IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 18 March 2024, accepted 17 April 2024, date of publication 25 April 2024, date of current version 17 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3393774

APPLIED RESEARCH

Multi-Granularity Spatio-Temporal Correlation Networks for Stock Trend Prediction

JIAHAO CHEN[1], LIANG XIE[1], WENJING LIN[1], YUCHEN WU[1], AND HAIJIAO XU[2]
[1]School of Science, Department of Mathematics, Wuhan University of Technology, Wuhan 430070, China
[2]School of Computer Science, Guangdong University of Education, Guangzhou 510303, China

Corresponding author: Liang Xie (whutxl@hotmail.com)

This work was supported in part by the Natural Science Foundation of Guangdong Province under Grant 2020A1515011208, in part by the Science and Technology Program of Guangzhou under Grant 202102080353, and in part by the Characteristic Innovation Project of Guangdong Province under Grant 2019KTSCX117.

ABSTRACT In recent years, time series forecasting has been widely used in various fields, especially in financial markets. Stock trend forecasting has become one of the most common and complex challenges faced by investors and researchers. However, much of the current research relies primarily on single-granularity stock data for forecasting, with relatively few studies on multi-granularity data and fewer studies on spatial correlation of multi-granularity data. This inherent limitation restricts the comprehensive extraction of valuable information. To address this challenge, we propose the Multi-Granularity Deep Spatio-Temporal Correlation Framework (MDSTCF). Our approach combines the strengths of a multi-granularity residual learning, gated recurrent units, and graph attention networks to extract spatio-temporal information specific to each granularity. Subsequently, predictions at each granularity are generated through the prediction layer. Finally, a soft attention mechanism is employed to assign weights to the predictions at each granularity to obtain the final result. Comprehensive experiments conducted on tow stock datasets show that the proposed forecasting model improves the F1 score by about 7.88% and 11.2%, and the cumulative relative returns are close to 80% and 40%, respectively, compared to the previously studied time series forecasting models. The results clearly indicates that fusing multi-granularity information can significantly improve the performance of time series forecasting.

INDEX TERMS Time series forecasting, multi-granularity learning, graph neural network, spatio-temporal correlation, financial markets.

I. INTRODUCTION

Time series forecasting has received extensive attention in academia and industry, including financial markets [1], [2], [3], retail sales [4], transportation [5], [6], and the energy sector [7]. In the financial markets, stock prediction plays a pivotal role in guiding the investment decisions of market participants. Accurate forecasts can help investors make informed choices and thus improve investment returns. However, stock data exhibit characteristics such as nonlinearity and chaos, making stock prediction a challenging task that has received significant attention from researchers over the past decade [8], [9], [10]. Currently, time series forecasts are categorized into two main types: long-term forecasting and short-term forecasting. Long-term forecasting [11] aim to reveal macro trends, formulate strategies and support decision-making, while short-term forecasting focus on coping with near-term volatility, making operational decisions and responding flexibly to unforeseen events. However, today's stock forecasting studies are tilted towards the short-term to cope with the rapidly changing market and business environment.

Traditional time series analysis methods such as SVR [12] and ARIMA [13] can handle nonlinear relationships to a certain extent, but they still have limitations. These methods mainly rely on domain expertise to construct parametric models and most of them can only handle linear relationships between different time steps. In order to simulate nonlinear

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Wang.

relationships, there are variations of autoregressive models such as LRidge [14], LSVR [15], etc. But they still assume that the time series has a certain distribution or functional form, so there are certain limitations when dealing with highly nonlinear real time series data.

In the past few years, deep learning and graph methods [16] have made remarkable achievements in time series forecasting. Deep learning techniques, such as Recurrent Neural Networks (RNN) [17], Long Short-Term Memory Networks (LSTM) [18], [19], and Gated Recurrent Units (GRU) [20], have demonstrated outstanding performance in the prediction of stock trend. Furthermore, stock data contains not only temporal features, but also spatial features due to their complex interrelationships. Graph neural networks, including Convolutional Neural Networks (CNN) [21] and Graph Convolutional Networks (GCN) [22], are gaining increasing popularity in the domain of time series forecasting. These networks leverage the complex interdependencies among stocks to enable more precise price predictions. Consequently, these advancements hold the potential to provide substantial benefits to investors in terms of decision-making and trading strategies. However, it is important to emphasize that many of these studies have largely ignored the exploration of various granularities.

Currently, Multi-Granularity Residual Learning Framework (MRLF) [23] achieves multi-granularity learning and mitigates validity discrepancies by utilizing coarse-grained prior knowledge, computing residuals to remove redundant information, and introducing a self-supervised objective. However, this method is challenged by the complexity of noise in multi-granularity data. To address this problem, Wang et al. proposed the Multi-Granularity Denoising Comparison framework (MDC) [24], which filters the noise, aligns the data, and handles trend differences using self-supervised learning. Despite the progress made, when applied these methods to stock markets which contains multiple time series data, they tend to ignore the spatial correlations between stocks, which affects the accuracy of prediction.

To address the limitations observed in the preceding models, this paper introduces the Multi-Granularity Deep Spatio-temporal Correlation Framework (MDSTCF) to improve the performance of stock trend prediction. Specifically, 1) We construct multiple spatio-temporal network blocks, with each block being responsible for learning spatio-temporal information at a specific granularity level. Our approach not only extracts temporal information, but also effectively captures dynamic spatial information for more effective information acquisition. 2) In order to make full use of the spatio-temporal information of each granularity data, reduce the validity difference and achieve more accurate prediction, we integrate the extracted spatio-temporal hidden states, and use the soft attention mechanism to determine the prediction weight coefficient of each granularity. The final prediction result is the weighted sum of the results of each particle size.

In summary, the main contributions of this work include:

- To learn spatio-temporal features for data with different granularities, we use GRU to capture temporal features and dynamic graphs constructed with specific granularities combined with GAT to capture spatial features, respectively, and finally fused to obtain spatio-temporal relationship embeddings with different granularities.
- After performing residual learning to remove redundant information in the granularity and extract spatio-temporal features, we introduce self-attentive attention and soft-attentive mechanisms to better fuse data of different granularities and make the information of each granularity complementary.
- Extensive experiments on two real-world stock datasets from American and Chinese market clearly show that our model exhibits superior predictive performance compared to other models. In addition, we validate the practical effectiveness of our model through trading strategy.

The rest of the paper is organized according to the following structure. First, in Section II, we present the background of the related work and the problem posed. Then, in Section III the definition of multigranularity is specified as well as the formulation of the proposed problem. Then, in Section IV, we describe the proposed methodology in detail. Next, in Section V, we present the experimental setup and analyze the results obtained. Finally, in Section VI, we will summarize the main points of the paper and suggest directions for subsequent improvements.

II. RELATED WORK

In this section, we explore the application of time series forecasting methods in the stock market and present all the methods organized and summarized in Table 1.

A. TIME SERIES FORECASTING

Some traditional time series methods applied to stock forecasting, such as SVR [12] and ARIMA [13], with roots tracing back to the 1970s, can handle simple time series but fail to balance spatial and temporal correlations. The Multilayer Perceptron (MLP) [28] can be considered as the first post-neural network solution for sequence modeling. With the progress of deep learning, Recurrent Neural Networks (RNNs) [17] gradually became the default choice for time series modeling. However, they encounter the issue of vanishing gradients when dealing with lengthy sequences. Long Short-Term Memory (LSTM) [18] and Gated Recurrent Unit (GRU) [29] help alleviate this problem to some extent but it cannot reach the point where it becomes long-term. In pursuit of more accurate predictions, complex structures like the Time Attention Layer (LSTNet-A) [19] and the novel Temporal Pattern Attention (TPA) [30] have been proposed. Subsequently, the well-known self-attention-based Transformer [25] has been successfully applied to sequence modeling. Despite their success, these methods are

TABLE 1. Comparison of time series forecasting methods.

Type	Tradition Learning/Deep Learning							
Method	ARIMA(1970) [13]	SVR(2001) [12]	GRU(2017) [20]	Transformer(2017) [25]	GCN(2021) [22]	MGTNN(2020) [26]	CMLF(2021) [27]	MRLF(2022) [23]
Multi-Granularity	No	No	No	No	No	No	Yes	Yes
Spatial Correlation	No	No	No	No	Yes	Yes	No	No

predominantly utilized for predicting at a single granularity, lacking extensive research into different granularities.

B. MULTI-GRANULARITY TIME SERIES FORECASTING

Multi-granularity time series prediction has become a hot research topic in the field of time series forecasting in recent years. Scholars from both domestic and international have conducted extensive research and exploration on this issue.

