DWISTGNN: Dynamic Spatio-Temporal Graph Neural Network with Wavelet Interaction for Multivariate Time Series Forecasting

Zhang Fengchen[a,*], Sun Fuchun[b], Lyu Xiao[a], Zhang Xian[a], Sun Lidong[a]

[a]*College of Electronic Engineering, Naval University of Engineering, Wuhan, 430030, Hubei, China*
[b]*Department of Computer Science and Technology, Tsinghua University, Haidian, 100084, Beijing, China*

Abstract

Multivariate Time Series (MTS) forecasting acts a vital role in numerous real-life scenarios. The emerging Spatio-Temporal Graph Neural Networks (STGNNs) have achieved brilliant progress by simultaneously modeling inter spatial relations between variates as a graph structure and capturing intra temporal patterns of series. However, the evolving spatial relations and the non-stationary temporal patterns pose challenges to accurate forecasting of MTS. To tackle these issues, we introduce a model named Dynamic Spatio-Temporal Graph Neural Network with Wavelet Interaction (DWIST-GNN). The Dynamic Graph Construction (DGC) module establishes the evolving graph structure by integrating learnable embeddings and dynamic node features. The Data Transform (DT) module first converts the input representation from time domain into frequency domain by utilizing Discrete Wavelet Transform (DWT) and then encodes the data with time position embeddings. The Frequency Interaction (FI) module facilitates deep interaction between different decomposed components to extract hidden spatio-temporal features. Abundant experiments are performed on eight benchmark datasets and significant performance reveals the effectiveness of the proposed model. Relevant codes and data are accessible at https://github.com/fczhang0606.

Keywords: MTS forecasting, dynamic graph construction, discrete wavelet transform, frequency interaction

1. Introduction

MTS refers to the structured data recorded during a period of time across multiple variates. As shown in **Fig.1**, when observed along the time axis, each variate's value changes according to certain temporal patterns. When observed along the spatial axis, these variates often exhibit spatial correlations at each timestamp. MTS forecasting is a field that studies how to model spatio-temporal relations and predict the consequent sequence of the given time series. Benefiting from the widely used low-cost sensors, MTS forecasting has found numerous applications in weather observation [1][2][3], energy prediction [4][5][6], traffic monitoring [7][8][9] and so on.

Attempting to explore the intricate spatio-temporal relations, MTS forecasting methods have undergone several development periods. Statistical methods [10][11][12][13][14] were first employed for the solution, but they fail to handle nonlinear patterns and unstable data in practice. Machine learning methods [15][16][17] were then developed to model more complex relations, but they require a great deal of manual feature engineerings. Recently, deep learning methods with all kinds of neural network variants have been widely utilized to capture high-dimensional non-linear dependencies. Recurrent Neural Networks (RNNs) [18][19] employ an inner hidden state variable to store historical information, combing the outer current input to describe temporal patterns. Convolutional Neural Networks (CNNs) [20] design a set of learnable kernels to pool multi-scale spatial features. By the integration of RNNs and CNNs [21][22], MTS forecasting can be addressed through simultaneously capturing temporal patterns and spatial relations. However, CNNs are constrained to process regular grid structures which limits the model to solve more irregular and realistic problems.

The emerging Graph Neural Networks (GNNs) [23][24][25] model the spatial relations of variates as a graph structure instead of grids, extending the data domain from Euclidean space to non-Euclidean space. Taking the generated graph and MTS data as inputs, STGNNs [26][27] combine temporal unit (like RNNs) and spatial unit GNNs to predict the consequent sequence of the given time series, achieving outstanding performance for MTS forecasting.

From the viewpoint of graph construction, existing STGNNs can be classified into predefined-graph methods, adaptive-graph methods and dynamic-

[*]Corresponding author

Email address: `21101002@nue.edu.cn` (Zhang Fengchen)

