**0.3336**	2.922%	0.5489	4.2017	3.7439	2.454%
CEEMDAN	0.5114	6.6005	5.8001	7.605%	0.5478	0.6207	0.5424	3.349%	0.5401	4.5671	4.0300	2.609%
SSA	0.5114	7.2526	6.5742	8.346%	0.5180	0.5942	0.5086	3.285%	0.5112	4.6329	4.0652	2.637%
Envelope entropy*	0.5161	8.7087	7.8157	8.669%	0.5210	0.6773	0.5855	4.925%	0.5553	4.5822	3.9336	2.600%
Residual*	0.5161	8.7087	7.8157	8.669%	0.5543	0.6424	0.5417	3.701%	0.5662	4.9976	4.0938	2.872%
Pearson*	0.5199	6.9558	5.8977	6.910%	0.5315	0.7157	0.6215	5.203%	0.5538	4.3048	3.6689	2.479%
GA-Envelope entropy	0.5105	8.2136	7.2894	8.745%	0.5248	0.6036	0.5055	3.966%	0.5619	4.1685	3.4866	2.327%
GA-Residual	0.5200	8.6846	7.6082	7.765%	0.5208	0.6231	0.5299	4.110%	0.5401	4.7690	4.1234	2.680%
GA-Pearson	**0.5285**	**5.9776**	**4.9362**	**5.865%**	**0.5673**	0.4380	0.3601	**2.415%**	**0.5712**	**4.0661**	**3.3558**	**2.285%**

(1) The portfolio optimization method that combines GA and residual correlation theory demonstrates superior prediction results on all three datasets, indicating its effectiveness for stock sequence decomposition.

(2) GA-based optimization produces smaller objective function values compared to the exhaustive method, especially when using average residual and average Pearson correlation coefficient. This highlights the effectiveness of the combined optimization method.

(3) Except for the GA approach incorporating residual correlation theory, other GA-based methods may be less effective than the exhaustive method on certain datasets. This observation underscores the complex



(a) (b) (c)

Fig. 9. The objective function values for different K at $\alpha = 2000$ on the SSE dataset.

Table 5
The parameters of VMD for different optimization methods.

Optimization method	ETF		SSE		DJIA	
	K	alpha	K	alpha	K	alpha
Envelope entropy*	2	2000	5	2000	5	2000
Residual*	2	2000	2	2000	2	2000
Pearson*	3	2000	8	2000	7	2000
GA-Envelope entropy	2	1553	4	1330	3	1462
GA-Residual	2	6787	3	1745	2	1812
GA-Pearson	2	1234	3	308	3	367

nature of stock prediction. However, the lower average residual similarity of the stock sequences correlates with better prediction performance of the model in the datasets used in this study.
(4)The VMD algorithm optimized with different parameters optimization methods more stable performance compared to other decomposition algorithms. Though its RMSE and MAE indicators on SSE dataset may not be as good as EEMD, VMD demonstrates stronger robustness.

5.2.2. Impact of the VMD parameters optimization methods

The impact of different stock relationship graphs on model prediction performance was investigated through experiments conducted on multiple datasets. The experimental results are presented in Table 6, where $\mathcal{G}_P$, $\mathcal{G}_{SP}$, and $\mathcal{G}_{DTW}$ refer to models utilizing only Pearson correlation graphs, Spearman correlation graphs, and FastDTW correlation graphs, respectively. The model denoted as $\mathcal{G}_P\&\mathcal{G}_{SP}$ combines Pearson and Spearman correlation graphs using Multi-GCN. The Global model incorporates more $\mathcal{G}_{DTW}$ graphs compared to $\mathcal{G}_P\&\mathcal{G}_{SP}$, while the Global&Local model combines additional local correlation coefficient graphs, including $\mathcal{G}_{wP}$, $\mathcal{G}_{wSP}$, and $\mathcal{G}_{wDTW}$, in addition to the Global model. Based on the findings from Table 6, the following observations and conclusions are drawn:
(1) The model combining global and local correlation graphs had the best prediction performance, followed by the Global model, which was better than the model using a single correlation graph. This confirms the effectiveness and superiority of Multi-GCN. The combination of local and global correlation graphs can capture inter-stock dependencies in a more comprehensive manner.
(2) Among the models using a single correlation graph, the models using $\mathcal{G}_P$ and $\mathcal{G}_{SP}$ had comparable prediction performance, both outperforming the models using $\mathcal{G}_{DTW}$. The model $\mathcal{G}_P\&\mathcal{G}_{SP}$ had better prediction performance than a single relationship graph but worse than the Global model that combines three relationship graphs. This suggests that $\mathcal{G}_{DTW}$ may not be suitable for standalone use, but it can be combined with other correlation graphs to supplement the correlations between stocks.