Presently, the predominant focus of research lies in single granularity prediction, as exemplified by the work of Feng et al. [31] and Qin et al. [32]. However, this single-granularity approach often results in a loss of information when viewed from a multi-granularity perspective. In reality, an effective time series prediction model should have the capacity to capture temporal patterns across various time granularities. Although there have been studies using ''multi-scale'' information in time series analysis, this information needs to be clearly distinguished from the ''multi-granularity'' information we talk about. From a more technical perspective, the concept of ''scale'' is mainly related to the division of spectrum. In contrast, ''multi-granularity'' methods, also known as multi-resolution methods, aim to address the challenges of processing data sets with different levels of granularity or aggregation. This includes aggregating statistics or features that occur at different resolutions or time intervals [33].

Over the past two years, Hou et al. introduced the Contrastive Multi-Granularity Learning Framework (CMLF) [27], a framework that leverages multi-granularity temporal data for stock trend prediction tasks. To address the disparity between multi-granularity inputs and single-granularity objectives, the CMLF incorporates an innovative contrastive learning mechanism with dual objectives. In 2022, Hou further extended this work with the introduction of the Multi-Granularity Residual Learning Framework (MRLF) [23] to enhance time series forecasting. However, the above methods do not explicitly consider the relationships between the data at each granularity. In fact, there are dynamic correlations between stocks, and fully utilizing these potential dependencies can improve the predictive performance of the model. Therefore, the spatial relationships formed under each granularity are considered to be incorporated into the model to further improve the accuracy of stock trend prediction.

C. GRAPH NEURAL NETWORKS

Graph neural networks (GNNs) [26] have gained increasing popularity in the field of time series forecasting, where variables are treated as nodes, and their interactions are effectively modeled through edges. It has been demonstrated that significant improvements can be achieved by extracting hidden graph structures, even for tasks where explicit graph structures do not naturally exist. In line with this trend, GNNs, including convolutional neural networks, are becoming increasingly prominent in prediction.

However, most existing works assume static graph structures in the context of multivariate time series forecasting. In reality, there are interdependencies among variables in multivariate time series forecasting methods. These interdependencies between variables can be dynamic, with each variable not only depending on its own historical values but also on other variables. State-of-the-art forecasting models, such as Multivariate Time Series Forecasting with Graph Neural Networks (MTGNN) [26], learn the adjacency matrix of the graph from the entire input time series and retain it for subsequent time modeling. In 2022, the Evolving Multi-Scale Graph Neural Network (ESG) [7] introduced a hierarchical graph structure that incorporates dilated convolutions to capture scale-specific correlations between time series. Most current research focuses on multi-scale graph learning, but this is different from the multi-granularity graph we propose. Multi-granularity graphs focus more on spatial relationships after data refinement, rather than just spatial correlations obtained through frequency or temporal changes.

III. PROBLEM FORMULATION

In this paper we primarily focuses on multi-granularity time series prediction. Given an original sequence $X = [x_1, \ldots, x_T]$ with time length T, time interval 1 and M granularities. Then the coarsest granularity $X^1 = [x_1, x_{[T/s]+1}, x_{[T/s](s-1)}, x_T]$ is to divide T into s equal parts, where s is the largest integer that satisfies the equation $1 = [T/(2^{M-1}s)]$, the 2nd granularity $X^2 = [x_1, x_{[T/(2s)]+1}, x_{[T/(2s)](2s-1)}, x_T]$ is to divide T into 2s equal parts, the mth granularity $X^m = [x_1, x_{[T/(2^{m-1}s)]+1}, x_{[T/(2^{m-1}s)](2^{m-1}s-1)}, x_T]$ is to divide T into $2^{m-1}s$ equal parts, and so on the finest granularity should be $2^{M-1}s$ parts. For example, given that the original sequence is $X^1 = [x_1, x_2 \ldots, x_{T_{28}}]$, the number of granularities is 2. Then $s = 14$ is computed, so here the coarsest granularity is $X^2 = [x_1, x_3 \ldots, x_{T_{27}}, x_{T_{28}}]$ and the finest granularity is $X^1 = [x_1, x_2 \ldots, x_{T_{28}}]$. Finally, we are given N stocks, and the data for M granularities is represented as $\{X^1, X^2, \ldots, X^M\}$ where the order of granularity is from coarse to fine and $X^M = [x_1, \ldots, x_{T^M}] \in R^{N \times T^M \times D}$, where T^M represents the point in time calculated by the method described above. Our model learns the future trend

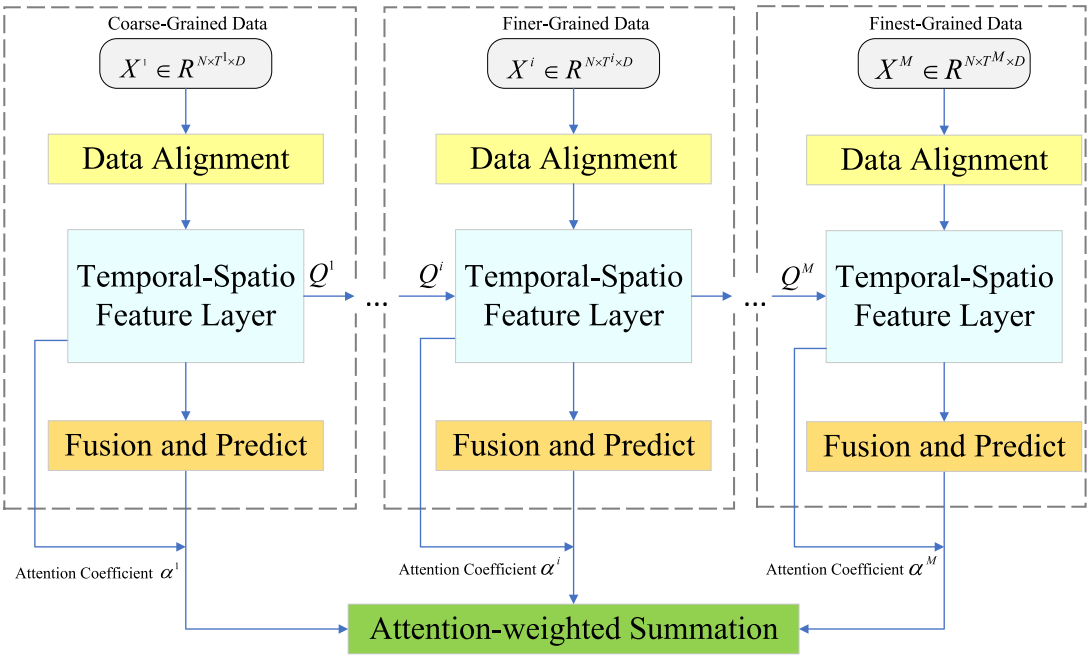


FIGURE 1. Illustration of multi-granularity deep spatio-temporal correlation model (MDSTCF).

function $y = \hat{f}(X^1, X^2, \ldots, X^M)$, based on historical data, where y exceeding the critical threshold (set to 0.5 in our experiments) indicates an upward trend, while falling below the threshold indicates a downward trend.

IV. METHODOLOGY

A. MODEL FRAMEWORK

We will start by outlining the model's overall structure. As illustrated in Fig. 1, our model's input consists of M data granularities from coarse to fine, corresponding to M linear alignment layers, M spatio-temporal feature extraction layers, M fusion prediction layers, and a weighted sum output module.

Firstly, in order to effectively eliminate information redundancy between different granularities, we employ residual connections to acquire non-redundant granularity information. Subsequently, the spatio-temporal feature layer is extracted by the interaction of GRU [20] and GAT [34], capturing temporal and spatial dependencies, respectively. Next, we use a soft attention mechanism to allow the model to assign appropriate correlation coefficients to different granularity, reducing effectiveness differences. Finally, we calculate the weight of each granularity prediction based on these coefficients of concern. The final forecast is obtained by aggregating weights. This process not only ensures that the model can effectively extract the spatio-temporal information of each granularity, but also combines their predictions to mitigate differences to produce more accurate predictions. Specifically, the details of our model are as follows.