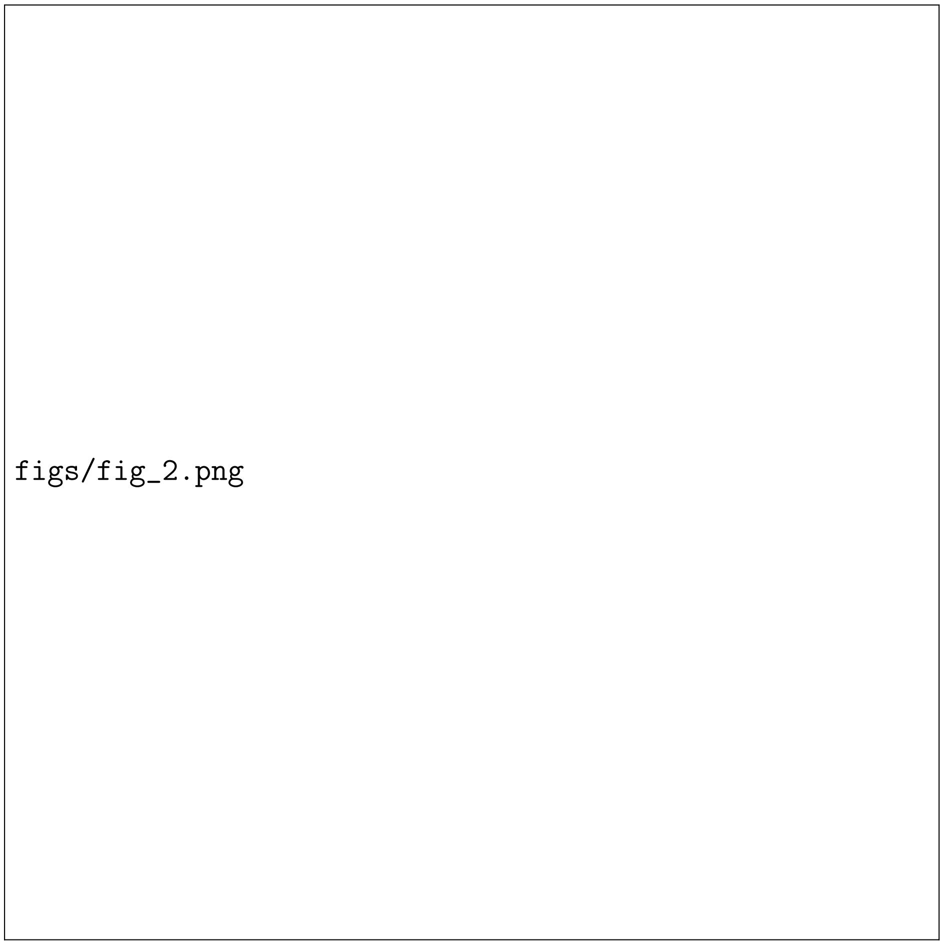


Figure 1: Spatial and Temporal Dependencies of MTS

graph methods. Predefined-graph methods [28] assume that the graph structure is constant and can be constructed in advance by prior knowledge. But generally there is a gap between prior knowledge and realistic spatial relations and these spatial relations are usually changing over time. Adaptive-graph methods [29][30][31] utilize the learnable embeddings to infer the optimized graph structure within each training iteration, aiming to reduce this gap. However, the adaptive graph structure is still considered static. Dynamic-graph methods [32][33][34][35] can learn the evolving graph structure at every timestamp by introducing the time-varying input nodes' features. But the constructed graphs across different timestamps can easily fall into unsmoothness due to discontinuous changes from nodes' features.

From the viewpoint of series decomposition, STGNNs can be divided into odd-even decomposition methods, trend-seasonal decomposition methods and frequency decomposition methods. Odd-even decomposition methods [36] separate the original time series into two subseries based on their indices, both of which share common temporal patterns. Trend-seasonal decomposition methods [37][38][39][40] disentangle the series into a trend- and a seasonal-component, respectively describing long- and short-term patterns. Frequency decomposition methods [41][42][43][44][45] decompose the time series into multiple frequency components, revealing distinct characteristics of different bands. The usually adopted Fourier Transform (FT) is unsuitable for handling non-stationary data and can only provide global frequency information of the time series.

From the viewpoint of model architecture, STGNNs can be grouped into coupled architecture methods and factorized architecture methods. Coupled architecture methods [28] embed the spatial modules into temporal modules to preserve the coupling relations. But such a fusion mechanism lacks flexibility to capture complex spatio-temporal relations. Factorized architecture methods [46][47] first separate spatial modules and temporal modules and then reorganize them in a serial or parallel manner. The separated modules first extract spatial or temporal features independently and then deliver their outputs as inputs to the subsequent module. Although factorized architecture is convenient to design, the interaction between the modules is insufficient.

To tackle the above issues, this work introduces a model named Dynamic Spatio-Temporal Graph Neural Network with Wavelet Interaction (DWIST-GNN). The Dynamic Graph Construction (DGC) module learns the evolving graph structure at every timestamp by integrating the learnable embeddings to describe the initial specific graph structure and the dynamic node features to reflect the gradual changes of the graph evolution. The Data Transform (DT) module is introduced to first decompose the time series into multiple frequency components via DWT to reveal distinct characteristics of different bands and then encode the transformation results with time position embeddings. The lower bands represent slower-varying smoothing trends, while the higher bands represent faster-varying drastic fluctuations. The Frequency Interaction (FI) module is designed to promote the deep interaction between different frequency components. Every component absorbs other component's information and all the processed components are then adaptively fused to acquire the representation for prediction.

Main contributions of this work are:

- A model named Dynamic Spatio-Temporal Graph Neural Network with Wavelet Interaction (DWISTGNN) is introduced to address the graph learning problem, series analysis problem and spatio-temporal patterns extraction problem in multivariate time series forecasting field.

- A joint framework is constructed consisting of Dynamic Graph Construction (DGC) module, Data Transform (DT) module and Frequency Interaction (FI) module to solve the aforementioned problems simultaneously and is trained in an end-to-end manner.

- Abundant experiments are conducted on eight real-world datasets from various fields. DWISTGNN achieves extraordinary performance, demonstrating the capability of the model.

2. Related Work

2.1. Multivariate Time Series Forecasting

A time series defines a cluster of sampling points measured and indexed in sequential order, depicting the changing patterns of one or more variates. A time series forecasting model aims to mine these underlying patterns from historical observations and analyze the future series.

Statistical methods are the earliest approaches for time series forecasting. Derived from autoregressive, moving average and their combination ARMA, autoregressive integrated moving average (ARIMA) [10] applied Box-Jenkins modeling methodology achieving a good performance for linear univariate time series. By treating the prediction as a classical regression task, support vector regression (SVR) [11] is also employed for univariate forecasting. As a generalization of AR-based methods, vector autoregression (VAR) [12] was first applied for multivariate time series forecasting. The models mentioned above all assume that there are only linear relations among the time series, which becomes the major limitation of these models. More efforts are then shifted to the kernel methods [13] and Gaussian Processes (GP) [14] to capture non-linearity.

Machine learning methods like Random Forest Regression (RFR) [15] and K-Nearest Neighbors (KNN) [16], were then developed to model more complex relations. They are more effective than the traditional groups, while requiring much effort to design manual features. Based on the theory that any mathematical function can be approximated by neural networks, deep learning methods are powerful tools to explore the non-linear relations. VARMLP [17] proposes a combined model integrating ARIMA and ANN to reduce the prediction error. RNNs like LSTM [18] and GRU [19] are widely applied to model temporal patterns. CNNs [20] are usually employed to extract the spatial dependencies. LSTNet [21] utilizes 1-D convolutional filters to uncover both variates' spatial relations and short-term temporal patterns and simultaneously employs RNN layers to model long-term temporal patterns. TPA-LSTM [22] integrates LSTM with an attention mechanism to address long-range forecasting and identify the principal temporal patterns. However, costly RNNs accumulate errors with every step due to iterative training and CNNs restrict the model to only process the regular grid structures, both of which constrain the development of the model.

A graph is a non-Euclidean data structure which is better suited for modeling irregular spatial relations in realistic scenarios. GNNs [23][24][25] are a cluster of models specifically operating on graph data. Through an information exchanging process, nodes iteratively aggregate messages from their neighbors and update their own embeddings to incorporate both local and global information. MTS forecasting can be formulated in a graph perspective naturally. At every timestamp, variates are treated as nodes and hidden dependencies are established as edges. As time goes by, both the values of the nodes and the strength of their connections change dynamically. STGNNs have been successfully employed for MTS forecasting in many fields [26][27]. However, significant challenges remain for better performance and broader applications.

2.2. Spatio-Temporal Graph Neural Networks

STGNN is a specialized neural network architecture simultaneously performing graph convolution for capturing non-Euclidean spatial dependencies and temporal modeling for capturing dynamic evolution patterns to learn the node representations to accomplish downstream tasks. The fundamental difference between STGNNs and traditional methods lies in the former's unified framework's ability to jointly extract spatio-temporal features, rather than analyzing spatial and temporal dependencies separately. Looking back along the timeline, the evolution of STGNNs can be classified into three main streams.