5.2.3. Ablation studies for each module

To investigate the impact of different modules on the predictive performance of the model, validation is conducted on multiple datasets, and several variants of the model are considered. The performance of the proposed model and its variants on different datasets is presented in Table 7, with consistent experimental configurations for all methods.

The model variants included VGC-GAN1, which did not use the sequence decomposition module, VGC-GAN2, which did not use Multi-GCN, and VGC-GAN3, which did not use the generative adversarial framework. The results show that each module significantly contributes to enhancing the predictive performance of the model. The sequence decomposition module mitigates non-stationarity, Multi-GCN captures inter-stock dependencies, and the generative adversarial framework improves predictive performance and robustness.

The findings highlight the importance of considering all these components collectively, as excluding any of them may result in suboptimal results.

5.3. Analysis of hyperparameters

In this subsection, the impact of two hyperparameters, the MSE loss adjustment parameter k and the sliding window size T_w of the local correlation graphs, on the prediction performance of the model is investigated using multiple datasets.

5.3.1. The setting of MSE loss adjustment parameter k

The prediction performance of the model was evaluated with different values of k on multiple datasets while keeping the experimental configurations consistent across all methods. The results are presented in Table 8, where setting k to 0 indicates that the generative adversarial model only undergoes adversarial training without supervised training. Based on the results in Table 8, the following observations and conclusions can be drawn:
(1) The average prediction performance of the model is best when k is set to 3, followed by k set to 5, indicating that the relationship between the value of k and the model's prediction performance is complex. The optimal value of k (at least the local optimal value) can be determined through experimentation, which is found to be 3.
(2) When k is set to 0, the model's prediction performance is the worst among all values of k on all datasets. This suggests that combining WGAN with MSE loss can more effectively enhance the training process of the generator. It also indicates that the combination of supervised training and adversarial training can significantly improve the prediction performance of the model.

5.3.2. The window length T_w setting for local correlation graphs

To investigate the impact of varying window length T_w on the prediction performance of the model, the number of edges in the local and global correlation graphs for different T_w is calculated and presented in Table 9. Subsequently, the prediction performance of the models combining global and local correlations at different T_w is reported in Table 10.

Based on Tables 9 and 10, it can be observed that the model achieves the most robust average performance on the three datasets when T_w is set to 30. However, the model does not achieve optimal performance on every dataset, indicating that the optimal value of T_w is not fixed but depends on the specific dataset. To further investigate the relationship between T_w and predictive performance, the concept of average volatility of the datasets is introduced, and the volatility of the

Table 6
The predictive performance of models using different stock relationship graphs.

Correlation graph	ETF				SSE				DJIA			
	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE
$\mathcal{G}_P$	0.5120	11.1586	9.9933	10.202%	0.5320	0.6183	0.5255	3.816%	0.5327	6.6933	5.7457	3.650%
$\mathcal{G}_{SP}$	0.5019	11.6316	10.6108	12.497%	0.5397	0.5703	0.4852	3.809%	0.5520	6.7969	5.9296	4.092%
$\mathcal{G}_{DTW}$	0.5071	12.8271	10.8936	14.118%	0.5233	1.1930	0.9921	6.370%	0.4972	13.1879	11.6656	7.808%
$\mathcal{G}_P$&$\mathcal{G}_{SP}$	0.5167	10.7573	9.6465	9.834%	0.5409	0.5857	0.4728	3.632%	0.5538	6.3247	5.4561	3.637%
Global	0.5172	8.7371	7.4255	8.895%	0.5495	0.5440	0.4360	3.354%	0.5612	5.4409	5.1924	3.156%
Global&Local	**0.5285**	**5.9776**	**4.9362**	**5.865%**	**0.5673**	**0.4380**	**0.3601**	**2.415%**	**0.5712**	**4.0661**	**3.3558**	**2.285%**