B. DATA ALIGNMENT

Due to the inconsistent dimensions of different granularity $\{X^1, X^2, \ldots, X^M\}$, we align them to the same space for subsequent residual operations, as shown in the yellow part of Fig. 1. Specifically, we apply MLP to the input $X^i \in R^{N\times T^i\times D}$, which represents the lagged original data of multiple stocks at granularity i over T^i time steps. $align^i$ involves first swapping the 2nd and 3rd dimensions of X^i, followed by linear transformation to align with the time length, and finally, restoring the original order by swapping back. We describe the feature alignment process as follows:

$$F^i = Align_i(X^i) \tag{1}$$

The aligned features are denoted as $F^i \in R^{N\times K\times D}$, where K represents the time length to which all granularity data are aligned.

C. TEMPORAL FEATURE LAYER

To capture key information in the time series and learn long-term dependencies, we utilize GRU for temporal feature extraction for each stock. The extracted hidden states are used for subsequent spatial feature extraction and prediction at the next level of granularity. The structure of this module is shown in the orange section of Fig. 2. As an illustrative example, we will provide a detailed exposition of the module at granularity level i.

When the upstream output P_i is received, it is then fed into N individual GRU models to capture the hidden features v_i for each stock and to generate predictions for the next-level granularity data v^i_{output}. Here, P_i represents the data obtained after the residual operation, and the specific process will be described in the next subsection. The specific formula is as follows:

$$v^i_{output}, v^i = GRU_i(P^i) \tag{2}$$

$$Q^{i+1} = MLP_i(v^i_{output}) \tag{3}$$

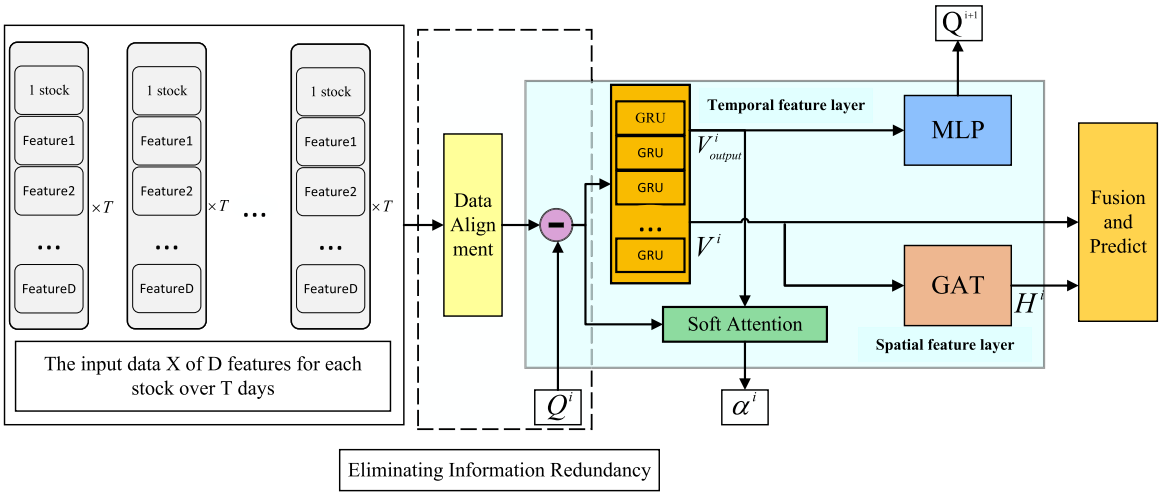


FIGURE 2. The overall structure of the i-th granularity.

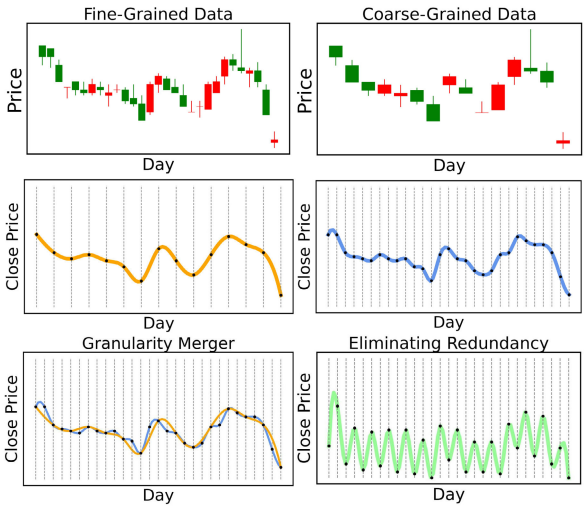


FIGURE 3. Trend of the movement of coarse-grained data and fine-grained data. The last one is the trend of the movement after the residual.

where $v^i \in \mathbb{R}^{N\times D'}$ represents the time features extracted by GRU at i-th granularity, where D' represents the hidden dimension. $v^i_{output} \in \mathbb{R}^{N\times K\times D'}$ predicts the next granularity value $Q^{i+1} \in \mathbb{R}^{N\times K\times D}$. For the prediction, we adopt a Multilayer Perceptron approach. In this case, our temporal feature layer produces two outputs: one for subsequent prediction and downstream spatial information feature extraction v^i, and another Q^{i+1} for the elimination of information redundancy between different granularity in the next layer.

D. ELIMINATING INFORMATION REDUNDANCY

In this study, in order to reduce the redundancy of data at different granularities, we used a residual-based approach. In Fig. 2, an example is used to illustrate the relationship of coarse-grained and fine-grained data. First, we extracted coarse-grained and fine-grained data from the raw data. It is difficult to capture local details by directly inputting fine-grained data because fine-grained data is dominated by coarse-grained trends, which poses a challenge to directly capture fine-grained trends. Specifically, coarse-grained data typically shows general movement trends, while fine-grained data provides more localized change details, as shown in the Fig 3. Since the coarse-grained data is part of the fine-grained data, the overall movement trend information is easier to capture, resulting in localized detail information that is difficult to accurately capture. Therefore, removing redundant information by calculating the residuals between coarse-grained and fine-grained data can capture unique patterns in fine-grained data more effectively. Second, considering that data of different granularities come from statistical measurements at different time levels, coarse-grained data contain a priori information about the distribution of fine-grained data. On this basis, for a given coarse-grained data, we can predict the possible fine-grained data. Finally, by calculating the residuals between different granularity data, redundant information is eliminated in order to obtain specific fine-grained data. For the first layer, we input coarse-grained data $P^1 = F^1$, and the remaining layers use residuals. The specific formula is as follows.

$$P^i = \begin{cases} F^i, i = 1 \\ F^i - Q^i, i \neq 1 \end{cases} \tag{4}$$

where $P^i \in R^{N\times K\times D}$ represents the de-redundant input of layer i, Q^i denotes the current level i granularity data predicted by the level i-th granularity data. By introducing a residual approach, the model is able to more acutely perceive small trend changes in the data, and thus learn and capture the complex correlations therein more accurately.

In order to capture trend information from coarse-grained data while retaining the unique characteristics of fine-grained information in the residual structure, the original fine-grained information must be represented in the coarse-grained information to the maximum extent possible.

We formulate this constraint as:

$$L_1 = \sum ||F^i - Q^i||_F^2 \tag{5}$$

where $||.||_F$ denotes the Frobenius norm. In order not to affect the fine-grained information extraction process, when optimizing Eqs.3, we fix the process of extracting fine-grained information F^i and only optimize the Q^i.

E. SPATIAL FEATURE LAYER

For the formation of spatial features, after the data features of each stock are captured by GRU, the hidden state features obtained can be regarded as nodes containing relevant information of stocks. We use the attention mechanism through GAT to dynamically assign different weights to adjacent nodes, allowing the model to prioritize the more relevant nodes in each prediction task, and then perform convolution to extract the spatial hidden state. In addition, in order to learn the trend relationships under each granularity more efficiently, we introduce weight selection in GAT. This mechanism ensures that each node can focus on learning the spatial state under the current granularity, thus better capturing the stock trend relationships at different time granularities.

Firstly we precompute an adjacency matrix $A \in \mathbb{R}^{N\times N}$, constructed using Dynamic Time Warping (DTW). DTW is primarily employed to capture dynamic relationships between different stocks.

$$A^i = DTW(P^i_{close_i}, P^i_{close_j}) \tag{6}$$

where matrix A^i is a matrix with values of 0 and 1. We use top-K to select entries with higher scores in the matrix. $P^i_{close_i}, P^i_{close_j} \in \times R^K$ represent the data corresponding to the closing price dimension of the i-th and j-th stocks, respectively.