According to the mined graph structure, STGNNs can be categorized into predefined-graph methods, adaptive-graph methods and dynamic-graph methods. The predefined-graph methods build graph structures on the basis of prior knowledge like physical distance, node similarity and so on. DCRNN [28] extracts a directed weighted graph derived from the traffic network, describing asymmetric relations and connection strengths between nodes. The adaptive-graph methods still assume that the graph structure is static but acknowledge that prior knowledge can't fully represent the graph structure and try to bridge this gap in an adaptive manner. GWN [29] combines a predefined graph structure with an adaptive matrix which is generated by learnable embeddings to explore the unknown part of the structure. MTGNN [30] further utilizes the learnable embeddings to represent the entire graph structure. At every timestamp, the better-learned graph matrix participates into the spatial relation extraction and the optimal graph structure can be obtained when the training ends. In realistic applications, the spatial relations between variates usually evolve over time, beyond static model's expressing ability. To overcome this limitation, some dynamic-graph methods bring in time-varying nodes' features to generate the dynamic graph matrix. ASTGNN [32] developes a Dynamic Graph Convolution Net (DGCN) to measure relation strength adaptively. DSTAGNN [33] extracts the spatio-temporal aware distance (STAD) from historical data and uses the Wasserstein Distance to calculate the differences of probability distribution to construct the spatio-temporal aware graph. ESG [34] designs an Evolving Graph Structure Learner (EGL) that both considers the current dependency and the latest graph structure to recursively construct dynamic graph matrices.

According to the form of decomposed series, STGNNs can be categorized into odd-even decomposition methods, trend-seasonal decomposition methods and frequency decomposition methods. SCINet [36] implements a recursive architecture by decomposing the time series into odd- and even-subseries, introducing interactive learning to facilitate information exchange. Autoformer [37] disentangles the series into a trend- and seasonal-component for capturing long- and short-term patterns respectively and incorporates an Auto-Correlation mechanism to discover dependencies by aggregating representations at subseries level. D2STGNN [48] proposes a Decoupled Spatio-Temporal Framework (DSTF) that decomposes traffic signals into diffusion and inherent components and employs an Estimation Gate to dynamically evaluate the ratio between them. While the above methods handle time series forecasting in time domain, there are other works exploring solutions in frequency domain. StemGNN [41] first converts the series of each variate to spectral domain for orthogonality by leveraging a graph Fourier operation and then shifts the results to frequency domain. FEDFormer [42] designs a Frequency Enhanced Block (FEB) to project the input series to frequency space and utilizes a Frequency Enhanced Attention (FEA) to transform the Q, K and V via Fourier Transform or Wavelet Transform to simulate a traditional attention process. STWave [49] disentangles the non-stationary time series into different frequency components using DWT and adopts a dual-channel encoder to process the components separately.

According to the model architecture, STGNNs can be categorized into coupled architecture methods and factorized architecture methods. In coupled architecture methods, spatial modules are usually integrated into temporal modules. DCRNN [28] embeds a graph diffusion convolution operator into GRU instead of standard multiplications to couple spatial-temporal relations. In factorized architecture methods, spatial and temporal modules are treated as separate blocks that exchange features with each other in a serial or parallel way. T-GCN [46] adopts a serial manner by employing GRUs for temporal modeling. The interfaces of each GRU include both the output from previous GRU and the output from a spatial GCN layer, forming a recursive and sequential structure. STGCN [47] takes a parallel way to capture temporal patterns using 1-D causal convolution. Within a single step, the convolution operation can be implemented simultaneously at every timestamp position by considering both previous and current timestamp states. To address shallow interaction limitation in factorized architecture methods, STIDGCN [50] introduces a binary tree structure to perform hierarchical interactive learning between modules, enabling the identification of deeper spatio-temporal patterns.

3. Preliminary

Key background knowledge is provided to facilitate comprehension and **Table 1** lists the notations employed.

3.1. Definition: Dynamic Graph

A dynamic graph is formulated as $G(t) = (V(t), E(t), A(t))$ where $V(t)$ denotes the set of nodes (variates), $E(t)$ denotes the set of edges and $A(t)$ denotes the dynamic adjacency matrix which describes the instantaneous connections between nodes at timestamp t.

Assuming $V(t)$ is constant in this work, namely $V(t) = V \in R^N$. N denotes the fixed number of nodes. In contrast, the set of edges $E(t)$ always changes, resulting in a dynamic graph represented as $A(t) \in R^{N\times N}$.

3.2. Problem formulation

MTS can be denoted as a sequence of T timestamps of observations across multiple variates $X = \{X_1, X_2, ..., X_T\} \in R^{D\times N\times T}$. $X_t \in R^{D\times N}$ is the value matrix at timestamp t of all N variates with D feature dimension. For a certain task, N and D are fixed and $t \in T$.

We set an observation window W for historical time series and a forecasting horizon H for prediction. The historical series is $W_t = \{X_{t-W+1:t}\}$ and the forecasting series is $H_t = \{\widehat{X}_{t+1:t+H}\}$.

Given the actual values $Y = \{X_{T+1:T+H}\}$, the forecasting task falls into two categories. Single-step task targets to forecast $\widehat{Y}_s = \{\widehat{X}_{T+H}\}$ at specific timestamp $T + H$ and multi-step task targets to forecast $\widehat{Y}_m = \{\widehat{X}_{T+1:T+H}\}$ within a range H of timestamps.

The proposed model equals building an approximation $\widehat{Y} = model(X, \Theta)$. Θ is the trainable parameter set.

$$\Theta^\star = \arg\min_{\Theta} LOSS(Y, model(X, \Theta)) \tag{1}$$

3.3. Wavelet Transform

Wavelet Transform (WT) [51] was proposed as a mathematical tool for signal analysis by computing the similarity between the signals and a set of math functions which are modified through scaling and translation based on the wavelet basis. WT surpasses the widely adopted Fourier Transform (FT) as it supplies unified information in both time and frequency dimension, while FT can only offer a global frequency distribution without any time localization.