Table 7
Prediction performance of the proposed model and its variants.

Method	ETF				SSE				DJIA			
	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE
VGC-GAN	**0.5285**	**5.9776**	**4.9362**	**5.865%**	**0.5673**	**0.4380**	**0.3601**	**2.415%**	**0.5712**	**4.0661**	**3.3558**	**2.285%**
VGC-GAN1	0.4941	7.6367	6.6604	7.598%	0.5232	0.6186	0.5135	3.866%	0.5063	5.0972	4.3633	3.002%
VGC-GAN2	0.5118	6.7219	5.7969	12.384%	0.5248	0.6370	0.5059	3.350%	0.5162	4.7962	3.7304	2.579%
VGC-GAN3	0.5278	10.1614	8.8475	10.183%	0.5669	0.4900	0.4073	3.384%	0.5664	5.9008	5.0736	3.245%

Table 8
The prediction performance of various models with different k.

k	ETF				SSE				DJIA			
	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE
0	0.5127	7.7303	6.7363	9.061%	0.5298	0.5177	0.4408	2.859%	0.5414	4.7027	3.9853	2.583%
0.5	0.5255	7.3656	6.3097	7.168%	0.5243	0.4614	0.3920	2.525%	0.5492	4.3140	3.6179	2.461%
1	0.5132	7.3842	6.3137	7.790%	0.5391	0.4467	0.3763	2.732%	0.5560	4.0869	3.3696	**2.281%**
3	**0.5285**	**5.9776**	4.9362	**5.865%**	**0.5673**	**0.4380**	**0.3601**	**2.415%**	**0.5712**	**4.0661**	**3.3558**	2.285%
5	0.5154	6.0322	**4.8980**	6.113%	0.5578	0.4745	0.4000	2.697%	0.5659	4.3962	3.7326	2.434%

Table 9
The number of edges in the correlation graphs for different T_w.

	ETF			SSE			DJIA		
	Pearson	Spearman	FastDTW	Pearson	Spearman	FastDTW	Pearson	Spearman	FastDTW
Global	700	756	1040	70	64	470	18	14	226
T_w=7	646	312	992	234	218	1030	82	48	84
T_w=15	522	342	480	136	96	728	20	14	48
T_w=30	198	132	252	70	64	402	12	12	4
T_w=45	356	264	166	52	52	252	58	20	2

Table 10
The predictive performance of models combining global and local correlations for different T_w.

T_w	ETF				SSE				DJIA			
	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE	Accuracy	RMSE	MAE	MAPE
7	0.5194	5.9730	5.2812	**5.550%**	0.5261	0.3984	0.3336	2.213%	0.5510	4.1780	3.5398	2.352%
15	0.5084	6.1861	5.6443	5.683%	0.5137	0.4138	0.3489	2.469%	0.5708	4.2467	3.5919	2.294%
30	**0.5285**	5.9776	**4.9362**	5.865%	**0.5673**	0.4380	0.3601	2.415%	**0.5712**	**4.0661**	**3.3558**	**2.285%**
45	0.5085	**5.5524**	5.0734	5.879%	0.5467	**0.3820**	**0.3058**	**2.073%**	0.5494	4.1418	3.4835	2.337%

datasets along with the predictive performance of different indicators are compared in Fig. 10. The average volatility is calculated as follows:

$$L_i^{sn} = \log\left(\frac{X_i^{sn}}{X_{i-1}^{sn}}\right) \tag{26}$$

$$volatility = \frac{1}{N}\sum_{sn=1}^{N}\sqrt{\sum_{i=1}^{n-1}\left(L_i^{sn} - \overline{L^{sn}}\right)^2} \tag{27}$$

where n represents the length of the historical stock price series and N represents the number of stocks in the dataset, the average volatility is normalized to a corresponding range in order to better reflect the trend of changes in average volatility.