Subsequently, the acquired matrix A^i and the hidden state features v_i are input to Graph Attention Network (GAT), generating latent spatial patterns. Here, we elaborate on the role of the matrix A^i: after GAT autonomously generates the weight matrix $\bar{A}^i$ among nodes, we utilize the adjacency matrix A^i for weight selection, and then use the Softmax operation to reassign the weights. The purpose of this step is to preserve only the weight of stocks that are highly correlated at a given granularity.

$$\begin{aligned}\bar{A}^i &= (\bar{a}^k_{i,j}) \\ &= \left(\frac{\exp(\text{LeakReLU}(a[Wv_i||Wv_j]))}{\sum_{k\in N_i}\exp(\text{LeakReLU}(a[Wv_i||Wv_K]))}\right)\end{aligned} \tag{7}$$

$$(a^k_{i,j}) = \text{Softmax}(A^i \bullet \bar{A}^i) \tag{8}$$

where $a^k_{i,j}$ is obtained by taking the dot product of matrix A^i and matrix $\bar{A}^i$, followed by applying softmax to each row to reassign weights.

Finally a convolutional operation makes the model focus on learning at a specific granularity.

$$H^i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in N_i} a^k_{i,j}\mathbf{W}^k v^i\right) \tag{9}$$

where $H^i \in R^{N\times D''}$ represents the extracted spatial features from GAT. By applying spatial convolution to the temporal data vector v^i, the resulting output H^i can effectively preserve spatio-temporal relationships simultaneously. The purpose of this step is to better retain nodes with higher relevance, eliminate nodes with lower relevance, and thereby focus the model more on learning spatial relationships at the i-th granularity layer.

F. ATTENTION-WEIGHTED SUMMATION

To fully utilize the spatio-temporal information at each granularity, we generate predictions of the probability of stock price increases or decreases by concatenating the captured spatio-temporal features and then passing them to MLP and softmax functions.

$$y^i = Softmax(MLP(H^i \oplus v^i)) \tag{10}$$

where $y^i \in R^N$ represents the probability prediction of the rise and fall of the i-th granularity layer.

To generate the final prediction result from each granularity prediction, we employ soft attention mechanism that assigns a weight to each granularity prediction and then aggregates all the results for the ultimate prediction. First, on the time step T^i of granularity i, in order to extract the historical trend information embedded in the time step, we directly use v^i_{output} as the potential information feature, and then aggregate the history information of the time step through the self-attention mechanism. We then use the soft attention mechanism to calculate the weight coefficient of each particle size prediction. The goal is to make the model more focused on important or relevant parts of learning

$$\begin{aligned}V^i &= \text{Self\_Attention}(v^i_{\text{output}}) \\ &= \sum_{m=1}^{T^i}\text{softmax}\left(\frac{(Q_s, K_m)}{\sqrt{(d_K)}}\right)V^i_m\end{aligned} \tag{11}$$

$$\alpha^i_j = \text{Soft\_Attention}(V^i, P^i_{\text{close}_j}) = \sum_{k=1}^{D}\frac{V^i_{j,2}}{V^i_{j,k}\bullet P^i_{\text{close}_j}} \tag{12}$$

$$\alpha^i = (\alpha^i_1, \ldots, \alpha^i_j, .., \alpha^i_N) \tag{13}$$

where $Q_s = W_Q \bullet v^i_{output,s}(s = T^i)$ is the query vector, $K_m = W_K \bullet v^i_{output,m}(m = 1, \ldots T^i)$ is a key-value vector, d_K is the dimension of the key-value vector, $V^i_m = W_V \bullet v_{output,m}$ is the dimension of the value vector. First, we extract the historical trend information $V^i \in R^{N\times D}$ of the vector v^i_{output} within the time step T^i using an self attention mechanism. Subsequently, we utilize a soft attention mechanism to calculate weight coefficients. In this paper, soft attention refers to the dot product of the D-dimensional features in V^i with our residual granularity closing price data.

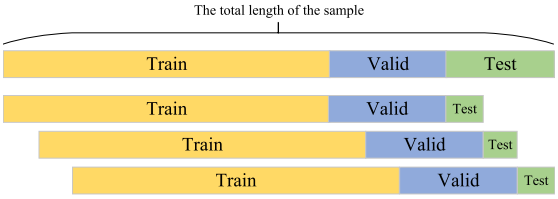


FIGURE 4. Rolling training process.

In Eqs.12 and 13, j represents the j-th stock. The resulting weights $\alpha^i \in R^N$ correspond to the proportion associated with the closing price dimension.

After computing these weight coefficients and assigning the weight α^i through the softmax function, we perform a weighted summation of y^i with the predicted value to derive the final prediction.

$$\bar{\alpha}_j^i = Softmax(\alpha_j{}^i) = \frac{exp(\alpha_j^i)}{\sum_{k\in M} exp(\alpha_k^i)} \tag{14}$$

$$y_j = \sum_{i=1}^{M} \bar{\alpha}_j^i y_j^i \tag{15}$$

$$y = (y_1, \ldots, y_j, .., y_N) \tag{16}$$

where $\bar{\alpha}_j^i$ represents the weight coefficients computed by the previously obtained correlation coefficients, which are further allocated to each stock under each granularity using the Softmax method. By calculating the weighted sum, the probability of an increase or decrease for the j-th stock, denoted as y_j, is determined. $y \in R^N$ represents the probability of upward or downward movement, determined by a threshold.

We adopt this approach to more effectively integrate the results of predictions at different granularities. By calculating the weighted sum, we comprehensively consider the contributions of various granularities to the probability of a specific stock's increase or decrease. This enables a more accurate assessment of the future trend of the stock.

G. MODEL OPTIMIZATION

Considering the optimization of the base model, we combine the cross-entropy loss for the prediction task with the reconstruction loss L in the cross-granularity residual learning process to obtain the following loss function:

$$L = \lambda L_1 + \sum y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \tag{17}$$

λ is the hyperparameter for balancing the loss, and $\hat{y}_i$ represents the true labels, either 0 or 1. We utilize the Adam algorithm to perform batch backpropagation for updating our model parameters. Since each granularity block is responsible for each granularity of learning, they are equally important and are not shared. When updating the parameters, the parameters of each granularity block are updated simultaneously. The training and optimization composition of the model is shown through Algorithm 1.

Algorithm 1 The Learning Algorithm of MDSTCF

Input: $O \in R^{N\times M\times T^i\times D}$,$Y^{label} \in R^{N\times M}$, initialize model parameters $f^i, i = 1, 2, \ldots, M$, learning rate γ, batch size b

Output: $Y \in R^{M\times N}$(This is Outputs)

for sample a batch$X^1, X^2, \ldots, X^M$ from O, y^{label} and from Y^{label} **do**
  for X^i **do**
    Compute P^i specific granular data, according to Eqs.1 and 4
    for each $j \in [0, N]$ **do**
      Compute v_j^i and $v_{output_j}^i$, according to $P^i[:, j, :, :]$ and Eqs.2
      if v_i=None **then**
        $v_i = v_j^i$
        v_{output}^i=$v_{output_j}^i$
      else
        v^i = concat(v_i,v_j^i)
        v_{output}^i = concat(v_{output}^i,$v_{output_j}^i$)
      end if
    end for
    First calculate the correlation coefficient between stock closing prices, and then select the top-K relationship matrix A, which is defined as 1 and the others as 0.
    Compute spatial hidden state vector H^i, according to Eqs.9
    Compute the current granularity prediction result y^i, according to Eqs.10
    Compute the correlation coefficient α^i, according to Eqs.11 and 12
  end for
  Compute final prediction y, according to Eqs.14 and 15
  Compute Loss = $\sum y_i \log y_i^{label} + (1 - y_i) \log(1 - y_i^{label}) + \lambda * L_1[Q^{i+1}, P^{i+1}])$
  Compute the stochastic gradient of θ according to Loss.
  Update model parameters θ according to their gradients and the learning rate γ.
end for

V. EXPERIMENTS

A. EXPERIMENTING SETTINGS

1) DATA SET DESCRIPTION

We utilize the CSI 300 and NASDAQ 100 stock datasets, covering the period from September 28, 2010, to May 9, 2023, encompassing a total of 3064 and 3149 trading days, respectively. Initially, we excluded stocks that went public after September 28, 2010. We then make a time alignment from these selected stocks. If some stocks are missing data on some days, we use the previous day's data to fill in the missing data for the day. In the end, we selected 183 and 83 stocks on the CSI300 and NASDAQ100 for training and

TABLE 2. Model input variables.