WT of a signal $\mathbf{x}$ is expressed as:

$$WT(x, s, \delta) =< x, \phi_{s,\delta} >= \frac{1}{\sqrt{s}} \int x(t)\phi^*(\frac{t-\delta}{s})dt$$

where $s > 0$ is the scale parameter, δ is the translation parameter and ϕ^* is the wavelet basis.

Continuous Wavelet Transform (CWT) analyzes signals across all possible scales and translations, ideal for detailed continuous time-frequency studies.

Discrete Wavelet Transform (DWT) adopts discrete scales and translations as parameters, having an advantage of computational efficiency in realistic applications. The common used wavelet basis functions include Haar, Daubechies, Coiflet, Symlet and Meyer.

Table 1: **Notations**

Notation	Definition
X	the input multivariate time series
X_t	the input tensor extracted from X at timestamp t
$Y, \widehat{Y}$	the ground truth and the predicted series
$\mathbf{g}, \mathbf{h}$	low- and high-pass filters in DWT
$X_{0,l}, X_{0,h}$	reconstructed components after DWT and IDWT
I_{day}, I_{week}	the day and week time indexing vectors
W_{day}, W_{week}	the day and week learnable projectors
E_{day}, E_{week}	the day and week embedding tensors
E_{time}	the time embedding tensor
X_l, X_h	frequency components encoded with time embeddings
E_s	the learnable embedding for specific graph construction
A_s	the specific graph
A_d	the gradual changing graph
ΔA	the compensatory graph
A	the entire dynamic graph
$X_l^{(2)}, X_h^{(2)}$	frequency components after two rounds of interaction

4. Methodology

4.1. Architecture

DWISTGNN adopts an encoder-decoder design, as displayed in **Fig.2**. The encoder comprises most of the proposed modules to refine the input into deeper semantic representations. The decoder then accepts these intermediate representations as input to generate the final predictions.

In the first stage, the raw input series is decomposed into two frequency components by applying DWT twice and IDWT. Each component then undergoes a position embedding process to incorporate temporal knowledge for enhancing periodicity. Simultaneously, the raw input series is also lead to the DGC module to build the dynamic graph structure. In the second stage, the time-embedded frequency components and the generated dynamic graph adjacency matrix are both utilized by the FI module to carry out a two-round interactive learning process. Subsequently, the components that have fully exchanged features with each other are adaptively fused to obtain the integrated representation. Finally, the decoder employs a GLU unit to selectively filter information from the integrated representation. The predicted output is obtained after a regression layer to align with the horizon.

figs/fig_1.png

Figure 2: Overview Framework of DWISTGNN

4.2. Data Transform

4.2.1. DWT and Inverse DWT

DWT can be performed multiple times recursively. With each iteration, the low- and high-pass decomposition filters $\mathbf{g}$ and $\mathbf{h}$ are used to decompose the input into components of distinct resolutions. The results filtered by two filters are respectively denoted as cA (Approximation Coefficients) and cD (Detail Coefficients). The approximation coefficients cA represent the coarse-grained patterns revealing the overall trend, while the detail coefficients cD depict the fine-grained details of the signal.

DWT can be formulated as cD_l, $cA_l = \text{DWT}(cA_{l-1})$.

$$\begin{aligned} cD_l = \mathbf{h} \star cA_{l-1} = \sum_{\lambda=1}^{\Lambda} h[2s-\lambda] cA_{l-1}[\lambda] \quad s = 1, 2, ..., \frac{\Lambda}{2} \\ cA_l = \mathbf{g} \star cA_{l-1} = \sum_{\lambda=1}^{\Lambda} g[2s-\lambda] cA_{l-1}[\lambda] \quad s = 1, 2, ..., \frac{\Lambda}{2} \end{aligned} \tag{2}$$

where l indicates the l-th decomposition, Λ represents the length of cA_{l-1}, s represents the scale and $cA_0 = \mathbf{x}$.

Given the raw input data $X \in R^{D \times N \times T}$, DWISTGNN conducts DWT twice to derive the approximation coefficient and multiple detail coefficients, listed as $\{X_{2,l}, X_{2,h}, X_{1,h}\}$:

$$\begin{aligned} X_{1,l} &= \mathbf{g} \star X \\ X_{1,h} &= \mathbf{h} \star X \\ X_{2,l} &= \mathbf{g} \star X_{1,l} \\ X_{2,h} &= \mathbf{h} \star X_{1,l} \end{aligned} \tag{3}$$

The standard Inverse DWT (IDWT) is used to recover the approximation coefficients of the higher-level signal from its decomposed components and can be defined as

$$\begin{aligned} \widehat{cA_{l-1}} &= IDWT(\widehat{cD_l}, \widehat{cA_l}) \\ &= \tilde{\mathbf{h}} \star \widehat{cD_l} + \tilde{\mathbf{g}} \star \widehat{cA_l} \end{aligned} \tag{4}$$

where $\tilde{\mathbf{g}}$ and $\tilde{\mathbf{h}}$ are reconstruction filters. The original $\tilde{\mathbf{x}}$ is recovered by applying IDWT recursively on all decomposed components.

DWISTGNN divides the decomposed coefficients into two parts to apply IDWT separately. One part recorded as $\{X_{2,l}, 0, 0\}$ includes the approximation coefficient only and is padded with zeros to align with the original list's length to reconstruct $X_{0,l}$. The other part recorded as $\{0, X_{2,h}, X_{1,h}\}$ contains all detail coefficients and is also padded with zeros accordingly to reconstruct $X_{0,h}$. Theoretically, the original signal $\mathbf{x} = X_{0,l} + X_{0,h}$, in which $X_{0,l}$ and $X_{0,h}$ respectively convey the low- and high-resolution information.

$$\begin{aligned} X_{0,l} &= W^{\tilde{\mathbf{g}}}(\tilde{\mathbf{g}} \star (\tilde{\mathbf{g}} \star X_{2,l})) + b^{\tilde{\mathbf{g}}} \\ X_{0,h} &= W^{\tilde{\mathbf{h}}}(\tilde{\mathbf{g}} \star (\tilde{\mathbf{h}} \star X_{2,h}) + \tilde{\mathbf{h}} \star X_{1,h}) + b^{\tilde{\mathbf{h}}} \end{aligned} \tag{5}$$

4.2.2. Position Embedding

Position embedding integrates temporal location information into raw data to uncover the position relations of the time series.