From Fig. 10, the following observations and conclusions were drawn:

(1) For datasets with small average volatility, a larger T_w, particularly T_w of 30, yields better model performance. However, for datasets with larger average volatility, the performance difference between the model with smaller T_w and the optimal model is smaller, and sometimes even the model with smaller T_w achieves the best performance. This suggests that in datasets with low average volatility, where stock price trends are more stable, a longer sliding window (larger T_w) captures more accurate inter-stock correlation information. Conversely, in datasets with higher average volatility, a smaller sliding window (smaller T_w) may be needed to capture inter-stock correlation information.
(2) The models with T_w of 15 and 30 generally outperform the other two models across the three datasets, indicating a potential local optimum between 15 and 30 that is suitable for the current datasets. To simplify the experiment, T_w of 30 is tentatively determined as the local optimal parameter for the current datasets.

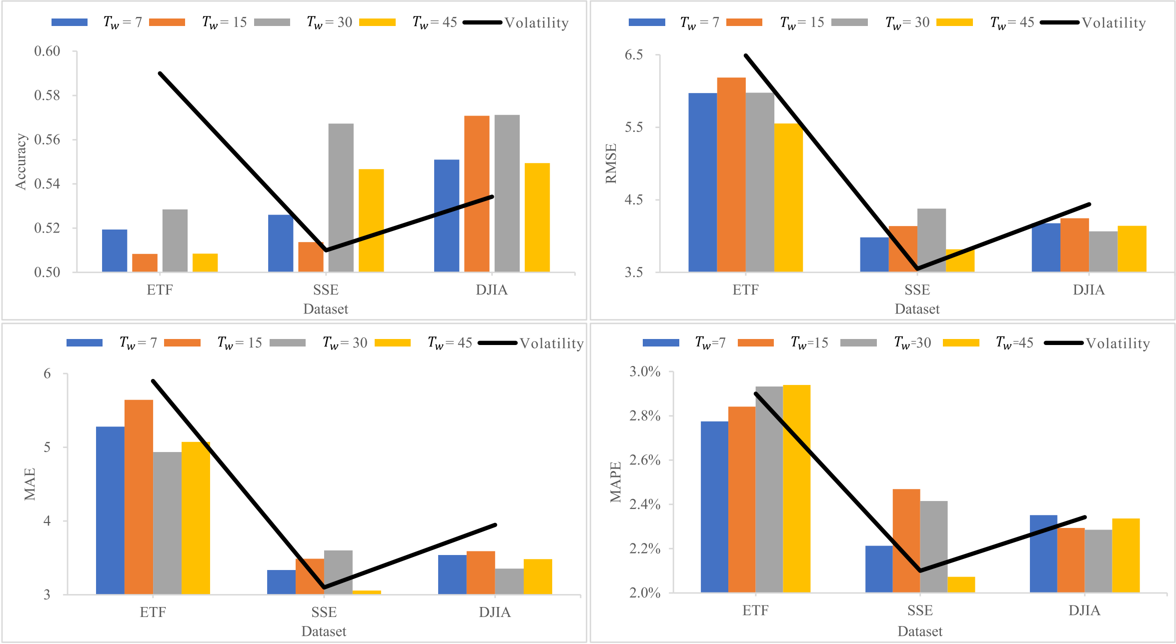


Fig. 10. Volatility of the Dataset and Predictive Performance of Different Indicators.