Technical indicator	Formula
Momentum	$C_t - C_{t-n}$
Moving average convergence divergence(MACD)	$EMA_{12} - EMA_{26}$
Relative strength index (RSI)	$100 - \frac{100}{1+(\sum_{t=0}^{n-1} Up_{t-I}/n)/(\sum_{t=0}^{n-1} Dw_{t-I}/n)}$
Moving average	EMA_5
Feature	**Formula**
z_{open}	$z_{open} = open_t/close_t - 1$
z_{high}	$z_{high} = high_t/close_t - 1$
z_{low}	$z_{low} = low_t/close_t - 1$
z_{close}	$z_{close} = close_t/close_{t-1} - 1$
z_{adj_close}	$z_{adj\_close} = adj\_close_t/adj\_close_{t-1} - 1$
z_{volume}	$z_{volume} = volume_t/volume_{t-1} - 1$

testing. All datasets were partitioned into training, validation, and test sets, following a distribution of 70%, 15%, and 15%, respectively.

2) INPUT DATA

In this paper, 6 features and 4 technical indicators are calculated, the effectiveness of which has been confirmed in numerous studies [14], [35] focusing on stock market forecasting. The detailed formula is shown in Table 2, where EMA stands for moving average, e.g. EMA12 means using 12 days of data summed and averaged to get. Finally, it is normalized and used as an input variable. In the stock market, since the market is in a constant state of change, we choose to use a rolling training method in order to adjust the model to the new market in time. To handle the case of different lengths of the CSI300 and NASDAQ100 test sets, we used the method of dividing the test set into 3 equal parts to determine the sliding window size. To keep the length of the training, validation, and test sets consistent, each time we slide to a new test set, we remove a segment of the head of the training and validation set that is equal in length to a sliding window, and then replenish the portion of the test set that was slid over. The specific the rolling process are shown in Fig. 4.

3) HYPERPARAMETER SETTINGS

In our method, the hidden layer dimension size is chosen as 64 from $[32, 64, 96, 128]$, the batch size is chosen as 64 from $[16, 32, 64, 128]$ and the learning rate is chosen from $[10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$ is chosen as 10^{-3}, the learning rate weight decay is chosen from $[10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$ as 10^{-3} The model predicts the rise-fall threshold is chosen to be 0.5 from the set $[0.5, 0.6]$ and k is chosen to be 20 from the set $[10, 20, 30, 40]$. Since the model tends to predict the fall, in order to solve the problem, we assign some weights to the cross-entropy loss function, where the rise and fall weights are set to be 1.2 from the set [1,1.5]. The size of λ in Eq. 17 is $[10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1]$ as 1. The number of training times is set to 10,

B. COMPARISON METHODS

To demonstrate the computational performance of our proposed method MDSTCF, we compared its results with those of other methods, including GRU [20], GCN [22], GAT [34], TRANSFORMER [36], MGTNN [26] and MRLF [23]. Each method was executed 50 times to reduce randomness and obtain mre robust results.

- GRU: This method can effectively capture long-term and short-term dependencies in a stock price series. This allows it to identify the impact of price movements over the past period of time on the future. For this method, the dimension size of the hidden layer is chosen as 64 from $[32, 64, 96, 128]$, the batch size is chosen as 64 from $[16, 32, 64, 128]$, the learning rate is chosen as 10^{-3} from $[10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$, the weight decay of the learning rate is chosen as 10^{-2} from $[10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$, and the threshold of the model prediction rise and fall is chosen as 0.5 from the set $[0.5, 0.6]$.
- GCN: This method is a method that combines GRU and GCN. First, GRU is used to extract trend information from historical stock data and convert it into time series features. Next, the GCN part captures the correlation between stocks by performing convolution operations on a predefined graph structure, realizes information transmission and fusion, and further refines valuable stock features and outputs the final stock trend prediction. The parameters of this method are the same as those of the our method.
- GAT: Firstly, GRU is used to capture the time series characteristics in stock price fluctuations. Then, GAT is used to calculate the attention of the extracted time series features to obtain the dependency weights between stock prices. Finally, through convolution, information transfer and fusion are realized, valuable features are extracted, and the final stock trend prediction is output. The parameters of this method are the same as those of the our method.
- TRANSFORMER: This method model uses the encoder-decoder structure and self-attention mechanism to effectively capture the time characteristics in the historical stock series, so as to improve the prediction accuracy of stock prices. Especially in capturing long-term dependencies, Transformer shows good predictive ability. The parameters of this method are the same as those of the GRU method.

- MGTNN: This method automatically extracts the relationship between variables through the graph learning module. In addition, the hybrid hop propagation layer and the extended initial layer can capture spatio-temporal dependencies in time series. Finally, the convolutional layer is used to predict the stock trend. The l2 regularization penalty is chosen as 10^{-2} from $[10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}]$. Apply Layernorm after each graph convolution module. Set the mix-hop propagation layer depth to 2. The retention ratio for the hybrid hop propagation layer is set to 0.05. The saturation rate of the graph learning layer activation function is set to 3. The number of dimensions embedded in the node is 40.
- MRLF: The method removes redundant coarse-grained trend information and captures useful information in fine-grained data for more reliable stock trend forecasting. The parameters of this method are the same as those of the GRU method.

C. EVALUATION METRICS

To demonstrate the effectiveness of the proposed model, we conducted a comparison using the following evaluation metrics. These metrics are categorized into two groups: one evaluates the accuracy of our model's trend predictions, and the other involves a simple investment portfolio analysis to validate our model's performance compared to other models.

Our method predicts the trend of whether the stock price will rise or fall for the next day. Therefore, we employed the following metrics to analyze the performance, with their specific formulas provided below.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{18}$$

$$Precision = \frac{TP}{TP + FP} \tag{19}$$

$$Recall = \frac{TP}{TP + FN} \tag{20}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{21}$$

where TP = true positives, TN = true negatives, FP = false positives and FN = false negatives.

Additionally, we evaluate the method's prediction results through a simple investment portfolio. Here is a description of the portfolio method: Firstly, we obtain the daily probability predictions for each stock. Then, we set a threshold (in our experiments, the threshold is set at 0.5) to identify which stocks are predicted to rise today based on the predicted probabilities. Next, we select the top-performing stocks (in our experiments, we buy the top 10 stocks with the highest probabilities), and we hold them for 3 days before selling.

In order to assess the validity of the results of our portfolio forecasts, we have used the following metrics.

$$CR = \frac{v_t - v_0}{v_0} \tag{22}$$

$$SR = \frac{E[ROR_P] - r_f}{\sigma_p} \tag{23}$$

$$ARR = (1 + CR)^{\frac{252}{e}} - 1 \tag{24}$$

$$MDD = \max_{i \in (0,t)} \left\{ \max_{j \in (0,t)} \frac{v_j - v_i}{v_j} \right\} \tag{25}$$

In Eqs.22 and 25, v_0,v_t,v_i,v_j represents the assets at that moment. Eqs.23 uses $E[ROR_P]$ for the expected return of the investment portfolio, r_f for the risk-free rate, and σ_p for the standard deviation of the investment portfolio return. In Eqs.24, e represents the duration of the trade.

D. MAIN RESULTS

After conducting extensive experiments, our obtained results are presented in Table 3, which respectively showcase various evaluation metrics and the final returns of both our method and the comparative method.

In the experimental evaluation, our method is compared with other methods on various performance metrics, which are compared in detail in Table 3, while Fig. 5 visualize the results of these comparisons. These confirm that our method has better results. The MDSTCF model we have constructed consistently exhibits superior performance when applied to the CSI 300 and NASDAQ 100 datasets, resulting in higher F1 scores and increased returns. These performance metrics underscore the effectiveness of our approach in the domain of stock prediction. Below we conduct a detailed analysis of the results.

On the CSI300, when our method was compared with traditional time series methods such as GRU and Transformer, the accuracy of our model improved by 0.35% and 2.04%, precision by 4.65% and 5.31%, recall by 8.39% and 6.12%, and F1/score by 6.96% and 5.23%, respectively. This suggests that incorporating multi-granularity information can enhance predictive performance.

In comparison to graph network models, specifically GCN and GAT, our model shows improvements in accuracy by 1.61% and 1.67%, precision scores improving by 8.48% and 8.11%, recall scores improving by 8.07% and 3.17%, and F1_score increase of 7.88% and 5.48%. This suggests that the performance of temporal feature fusion prediction using only to graph networks may not be as effective, and that the combination of temporal and spatial information can help improve prediction performance.

When compared to recent time series models, namely MGTNN and MRLF, our model achieves an accuracy score increase of 4.36% and 1%, precision scores increase by 2.53% and 3.86%, recall scores increase by 8.07% and 3.86%, and F1_score increases by 4.07% and 5.48%. Similarly, on the NASDAQ 100 dataset, we observe performance improvements compared to other models. These results underscore the superior performance of our model across multiple datasets and highlight the potential of multi-granularity information integration for enhancing time series forecasting.