First, we assign indexes of "day" and "week" to every timestamp position in the series, resulting in $I_{day} \in R^{N\times T}$ and $I_{week} \in R^{N\times T}$.

$$\begin{aligned} I_{day} &= Duplicate(\{i\%G/G\}_{i\in T})^N \qquad && \in R^{N\times T} \\ I_{week} &= Duplicate(\{i//G\%7\}_{i\in T})^N \qquad && \in R^{N\times T} \end{aligned} \tag{6}$$

i denotes the index of timestamp. G denotes the granularity, namely the totality of timestamps in a day. I_{day} and I_{week} denote which hour and day that every timestamp belongs to.

Then we define two learnable temporal projectors, i.e., $W_{day} \in R^{C\times G}$ and $W_{week} \in R^{C\times 7}$ to learn the periodicities adaptively. $E_{day} \in R^{C\times N\times T}$ and $E_{week} \in R^{C\times N\times T}$ are generated by indexing learnable embedding matrix W_{day} and W_{week} with every element of I_{day} and I_{week} respectively. C represents the expanded dimension. The combined temporal periodic embedding $E_{time} \in R^{C\times N\times T}$ is

$$E_{time} = (E_{day} + E_{week}) \quad \in R^{C\times N\times T} \tag{7}$$

Subsequently, the components $X_{0,l} \in R^{D\times N\times T}$, $X_{0,h} \in R^{D\times N\times T}$ ascend their dimension to $R^{C\times N\times T}$ to enrich semantic information through a fully connected layer respectively. The final X_l and X_h are yielded by concatenating E_{time} along the channel dimension to absorb the prior position information.

$$\begin{aligned} X_l &= (W_l X_{0,l})||E_{time} \qquad && \in R^{2C\times N\times T} \\ X_h &= (W_h X_{0,h})||E_{time} \qquad && \in R^{2C\times N\times T} \end{aligned} \tag{8}$$

4.3. Dynamic Graph Construction

Generally, the spatial relations between variates change over time, which start from an unknown specific graph structure and evolve to the present structure. Assuming the change during this period is continuous, the current structure can be considered as a sum of three distinct structure components. The initial specific graph structure component is optimized by the learnable embeddings. The gradual changing graph component describing the evolution is captured by tracking the dynamic node-level inputs. A third compensatory component employs a GRU to integrate the two aforementioned components and applies a self-attention mechanism to ensure a smoother transition process.

To learn the initial specific graph, a trainable embeddings $E_s \in R^{d\times N}$ is employed to construct the structure adaptively. The dimension d is usually far less than N, helpful to reduce the computational cost. The adjacency matrix of initial specific graph is inferred:

$$A_s = SoftMax(ReLU(E_s \cdot E_s^{'})) \tag{9}$$

$E_s^{'}$ is the transpose of E_s. ReLU blocks the weak connections and SoftMax normalizes the results.

The gradual changes of the graph evolution can be associatied with the node-level inputs $X_t \in R^{D\times N}$. Similarly,

$$A_d = SoftMax(ReLU(X_t \cdot X_t^{'})) \tag{10}$$

By integrating the learnable embeddings dictionary E_s and the dynamic node-level inputs X_t with a gated mechanism, we can get a fused hidden representation $h_t \in R^{d\times N}$.

$$\begin{aligned} r_t &= \sigma(W_r \cdot E_s + U_r \cdot X_t) \\ z_t &= \sigma(W_z \cdot E_s + U_z \cdot X_t) \\ \widehat{h_t} &= tanh(W_h \cdot X_t + r_t(U_h \cdot E_s)) \\ h_t &= (1 - z_t) \cdot E_s + z_t \cdot \widehat{h_t} \end{aligned} \tag{11}$$

The hidden intermediate representation h_t undergoes a self-attention to produce a dynamic compensatory matrix ΔA.

$$\begin{aligned} Q &= W_q \cdot h_t \\ K &= W_k \cdot h_t \\ V &= W_v \cdot h_t \\ \Delta A &= Dropout(\frac{Q \cdot K^{'}}{\sqrt{d}}) \end{aligned} \tag{12}$$

where $K^{'}$ is the transpose of K and W_q, W_k and W_v are linear projectors.

Finally, the dynamic adjacency matrix is generated by integrating the above three matrices.

$$A = SoftMax(A_s + A_d + \Delta A) \tag{13}$$

4.4. Frequency Interaction

4.4.1. Frequency Temporal Convolution

Temporal Convolution (TC) aims to extract intra temporal patterns. As shown in **Fig.2**, four TC modules labeled as TC1, TC2, TC3 and TC4 are embedded into the FI module to jointly capture the global representation. Different labeled TC modules have different inputs and outputs, while all of them share the same structure. TC is composed of two convolutional layers with different kernels:

$$H_{out} = \sigma(Conv_2(\sigma(Conv_1(H_{in})))) \tag{14}$$

H_{in} and H_{out} represents the interfaces of TC and σ is the activation function.

4.4.2. Frequency Spatial Convolution

Spatial Convolution (SC) adopts a hix-hop graph convolution aiming to incorporate a variate's feature with its neighbors' to extract the spatial relations.

$$\begin{aligned} H^{(k)} &= (1-\beta)AH^{(k-1)} + \beta H_{in} \\ H_{out} &= \sum_{k=0}^{K} W^{(k)} H^{(k)} \end{aligned} \tag{15}$$

$H_{in} = H^{(0)}$ denotes the input of SC, usually fed directly by the previous TC. β is the controlled ratio to retain a portion of raw information. A is the dynamic adjacency matrix generated by the DGC module. K is the depth of propagation.

4.4.3. Frequency Interaction and Fusion

Frequency Interaction completes an exchange of spatio-temporal features between different frequency components. The components obtained by data transform can be denoted as $X_l \in R^{2C\times N\times T}$ and $X_h \in R^{2C\times N\times T}$, which are inputs of the FI module. In the first round of interaction, each component absorbs the processed information from other components' spatio-temporal extraction $X_*^{(ts)}$. Similarly in the second round of interaction, components obtained by the first round $X_*^{(1)}$ continuously incorporate information from other components' another extraction. This two-round interaction promotes the deep multi-resolution spatio-temporal exchange of different components and can be defined as follows:

$$\begin{aligned}
X_l^{(ts)} &= tanh(SC(TC1(X_l))) \\
X_h^{(ts)} &= tanh(SC(TC2(X_h))) \\
X_l^{(1)} &= X_h^{(ts)} \otimes X_l \\
X_h^{(1)} &= X_l^{(ts)} \otimes X_h \\
X_l^{(1ts)} &= tanh(SC(TC3(X_l^{(1)}))) \\
X_h^{(1ts)} &= tanh(SC(TC4(X_h^{(1)}))) \\
X_l^{(2)} &= X_h^{(1ts)} \otimes X_l^{(1)} \\
X_h^{(2)} &= X_l^{(1ts)} \otimes X_h^{(1)}
\end{aligned} \tag{16}$$

Different frequency components represent distinct temporal patterns in different bands. The fusion of the components represents holistic characteristics of the series. Since the amplitudes of the components are varisized and unknown, we adopt an adaptive fusion mechanism to integrate the components to reconstruct the time series.

$$X_{out} = sigmoid(\mu)X_l^{(2)} + (1 - sigmoid(\mu))X_h^{(2)} \tag{17}$$

where μ is a learnable parameter to control the fusion ratio.