6. Conclusion

This paper proposes a multi-graph convolutional adversarial network (VGC-GAN) for stock price prediction. The model leverages multiple correlation graphs created based on various perspectives, such as similarity, correlation, and spatial distance among historical stock data. The effect of noise is reduced using the VMD algorithm, and the optimal parameters of VMD are determined using GA and residual correlation theory. Furthermore, we construct a WGAN framework that incorporates MSE loss. The generators in the proposed model include Multi-GCN and GRU, while the discriminator is CNN. Adversarial training and supervised training between the generators and discriminator enhance the generator training process and improve the predictive performance of the model.

Our proposed model combines synchronous and asynchronous correlation between stocks, but it may not be able to handle the dynamic changes in stock relationships. We recognize that changes in the relationship between stocks can introduce noise and impact the performance of our proposed model. Therefore, we agree that future research should explore ways to incorporate dynamic graph convolution to address the changing stock relationships. This will enable our proposed model to adapt to evolving market conditions and provide more accurate predictions over time. We also aim to explore the application of the proposed model to other domains, such as wind power prediction and traffic flow prediction. Overall, our future research will aim to address challenges, improve the model's performance, and extend its application to different domains.

CRediT authorship contribution statement

Dongbo Ma: Conceptualization, Methodology, Software, Validation, Writing – original draft. **Da Yuan:** Conceptualization, Methodology, Validation, Writing – review & editing. **Maojun Huang:** Conceptualization, Methodology, Writing – review & editing. **Ling Dong:** Investigation, Supervision, 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.

Data availability

Data will be made available on request.

Acknowledgments

The authors would like to thank all the anonymous reviewers for their helpful comments and constructive guidance. This work was supported by the National Natural Science Foundation of China under Grant 62072285.

References

Baek, Y., & Kim, H. Y. (2018). ModAugNet: A new forecasting framework for stock market index value with an overfitting prevention LSTM module and a prediction LSTM module. *Expert Systems with Applications*, *113*, 457–480.

Basak, S., Kar, S., Saha, S., Khaidem, L., & Dey, S. R. (2019). Predicting the direction of stock market prices using tree-based classifiers. *The North American Journal of Economics and Finance*, *47*, 552–567.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327.

Cao, D., Wang, Y., Duan, J., Zhang, C., Zhu, X., Huang, C., et al. (2020). Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in Neural Information Processing Systems*, *33*, 17766–17778.

Challu, C., Olivares, K. G., Oreshkin, B. N., Garza, F., Mergenthaler-Canseco, M., & Dubrawski, A. (2022). N-hits: Neural hierarchical interpolation for time series forecasting. arXiv preprint arXiv:2201.12886.

Chen, W., Jiang, M., Zhang, W.-G., & Chen, Z. (2021). A novel graph convolutional feature based convolutional neural network for stock trend prediction. *Information Sciences*, *556*, 67–94.

Chen, Y., Wei, Z., & Huang, X. (2018). Incorporating corporation relationship via graph convolutional neural networks for stock price prediction. In *Proceedings of the 27th ACM international conference on information and knowledge management* (pp. 1655–1658).

Chen, Y., Wu, J., & Wu, Z. (2022). China's commercial bank stock price prediction using a novel K-means-LSTM hybrid approach. *Expert Systems with Applications*, *202*, Article 117370.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

De Haan, L., Mercadier, C., & Zhou, C. (2016). Adapting extreme value statistics to financial time series: Dealing with bias and serial dependence. *Finance and Stochastics*, *20*(2), 321–354.

Ding, G., & Qin, L. (2020). Study on the prediction of stock price based on the associated network model of LSTM. *International Journal of Machine Learning and Cybernetics*, *11*, 1307–1317.

Dragomiretskiy, K., & Zosso, D. (2013). Variational mode decomposition. *IEEE Transactions on Signal Processing*, *62*(3), 531–544.

Enke, D., & Thawornwong, S. (2005). The use of data mining and neural networks for forecasting stock market returns. *Expert Systems with Applications*, *29*(4), 927–940.

Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, *25*(2), 383–417.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. *Advances in Neural Information Processing Systems*, *30*.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT Press.