To provide a more comprehensive overview of our model's real-world performance, we conduct a detailed analysis in

TABLE 3. Performance of MDSTCF and comparison methods on stock dataset.

Dataset	model	Accuracy	Precision	Recall	F1_score	MDD	SR	CR	ARR
CSI300	Transformer	0.5061	0.4870	0.5145	0.5002	0.2916	0.5699	0.1537	0.2637
	GCN	0.5084	0.4728	0.5052	0.4880	0.3813	-0.7699	-0.2473	-0.3718
	GRU	0.5148	0.4901	0.5037	0.4922	0.2710	0.3348	0.0283	0.0467
	GAT	0.5081	0.4744	0.5292	0.4991	0.2353	-0.7819	-0.1332	-0.2087
	MTGNN	0.4950	0.5002	0.5246	0.5107	0.4026	-1.216	-0.3483	-0.5037
	MRLF	0.5115	0.4938	0.5168	0.5050	0.3789	-0.4267	-0.1497	-0.2331
	MDSTCF	0.5166	0.5129	0.5460	0.5265	0.2253	1.0927	0.5588	1.0676
NASDAQ100	Transformer	0.5104	0.4994	0.5249	0.5056	0.3850	0.0220	0.0563	0.0905
	GCN	0.5180	0.4796	0.5099	0.4956	0.2830	0.5168	0.0875	0.1423
	GRU	0.5104	0.5062	0.5451	0.5238	0.3158	0.4979	0.0820	0.1330
	GAT	0.5032	0.4874	0.5325	0.5089	0.3711	0.1759	0.0213	0.0336
	MTGNN	0.5016	0.5071	0.5628	0.5309	0.3492	0.6563	0.1360	0.2241
	MRLF	0.5122	0.4816	0.5248	0.5001	0.3313	0.0827	-0.0613	-0.0954
	MDSTCF	0.5186	0.5175	0.5892	0.5512	0.2523	1.0432	0.3339	0.5788

TABLE 4. Mann-Whitney U test results on two datasets.

Test of significance method		Wilcoxon signed-rank test $\alpha = 0.05$ p-value			
Dataset	Compared models	Accuracy	Precision	Recall	F1_score
CSI300	MDSTCF VS Transformer	0.0328	0.0264	0.0149	0.0204
	MDSTCF VS GCN	0.0297	0.0292	0.0105	0.00**
	MDSTCF VS GRU	0.0220	0.0286	0.0149	0.00**
	MDSTCF VS GAT	0.0144	0.0251	0.0227	0.00**
	MDSTCF VS MTGNN	0.0290	0.0273	0.0144	0.0205
	MDSTCF VS MRLF	0.0182	0.00**	0.0252	0.00**
NASDAQ100	MDSTCF VS Transformer	0.0105	0.0113	0.0182	0.00**
	MDSTCF VS GCN	0.0249	0.0253	0.00**	0.00**
	MDSTCF VS GRU	0.0317	0.0326	0.0268	0.00**
	MDSTCF VS GAT	0.0206	0.0294	0.0218	0.00**
	MDSTCF VS MTGNN	0.0260	0.0241	0.0275	0.00**
	MDSTCF VS MRLF	0.0252	0.0174	0.00**	0.00**

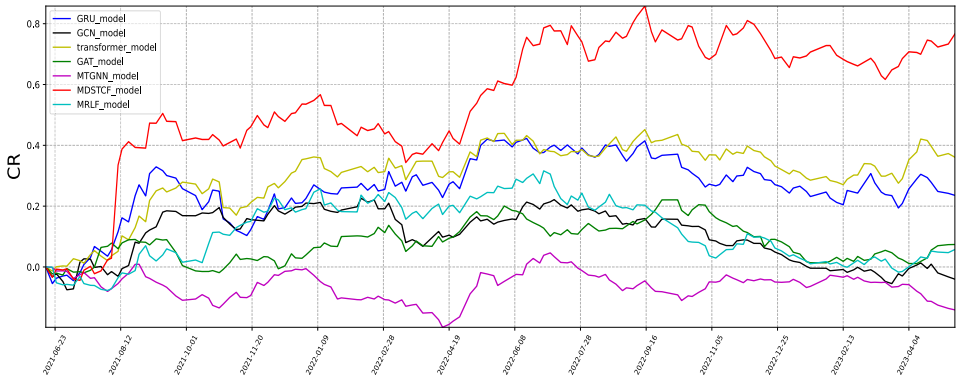
conjunction with the relative cumulative returns shown in Fig. 5, where the relative cumulative return is the actual return of the portfolio over time compared to the return of the benchmark index over the same period. The higher relative returns highlight the model's sensitivity to market changes, suggesting that the model is able to make relatively accurate predictions. From 2021 to 2023, our model consistently outperforms other models in terms of returns. To be more precise, our model exhibits an approximately 80% increase in returns relative to the CSI 300 index and a nearly 40% increase in returns relative to the NASDAQ 100 index. These figures underscore the superior performance of our model in real-world market trading.

In order to verify the superiority of our proposed MDSTCF model compared to other models in terms of accuracy, precision and recall, we conducted a significance test experiment using Wilcoxon signed-rank test. The results in Table 4 show that the accuracy, precision, recall and F1 score of our proposed MDSTCF model are significantly different from the results of other models with the significance level set at 0.05. For example, the first value of 0.0328 (<0.05) in the table indicates that our model is significantly different from Transformer in terms of precision rate prediction. This experiment-al result strongly demonstrates the significant performance improvement of our proposed model.

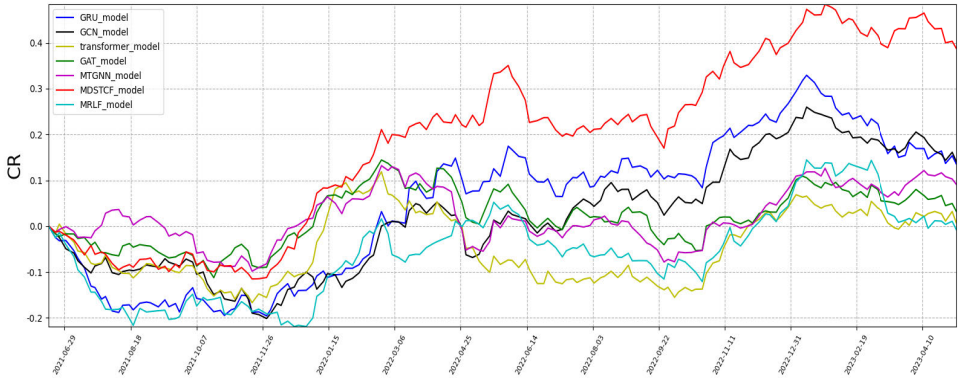
E. ABLATION STUDY

In this experimental research, ablation experiments was carried out to investigate on investigating the impact of certain model components on the overall model performance. These experiments were deliberately designed to unveil and elucidate the distinct contributions and significance of each specific component, thereby facilitating a more profound insight into the model's internal mechanisms. A more comprehensive understanding of the model's performance and fundamental characteristics can be attained through a meticulous analysis of the individual impact of each component. The following are some variants of MDSTCF, where w/o means that the corresponding module that follows is not used.

- MDSTCF(Single granularity): Represents a single layer spatio-temporal network, that is, spatio-temporal feature extraction for a single granularity.
- MDSTCF(w/o Sptial): The model employs a two-layer method, specialized for extracting spatial features at two different granularities.
- MDSTCF(w/o Temporal): The model employs a two-layer method, specialized for extracting temporal features at two different granularities.
- MDSTCF(w/o Soft_Attention): Indicates a method of using only two spatial extraction layers to extract spatial features of two different granularities, but only



(a) Cumulative return rate relative to the CSI 300 Index



(b) Cumulative return rate relative to the NASDAQ100 Index

FIGURE 5. **Some comparative methods and cumulative returns of our model on the two datasets.**

doing a simple average without using the attention mechanism.

The results in Table 5 clearly show that removing specific model components leads to performance degradation, with the removal of the temporal feature layer having the most pronounced effect, leading to a significant degradation in performance, with the accuracy dropping to 46% on the CSI 300 dataset and the F1 score decreasing by 6.1% and 9.7% on both datasets, respectively. For not using the spatial feature layer component and using only the single-granularity layer, the F1 scores were reduced by 4% and 5.9%, respectively. The impact of not using the soft attention mechanism is relatively minor, with a 2.3% and 5.4% reduction in F1 score. These results highlight the critical and irreplaceable role of each model component in the overall effectiveness of our model.