4.5. Decoder

The X_{out} generated by the encoder first passes through a GLU, which processes the representation into two branches. One branch is activated by a sigmoid function to produce a gating signal, which then selectively modulates the other branch's representation to filter out irrelevant information. The output of this gating operation is then integrated with the original input to mitigate the gradient vanishing problem. Finally, a regression layer utilizing a convolution unit reduces the channel dimension from $2C$ to H to match the forecasting horizon, yielding the final predictions. The decoder can be formulated as follows:

$$\widehat{Y} = Conv(MLP(ReLU(\sigma(MLP(X_{out})) \otimes MLP(X_{out}))) + X_{out}) \tag{18}$$

$\widehat{Y} \in R^{N \times H}$ denotes the predicted time series.

We choose RRSE loss as training objective for single-step forecasting task and MAE loss for multi-step forecasting task:

$$\begin{aligned} LOSS_s(Y, X, \Theta) &= RRSE(Y, \widehat{Y}_s) \\ LOSS_m(Y, X, \Theta) &= MAE(Y, \widehat{Y}_m) \end{aligned} \tag{19}$$

The definition of RRSE and MAE can be found in Experiment section and the detailed learning process of DWISTGNN is shown in **Algorithm 1**.

Algorithm 1: DWISTGNN

```
Input: Dataset D
Parameters: $wavelet$, $epoch$, $iter$, $B$, $W$, $H$, $C$, $d$, $K$, $dropout$, $\alpha$, $\gamma$
Output: The learned DWISTGNN model
Init $E_{time}$, $E_s$ and $\Theta$ ;
for $i \leftarrow 1$ to $epoch$ do
    for $j \leftarrow 1$ to $iter$ do
        Sample $X \in R^{B \times D \times N \times W}$, $Y \in R^{B \times N \times H}$ from $D$ ;
        $X_{0,l}, X_{0,h} \leftarrow \{X\}$                      ▷ Eq. (3)(5): DWT and IDWT
        $X_l, X_h \leftarrow \{X_{0,l}, X_{0,h}, E_{time}\}$
                                                 ▷ Eq. (6)-(8): Position Embedding
        $A \leftarrow \{X, E_s\}$        ▷ Eq. (9)-(13): Dynamic Graph Construction
        $X_l^{(2)}, X_h^{(2)} \leftarrow \{X_l, X_h, A\}$
                                               ▷ Eq. (14)-(16): Frequency Interaction
        $X_{out} \leftarrow \{X_l^{(2)}, X_h^{(2)}\}$                  ▷ Eq. (17): Adaptive Fusion
        $\widehat{Y} \leftarrow \{X_{out}\}$                                  ▷ Eq. (18): Decoder
        $L \leftarrow \{Y, \widehat{Y}\}$                               ▷ Eq. (19): Loss Function
        $\Theta \leftarrow \{L, \alpha, \gamma\}$                       ▷ Eq. (1): Parameters Update
        $j \leftarrow j + 1$ ;
    end
    $i \leftarrow i + 1$ ;
end
return $\Theta^\star$ ;
```

5. Experiment

5.1. General Settings

5.1.1. Dataset Statistics

Table 2 summarizes the corpus statistics for two benchmark dataset groups, i.e., single- and multi-step forecasting tasks. Each dataset is chronologically split into training, validation, and test sets in a 6:2:2 ratio. For single-step tasks, the look-back size is 168. The prediction horizons in independent trained cases are set to 3/6/12/24 and the prediction size is 1. For multi-step tasks, the look-back size is 12 and the prediction series consists of 12 successive steps.

Table 2: **Dataset Statistics**

Task Type	Dataset	Partition	Variates	Timestamps	Granularity	Input Size	Predicted Size
Single-step	Electricity	0.6/0.2/0.2	321	26304	1 h	168	3/6/12/24
	Exchange-Rate	0.6/0.2/0.2	8	7588	1 day	168	3/6/12/24
	Solar-Energy	0.6/0.2/0.2	137	52560	10 mins	168	3/6/12/24
	Traffic	0.6/0.2/0.2	862	17544	1 h	168	3/6/12/24
Multi-step	PeMS03	0.6/0.2/0.2	358	26208	5 mins	12	0-12
	PeMS04	0.6/0.2/0.2	307	16992	5 mins	12	0-12
	PeMS07	0.6/0.2/0.2	883	28224	5 mins	12	0-12
	PeMS08	0.6/0.2/0.2	170	17856	5 mins	12	0-12

- **Electricity**: This dataset is a widely used benchmark particularly for energy consumption prediction. The data recorded at 1-hour intervals from 321 clients spans from 2011 to 2014 and contains 26,304 timestamps.

- **Exchange-Rate**: This dataset is a widely used benchmark particularly for financial data prediction. The data recorded at 1-day intervals from 8 countries spans from 1990 to 2016 and contains 7,588 timestamps.

- **Solar-Energy**: This dataset is a widely used benchmark particularly for renewable energy forecasting. The data recorded at 10-minute intervals from 137 solar power facilities in Alabama in 2006 contains 52,560 timestamps.

- **Traffic**: This dataset is a widely used benchmark particularly for traffic flow forecasting. The data recorded at 1-hour intervals from 862 detectors in San Francisco spans from 2015 to 2016 and contains 17,544 timestamps.

- **PeMS03**: This dataset contains transportation data from Los Angeles. The data collected at 5-minute intervals from 358 detectors spans from 2018.09.01 to 2018.11.30 and includes 26,208 timestamps.