Hoseinzade, E., & Haratizadeh, S. (2019). CNNpred: CNN-based stock market prediction using a diverse set of variables. *Expert Systems with Applications*, *129*, 273–285.

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Jia, P., Zhang, H., Liu, X., & Gong, X. (2021). Short-term photovoltaic power forecasting based on VMD and ISSA-GRU. *IEEE Access*, *9*, 105939–105950.

Jothimani, D., & Başar, A. (2019). Stock index forecasting using time series decomposition-based and machine learning models. In *Artificial intelligence XXXVI: 39th SGAI international conference on artificial intelligence, AI 2019, Cambridge, UK, December 17–19, 2019, Proceedings 39* (pp. 283–292). Springer.

Jothimani, D., Shankar, R., & Yadav, S. S. (2016). A hybrid EMD-ANN model for stock price prediction. In *Swarm, evolutionary, and memetic computing: 6th international conference, SEMCCO 2015, Hyderabad, India, December 18-19, 2015, Revised Selected Papers 6* (pp. 60–70). Springer.

Khaled, A., Elsir, A. M. T., & Shen, Y. (2022). TFGAN: Traffic forecasting using generative adversarial network with multi-graph convolutional network. *Knowledge-Based Systems*, *249*, Article 108990.

Kumar, A., Alsadoon, A., Prasad, P., Abdullah, S., Rashid, T. A., Pham, D. T. H., et al. (2022). Generative adversarial network (GAN) and enhanced root mean square error (ERMSE): Deep learning for stock price movement prediction. *Multimedia Tools and Applications*, 1–19.

Kumar, D. A., & Murugan, S. (2013). Performance analysis of Indian stock market index using neural network time series model. In *2013 international conference on pattern recognition, informatics and mobile engineering* (pp. 72–78). IEEE.

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

Li, W., Bao, R., Harimoto, K., Chen, D., Xu, J., & Su, Q. (2021). Modeling the stock relation with graph network for overnight stock movement prediction. In *Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence* (pp. 4541–4547).

Li, M., Zhu, Y., Shen, Y., & Angelova, M. (2023). Clustering-enhanced stock price prediction using deep learning. *World Wide Web*, *26*(1), 207–232.

Liang, X., Ge, Z., Sun, L., He, M., & Chen, H. (2019). LSTM with wavelet transform based data preprocessing for stock price prediction. *Mathematical Problems in Engineering*, *2019*.

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, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., et al. (2022). SCINet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, *35*, 5816–5828.

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).

Madan, R., & Mangipudi, P. S. (2018). Predicting computer network traffic: A time series forecasting approach using DWT, ARIMA and RNN. In *2018 eleventh international conference on contemporary computing* (pp. 1–5). IEEE.

Matsunaga, D., Suzumura, T., & Takahashi, T. (2019). Exploring graph neural networks for stock market predictions with rolling window analysis. arXiv preprint arXiv:1909.10660.

Nie, Y., H. Nguyen, N., Sinthong, P., & Kalagnanam, J. (2023). A time series is worth 64 words: Long-term forecasting with transformers. In *International conference on learning representations*.

Niu, H., Xu, K., & Wang, W. (2020). A hybrid stock price index forecasting model based on variational mode decomposition and LSTM network. *Applied Intelligence*, *50*, 4296–4309.

Ozbayoglu, A. M., Gudelek, M. U., & Sezer, O. B. (2020). Deep learning for financial applications: A survey. *Applied Soft Computing*, *93*, Article 106384.

Parray, I. R., Khurana, S. S., Kumar, M., & Altalbe, A. A. (2020). Time series data analysis of stock price movement using machine learning techniques. *Soft Computing*, *24*, 16509–16517.

Picasso, A., Merello, S., Ma, Y., Oneto, L., & Cambria, E. (2019). Technical analysis and sentiment embeddings for market trend prediction. *Expert Systems with Applications*, *135*, 60–70.

Salvador, S., & Chan, P. (2007). Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, *11*(5), 561–580.