F. PARAMETER ANALYSIS

The following is an analysis of the impact of the model from three aspects: different granularity(G^1, G^2, G^3 represent from one to three grain sizes, respectively), test set sliding length(W^1, W^2, W^3, W^4, W^5 represent the values corresponding to sliding window sizes after dividing the test set into 1 to 5 equal parts), and hidden layer size(H^1, H^2, H^3, H^4 represent the hidden state sizes of 32, 64, 96, and 128, respectively).

1) DIFFERENT GRANULARITY

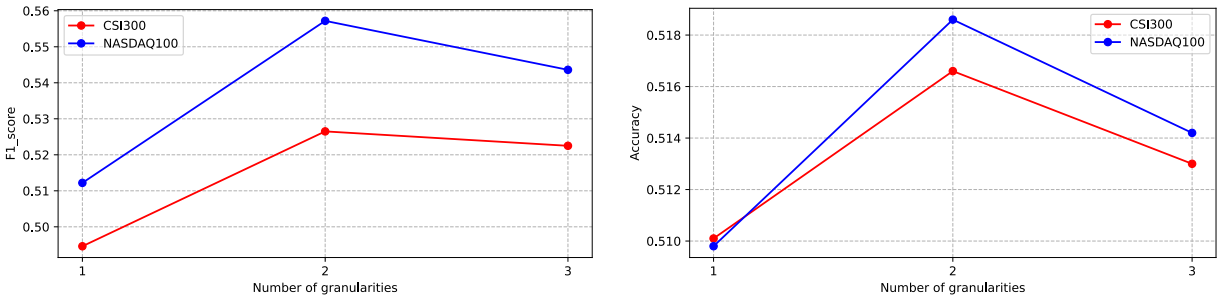
In our experiments, our primary focus is on the analysis of different granularities in terms of accuracy and overall performance, as measured by the F1 score. Examining Fig. 6, it becomes evident that the results of the single-grain-size model are comparatively suboptimal. However, the score of the two-granularity model is slightly superior to that of the three-granularity model. Furthermore, our experiments have revealed that the spatio-temporal complexity of the three-granularity model increases by approximately two times when compared to the two-granularity model. Considering all these aspects, the two-granularity model appears to be the most suitable choice at the current stage.

2) TEST SET SLIDING LENGTH

To ensure consistent prediction lengths, we divided the test set into equal segments to determine the size of the sliding window. Initially, the entire test set was fed into the model for testing, resulting in relatively poor performance. Subsequently, we attempted partitioning the test set into 2 to 5 segments, observing a gradual improvement in performance. Moreover, finer partitioning demonstrated better results, as illustrated in Fig. 7. However, when partitioning the test set into 3 segments or more, the improvement was marginal. Considering the constraint of runtime, we ultimately opted to partition the test set into

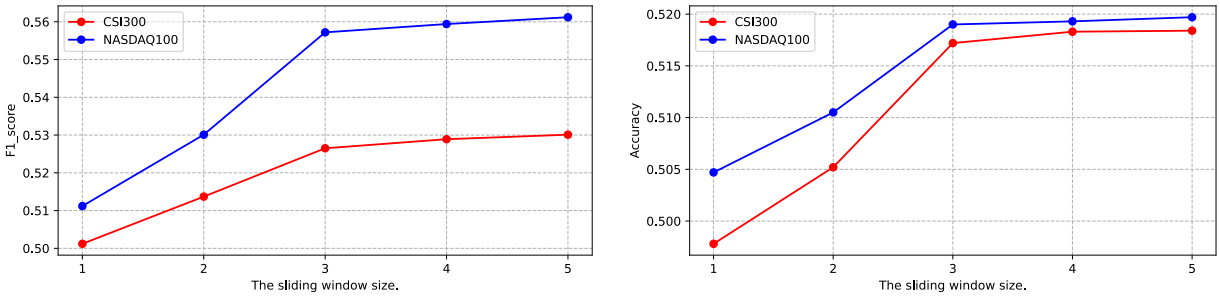
TABLE 5. Performance analysis of ablation experiments for MDSTCF.

Dataset	model	Accuracy	Precision	Recall	F1_score
CSI300	MDSTCF(Single granularity)	0.5021	0.4921	0.4967	0.4946
	MDSTCF(w/o Temporal)	0.5015	0.4638	0.5168	0.4868
	MDSTCF(w/o Sptial)	0.5130	0.5004	0.5107	0.5058
	MDSTCF(w/o Soft_Attention)	0.5074	0.5061	0.5254	0.5144
NASDAQ100	MDSTCF(Single granularity)	0.5138	0.5156	0.5274	0.5208
	MDSTCF(w/o Temporal)	0.5027	0.4732	0.5255	0.4982
	MDSTCF(w/o Sptial)	0.5142	0.5052	0.5354	0.5191
	MDSTCF(w/o Soft_Attention)	0.5076	0.5089	0.5342	0.5215



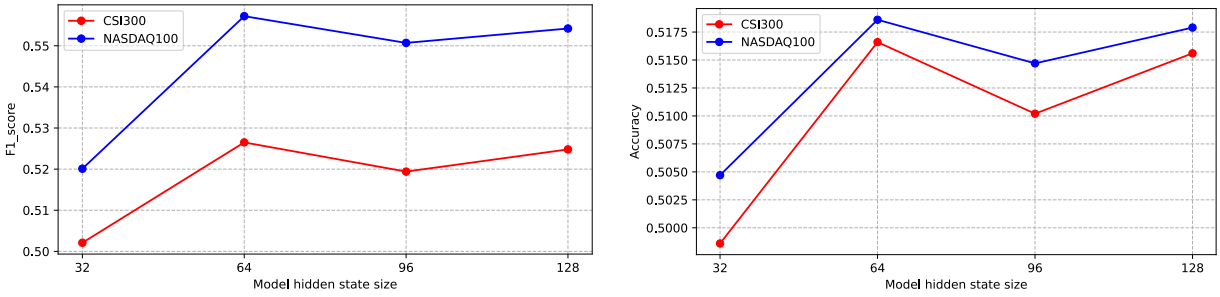
(a) F1 score at different granularities (b) Accuracy at different granularities

FIGURE 6. (a) Is F1 score at different granularities, (b) is the accuracy rate at different granularities.



(a) F1 scores for different sliding window sizes (b) Accuracy for different sliding window sizes

FIGURE 7. The x-axis in (a) and (b) represents the division of the test set into 1 to 5 segments. The sliding window size is determined based on the length of each segment. Therefore, here, 1 represents the entire length of the test set as the sliding window size, and 5 represents the sliding window size after dividing the dataset into 5 segments.



(a) F1 scores for different hidden layer sizes (b) Accuracy for different hidden layer sizes

FIGURE 8. (a) Represents the F1 score of the model at different hidden dimensions, (b) represents the accuracy of the model at different hidden dimensions.

3 segments as the sliding length. This decision was made after considering both model performance and operational efficiency to ensure an appropriate balance of predictive effects.

3) HIDDEN STATE SIZE

In the experiments, we adjusted the size of the hidden layer, initially opting for 64 and subsequently exploring 32, 96, and 128 to investigate the impact of different parameters

on the model. As shown in Fig. 8, on the one hand, the model complexity decreases when the hidden state size is 32, leading to underfitting and poor performance on the training data. On the other hand, hidden layer sizes of 96 and 128 increased model complexity, aiding in better learning of complex features in the training data but causing a slight decrease in performance due to overfitting. Considering a comprehensive assessment of performance, generalization capability, and computational efficiency, we ultimately chose a hidden layer size of 64 for the model.

VI. CONCLUSION

In this study, a new deep learning model is proposed to deal with spatio-temporal information as well as the union of spatio-temporal information for data of different granularities. Our approach incorporates the advantages of multi-granularity residual learning, gated recurrent units, and graph attention networks to extract spatio-temporal information specific to each granularity. In multi-granularity residual learning, we are able to better capture spatio-temporal correlations at different temporal granularities and improve the sensitivity of the model to data features. In gated recursive units, the introduction of gated recursive units helps to process sequential data, enabling the model to better capture and utilize long-term dependencies in time series. Then GAT and DTW are integrated to deal with complex spatial relationships in stock data and improve the model's ability to model inter-stock correlations. In soft attention mechanism, we are able to assign weights to the prediction results of each granularity, making the model pay more attention to the granularity with better performance, thus improving the overall prediction accuracy. By conducting rich experiments on CSI300 and NASDAQ100, our model outperforms other models in comparison experiments. In the ablation experiments, the performance of the model decreases accordingly when each component of the model is gradually removed, further validating the importance of each component.