- **PeMS04**: This dataset contains transportation data from San Francisco. The data collected at 5-minute intervals from 307 detectors spans from 2018.01.01 to 2018.02.28 and includes 16,992 timestamps.

- **PeMS07**: This dataset contains transportation data from California District 7. The data collected at 5-minute intervals from 883 detectors spans from 2017.05.01 to 2017.08.31 and includes 28,224 timestamps.

- **PeMS08**: This dataset contains transportation data from San Bernardino. The data collected at 5-minute intervals from 170 detectors spans from 2016.07.01 to 2016.08.31 and includes 17,856 timestamps.

5.1.2. Hyperparameters and Metrics

Table 3 summarizes the hyperparameters used in DWISTGNN. Epoch (E) limits the iterations to prevent overfitting. Wavelets including coif1, db1 and sym2 of distinct characteristics are used to implement DWT across different datasets. The scale of Batch Size (B) affects the model's generalization. Window (W) refers to the length of the observed series. Horizon (H) indicates the steps ahead to be predicted. Channel (C) means the expanded dimensionality compared with the input. Dimension (d) corresponds to the dimensionality of the learnable graph embeddings. Diffusion Depth (K) determines the extent of neighborhood aggregation in diffusion convolution. Dropout rate is applied to mitigate overfitting. Learning Rate (α) controls the step size during gradient descent. Weight Decay (γ) serves as a regularization term that dynamically adjusts α to balance convergence speed and training stability.

The hardware environment of experiments consists of an Intel Core i9-13900KF CPU @ 3.0 GHz and an NVIDIA GeForce RTX 4090 GPU with 24 GB of memory. The software environment takes Python 3.8 and the PyTorch 1.12.1 framework.

Table 3: **Hyperparameters**

Hyperparameters	Electricity	Exchange-Rate	Solar-Energy	Traffic	PeMS03	PeMS04	PeMS07	PeMS08
Wavelet	db1	db1	sym2	db1	db1	db1	coif1	db1
E	1000	1000	1000	1000	1000	1000	1000	1000
B	64	64	64	64	96	96	96	96
W	168	168	168	168	12	12	12	12
H	3/6/12/24	3/6/12/24	3/6/12/24	3/6/12/24	0-12	0-12	0-12	0-12
C	64	32	64	128	64	64	128	64
d	10	10	10	10	10	10	10	10
K	2	1	2	2	1	1	2	1
dropout	0.3	0.5	0.3	0.3	0.5	0.5	0.5	0.5
α	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
γ	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005

Supposing $\{y = y_1, y_2, ..., y_n\}$ denotes the actual values, $\{\widehat{y} = \widehat{y_1}, \widehat{y_2}, ..., \widehat{y_n}\}$ denotes the forecasted series and Ψ denotes the sample set.

Mean Absolute Error (MAE)

$$MAE(y, \widehat{y}) = \frac{1}{|\Psi|} \sum_{i \in \Psi} |y_i - \widehat{y_i}|$$

Mean Absolute Percentage Error (MAPE)

$$MAPE(y, \widehat{y}) = \frac{1}{|\Psi|} \sum_{i \in \Psi} |\frac{y_i - \widehat{y_i}}{y_i}|$$

Root Mean Square Error (RMSE)

$$RMSE(y, \widehat{y}) = \sqrt{\frac{1}{|\Psi|} \sum_{i \in \Psi} (y_i - \widehat{y_i})^2}$$

Root Relative Squared Error (RRSE)

$$RRSE(y, \widehat{y}) = \sqrt{\frac{\sum_{i \in \Psi} (y_i - \widehat{y_i})^2}{\sum_{i \in \Psi} (y_i - \overline{\widehat{y}})^2}}$$

Correlation Coefficient (CORR)

$$CORR(y, \widehat{y}) = \frac{\sum_{i \in \Psi} (y_i - \overline{y})(\widehat{y_i} - \overline{\widehat{y}})}{\sqrt{\sum_{i \in \Psi} (y_i - \overline{y})^2 (\widehat{y_i} - \overline{\widehat{y}})^2}}$$

Lower MAE, MAPE, RMSE and RRSE and higher CORR indicate better model performance.

5.1.3. Baselines

DWISTGNN is compared with the following baseline models.

Single-step Forecasting Model

- AR [10]: A classical forecasting model that assumes the current value of a variate depends linearly on its own previous values plus a stochastic error term.
- GP [14]: It treats a time series as a sample from a Gaussian process and uses Bayes theorem to derive a posterior distribution for point forecasting.
- VARMLP [17]: This hybrid time series forecasting model combines VAR and MLP, where the VAR component captures linear dependencies and the MLP component captures nonlinear dependencies.
- RNN-GRU [19]: This network introduces gating mechanisms to address the constraints of traditional RNNs, like the vanishing gradient problem.
- LSTNet [21]: A method effectively analyzes both long- and short-term temporal dependencies by combining RNNs and CNNs.
- TPA-LSTM [22]: An advanced forecasting model that integrates LSTM with a Temporal Pattern Attention (TPA) mechanism, dynamically focusing on critical temporal patterns to enhance prediction accuracy.
- MTGNN [30]: A classical model equipped with a graph learning module that learns the relations between variates via a self-adaptive adjacency matrix.
- Informer [52]: A variant of Transformer to address high computational cost and memory usage by employing efficient attention mechanisms.
- Autoformer [37]: By integrating decomposition and auto-correlation mechanisms, the model effectively captures complex temporal patterns.
- GTS [53]: GTS proposes a method for the joint learning of an unknown graph structure and a GNN model by optimizing the expectation of the performance.

- ESG [34]: A framework that jointly learns the evolutionary graph structure and multi-scale temporal patterns.
- SCINet [36]: A model designed to capture complex temporal patterns through hierarchical decomposition and multi-resolution analysis.

Multi-step Forecasting Model

- VAR [12]: A statistical forecasting model in which each variate is expressed as the linear combination of its own lags and the lags of every other variate.
- SVR [11]: A method that projects the series into a latent space using kernels to analyze the relations between variates.
- LSTM [18]: A specialized type of RNNs aimed to model temporal dependencies in series.
- DCRNN [28]: DCRNN integrates diffusion convolution into GRU to couple the spatio-temporal features.
- STGCN [47]: A fully convolutional model that combines GCNs and TCNs to model spatio-temporal dynamics.
- ASTGCN [54]: A model leveraging attention mechanisms to dig relations in complex systems.
- GWN [29]: The model utilizes TCNs to extract temporal patterns and GCNs to capture spatial relations through a learnable adaptive adjacency matrix.
- STSGCN [55]: It constructs multiple localized graphs to process spatial and temporal features synchronously within a unified framework.
- DSTAGNN [33]: DSTAGNN proposes a novel architecture that captures dynamic spatial relations using an improved multi-head attention mechanism and models long-term dynamic temporal patterns through multi-receptive field features.
- D2STGNN [48]: It decomposes the time series into a diffusion component and an inherent component. The ratio of the two components is dynamically estimated to promote the prediction performance.