Syu, J.-H., Lin, J. C.-W., Wu, C.-J., & Ho, J.-M. (2023). Stock selection system through suitability index and fuzzy-based quantitative characteristics. *IEEE Transactions on Fuzzy Systems*, *31*(1), 322–334. http://dx.doi.org/10.1109/TFUZZ.2022.3187192.

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. *Expert Systems with Applications*, *177*, Article 114800.

Wang, H., Li, S., Wang, T., & Zheng, J. (2021). Hierarchical adaptive temporal-relational modeling for stock trend prediction. In *IJCAI* (pp. 3691–3698).

Wang, W., Lin, W., Wen, Y., Lai, X., Peng, P., Zhang, Y., et al. (2023). An interpretable intuitionistic fuzzy inference model for stock prediction. *Expert Systems with Applications*, *213*, Article 118908.

Wu, M.-E., Syu, J.-H., Lin, J. C.-W., & Ho, J.-M. (2021). Effective fuzzy system for qualifying the characteristics of stocks by random trading. *IEEE Transactions on Fuzzy Systems*, *30*(8), 3152–3165.

Xiang, S., Cheng, D., Shang, C., Zhang, Y., & Liang, Y. (2022). Temporal and heterogeneous graph neural network for financial time series prediction. In *Proceedings of the 31st ACM international conference on information & knowledge management* (pp. 3584–3593).

Xu, H., Chai, L., Luo, Z., & Li, S. (2022). Stock movement prediction via gated recurrent unit network based on reinforcement learning with incorporated attention mechanisms. *Neurocomputing*, *467*, 214–228.

Yan, W.-L. (2023). Stock index futures price prediction using feature selection and deep learning. *The North American Journal of Economics and Finance*, *64*, Article 101867.

Yang, F., Chen, J., & Liu, Y. (2023). Improved and optimized recurrent neural network based on PSO and its application in stock price prediction. *Soft computing*, *27*(6), 3461–3476.

Yang, F., Li, X., Wang, M., Zang, H., Pang, W., & Wang, M. (2022). WaveForM: Graph enhanced wavelet learning for long sequence forecasting of multivariate time series. In *AAAI'23: proceedings of the thirty-seventh aaai conference on artificial intelligence*. AAAI Press.

Ye, J., Zhao, J., Ye, K., & Xu, C. (2021). Multi-graph convolutional network for relationship-driven stock movement prediction. In *2020 25th international conference on pattern recognition* (pp. 6702–6709). IEEE.

Yin, X., Han, Y., Sun, H., Xu, Z., Yu, H., & Duan, X. (2021). Multi-attention generative adversarial network for multivariate time series prediction. *IEEE Access*, *9*, 57351–57363.

Yin, T., Liu, C., Ding, F., Feng, Z., Yuan, B., & Zhang, N. (2022). Graph-based stock correlation and prediction for high-frequency trading systems. *Pattern Recognition*, *122*, Article 108209.

Yin, X., Yan, D., Almudaifer, A., Yan, S., & Zhou, Y. (2021). Forecasting stock prices using stock correlation graph: A graph convolutional network approach. In *2021 international joint conference on neural networks* (pp. 1–8). IEEE.

Zeng, A., Chen, M., Zhang, L., & Xu, Q. (2023). Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*.

Zhang, H., Zhang, S., Wang, P., Qin, Y., & Wang, H. (2017). Forecasting of particulate matter time series using wavelet analysis and wavelet-ARMA/ARIMA model in Taiyuan, China. *Journal of the Air & Waste Management Association*, *67*(7), 776–788.

Zhao, J., Zeng, D., Liang, S., Kang, H., & Liu, Q. (2021). Prediction model for stock price trend based on recurrent neural network. *Journal of Ambient Intelligence and Humanized Computing*, *12*, 745–753.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022). FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, & S. Sabato (Eds.), *Proceedings of machine learning research*: *vol.162*, *Proceedings of the 39th international conference on machine learning* (pp. 27268–27286). PMLR, URL https://proceedings.mlr.press/v162/zhou22g.html.