In our future research, we will further optimize two aspects of the MDSTCF model. Firstly, we will deeply explore the relationship between temporal and spatial correlations to enhance the accuracy of stock prediction. Secondly, we plan to optimize the model by considering textual content such as market sentiment and relevant news headlines to obtain more spatial relationships.

ACKNOWLEDGMENT

The authors would like to sincerely thank all the anonymous reviewers, who invested a great deal of time and effort in the review process. Their professional comments were helpful in improving the quality of the manuscript.

REFERENCES

[1] S. Deng, N. Zhang, W. Zhang, J. Chen, J. Z. Pan, and H. Chen, "Knowledge-driven stock trend prediction and explanation via temporal convolutional network," in *Companion Proc. World Wide Web Conf.*, May 2019, pp. 678–685.

[2] H. Wang, S. Li, T. Wang, and J. Zheng, "Hierarchical adaptive temporal-relational modeling for stock trend prediction," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, Aug. 2021, pp. 3691–3698.

[3] R. Sawhney, S. Agarwal, A. Wadhwa, and R. R. Shah, "Spatiotemporal hypergraph convolution network for stock movement forecasting," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2020, pp. 482–491.

[4] C. Li, B. Cheang, Z. Luo, and A. Lim, "An exponential factorization machine with percentage error minimization to retail sales forecasting," *ACM Trans. Knowl. Discovery Data*, vol. 15, no. 2, pp. 1–32, Apr. 2021.

[5] Y. Cui, K. Zheng, D. Cui, J. Xie, L. Deng, F. Huang, and X. Zhou, "METRO: A generic graph neural network framework for multivariate time series forecasting," *Proc. VLDB Endowment*, vol. 15, no. 2, pp. 224–236, Oct. 2021.

[6] S. Li, L. Ge, Y. Lin, and B. Zeng, "Adaptive spatial–temporal fusion graph convolutional networks for traffic flow forecasting," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2022, pp. 4189–4196.

[7] J. Ye, Z. Liu, B. Du, L. Sun, W. Li, Y. Fu, and H. Xiong, "Learning the evolutionary and multi-scale graph structure for multivariate time series forecasting," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 2296–2306.

[8] C. K.-S. Leung, R. K. MacKinnon, and Y. Wang, "A machine learning approach for stock price prediction," in *Proc. 18th Int. Database Eng. Appl. Symp.*, 2014, pp. 274–277.

[9] X. Ying, C. Xu, J. Gao, J. Wang, and Z. Li, "Time-aware graph relational attention network for stock recommendation," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 2281–2284.

[10] X. Teng, X. Zhang, and Z. Luo, "Multi-scale local cues and hierarchical attention-based LSTM for stock price trend prediction," *Neurocomputing*, vol. 505, pp. 92–100, Sep. 2022.

[11] Z. Chen, M. Ma, T. Li, H. Wang, and C. Li, "Long sequence time-series forecasting with deep learning: A survey," *Inf. Fusion*, vol. 97, Sep. 2023, Art. no. 101819.

[12] T. Van Gestel, J. A. K. Suykens, D.-E. Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewalle, "Financial time series prediction using least squares support vector machines within the evidence framework," *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 809–821, Jul. 2001.

[13] G. E. P. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *J. Amer. Stat. Assoc.*, vol. 65, no. 332, p. 1509, Dec. 1970.

[14] J. Yoo, Y. Soun, Y.-C. Park, and U. Kang, "Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 2037–2045.

[15] Q. Ding, S. Wu, H. Sun, J. Guo, and J. Guo, "Hierarchical multi-scale Gaussian transformer for stock movement prediction," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 4640–4646.

[16] W. Xia, T. Li, and C. Li, "A review of scientific impact prediction: Tasks, features and methods," *Scientometrics*, vol. 128, no. 1, pp. 543–585, Jan. 2023.

[17] M. Han and M. Xu, "Laplacian echo state network for multivariate time series prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 238–244, Jan. 2018.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[19] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jun. 2018, pp. 95–104.

[20] M. R. Vargas, B. S. L. P. de Lima, and A. G. Evsukoff, "Deep learning for stock market prediction from financial news articles," in *Proc. IEEE Int. Conf. Comput. Intell. Virtual Environments Meas. Syst. Appl. (CIVEMSA)*, Jun. 2017, pp. 60–65.

[21] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 2014–2023.

[22] W. Chen, M. Jiang, W.-G. Zhang, and Z. Chen, "A novel graph convolutional feature based convolutional neural network for stock trend prediction," *Inf. Sci.*, vol. 556, pp. 67–94, May 2021.

[23] M. Hou, C. Xu, Z. Li, Y. Liu, W. Liu, E. Chen, and J. Bian, "Multi-granularity residual learning with confidence estimation for time series prediction," in *Proc. ACM Web Conf.*, Apr. 2022, pp. 112–121.

[24] M. Wang, F. Chen, J. Guo, and W. Jia, "Improving stock trend prediction with multi-granularity denoising contrastive learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jun. 2023, pp. 1–10.
[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–17.
[26] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 753–763.
[27] M. Hou, C. Xu, Y. Liu, W. Liu, J. Bian, L. Wu, Z. Li, E. Chen, and T.-Y. Liu, "Stock trend prediction with multi-granularity data: A contrastive learning approach with adaptive fusion," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 700–709.
[28] C.-T. Lin, Y.-K. Wang, P.-L. Huang, Y. Shi, and Y.-C. Chang, "Spatial–temporal attention-based convolutional network with text and numerical information for stock price prediction," *Neural Comput. Appl.*, vol. 34, no. 17, pp. 14387–14395, Sep. 2022.
[29] J. Zhang and Z. Zhang, "Research on fermented grains steam running condition prediction model of make Chinese liquor robot based on GRU," in *Proc. 3rd Int. Conf. Adv. Inf. Sci. Syst.*, Nov. 2021, pp. 1–4.
[30] S.-Y. Shih, F.-K. Sun, and H.-Y. Lee, "Temporal pattern attention for multivariate time series forecasting," *Mach. Learn.*, vol. 108, nos. 8–9, pp. 1421–1441, Sep. 2019.
[31] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, "Enhancing stock movement prediction with adversarial training," 2018, *arXiv:1810.09936*.
[32] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," 2017, *arXiv:1704.02971*.
[33] M. Reis, "Multiscale and multi-granularity process analytics: A review," *Processes*, vol. 7, no. 2, p. 61, Jan. 2019.
[34] Y. Wang, C. Jing, S. Xu, and T. Guo, "Attention based spatiotemporal graph attention networks for traffic flow forecasting," *Inf. Sci.*, vol. 607, pp. 869–883, Aug. 2022.
[35] H. Chung and K.-S. Shin, "Genetic algorithm-optimized multi-channel convolutional neural network for stock market prediction," *Neural Comput. Appl.*, vol. 32, no. 12, pp. 7897–7914, Jun. 2020.
[36] C. Wang, Y. Chen, S. Zhang, and Q. Zhang, "Stock market index prediction using deep transformer model," *Expert Syst. Appl.*, vol. 208, Dec. 2022, Art. no. 118128.

JIAHAO CHEN received the B.S. degree in mathematics from Hubei University of Engineering, in 2020. He is currently pursuing the M.S. degree in mathematics with Wuhan University of Technology. His research interest includes the application of deep learning to time series.

LIANG XIE received the B.S. degree from Wuhan University of Technology, China, in 2009, and the Ph.D. degree from the Huazhong University of Science and Technology, China, in 2015. He is currently an Associate Professor with the School of Science, Wuhan University of Technology. His current research interests include machine learning, data mining, and information retrieval.

WENJING LIN received the B.S. degree in economics from Henan University, China, in 2021. She is currently pursuing the M.S. degree in science with Wuhan University of Technology. Her current research interests include quantitative trading, deep learning, and data mining.

YUCHEN WU received the bachelor's degree in mathematics from Wuhan University of Technology, in 2022, where he is currently pursuing the master's degree in mathematics. His research interest includes the application of deep learning to time series.

HAIJIAO XU received the Ph.D. degree in computer science and technology from the Huazhong University of Science and Technology, Wuhan, in 2015. His research interests include automatic image annotation, deep neural networks, and big data analysis.