- STWave [49]: By applying DWT multiple times, the model derives multi-resolution temporal patterns. The resulting features are integrated with spatial features to jointly inform the final prediction.

- STIDGCN [50]: A model introducing a hierarchical tree architecture and embedding dynamic graph construction into a feature interaction.

5.2. Result Analysis

5.2.1. Overall Comparison

Table 4 and **5** record the detailed results for single- and multi-step forecasting tasks respectively.

Single-step Forecasting Results The results of DWISTGNN compared with 12 baselines on the four datasets are summarized in **Table 4**. From a dataset perspective, DWISTGNN exceeds all baselines on the Traffic dataset, yielding a 12.27‰ RRSE reduction and a 7.33‰ CORR improvement across all horizons compared to the second-best results. It also obtains the optimal results at 3/6/12 horizons on the Solar-Energy dataset with a 29.96‰ RRSE reduction and a 1.29‰ CORR improvement. On the Exchange-Rate dataset, DWISTGNN gets the top rank in 3 cases and the second rank in the remaining 5 cases, guessing that the model may be slightly overfitting on this small dataset, which only contains 8 variates. On the Electricity dataset, DWISTGNN performs well in terms of a 8.67‰ RRSE reduction, but the CORR results are less satisfactory with a 9.21‰ deterioration. This may be because the Electricity dataset contains many patterns that are not easily captured by wavelet extraction. From a model perspective, machine learning models like LSTNet and TPA-LSTM make remarkable progress over traditional statistical models like AR, GP and VARMLP. MTGNN and GTS extract hidden spatial dependencies represented by a graph structure. ESG goes further by capturing evolutionary graph structures, which is similar to the DGC module in DWISTGNN. Autoformer resolves the input into trend and seasonal components to capture temporal patterns at different periods, while DWISTGNN applies DWT to obtain multi-frequency components. SCINet employs an interactive learning architecture which is similar to the FI module in DWISTGNN. However, all the above models do not perform as well as DWISTGNN, indicating that its modified and combined approach works better.

Multi-step Forecasting Results The results of DWISTGNN compared with 12 baselines on the four datasets are summarized in **Table 5**. From a dataset perspective, DWISTGNN obtains the highest accuracy on the PeMS04/07/08 datasets, reducing MAE by 14.03‰, RMSE by 9.65‰ and MAPE by 16.25‰. On the PeMS03 dataset, DWISTGNN achieves the best MAE result with a 1.36‰ reduction, the second best in RMSE and a mid-level result in MAPE. Repeated experiments with different sets of hyper-parameters on PeMS03 have shown that it's quite challenging to achieve the best performance across all three metrics simultaneously. Typically, an improvement in one metric leads to a decline in another, unless the model makes a significant breakthrough over previous approaches as DWISTGNN does on the PeMS04/07/08 datasets. From a model perspective, the emergence of DCRNN and STGCN indicates that the STGNN-based models have gradually become mainstream in spatio-temporal forecasting tasks, replacing traditional statistical models (VAR and SVR) and machine learning models (LSTM and GRU). Models such as GWN, STSGCN and DSTAGNN have made efforts to explore spatial dependencies, which inspired the design of the DGC module in DWISTGNN. D2STGNN decomposes the time series into two components and STWave applies DWT to extract multi-resolution temporal patterns, both of which share similarities with the DWT module in DWISTGNN. STIDGCN incorporates dynamic graph convolution into an interactive learning framework, which aligns with the FI module in DWISTGNN. However, all the aforementioned models underperform compared to DWISTGNN, demonstrating the effectiveness of the proposed integrated architecture.

Table 4: **Single-step Forecasting Results**

Methods	Metrics	Solar-Energy Horizon				Electricity Horizon				Exchange-Rate Horizon				Traffic Horizon			
		3	6	12	24	3	6	12	24	3	6	12	24	3	6	12	24
AR	RRSE	0.2435	0.3790	0.5911	0.8699	0.0995	0.1035	0.1050	0.1054	0.0228	0.0279	0.0353	0.0445	0.5991	0.6218	0.6252	0.6293
	CORR	0.9710	0.9263	0.8107	0.5314	0.8845	0.8632	0.8591	0.8595	0.9734	0.9656	0.9526	0.9357	0.7752	0.7568	0.7544	0.7519
GP	RRSE	0.2259	0.3286	0.5200	0.7973	0.1500	0.1907	0.1621	0.1273	0.0239	0.0272	0.0394	0.0580	0.6082	0.6772	0.6406	0.5995
	CORR	0.9751	0.9448	0.8518	0.5971	0.8670	0.8334	0.8394	0.8818	0.8713	0.8193	0.8484	0.8278	0.7831	0.7406	0.7671	0.7909
VARMLP	RRSE	0.1922	0.2679	0.4244	0.6841	0.1393	0.1620	0.1557	0.1274	0.0265	0.0394	0.0407	0.0578	0.5582	0.6579	0.6023	0.6146
	CORR	0.9829	0.9655	0.9058	0.7149	0.8708	0.8389	0.8192	0.8679	0.8609	0.8725	0.8280	0.7675	0.8245	0.7695	0.7929	0.7891
RNN-GRU	RRSE	0.1932	0.2628	0.4163	0.4852	0.1102	0.1144	0.1183	0.1295	0.0192	0.0264	0.0408	0.0626	0.5358	0.5522	0.5562	0.5633
	CORR	0.9823	0.9675	0.9150	0.8823	0.8597	0.8623	0.8472	0.8651	0.9786	0.9712	0.9531	0.9223	0.8511	0.8405	0.8345	0.8300
LSTNet	RRSE	0.1843	0.2559	0.3254	0.4643	0.0864	0.0931	0.1007	0.1007	0.0226	0.0280	0.0356	0.0449	0.4777	0.4893	0.4950	0.4973
	CORR	0.9843	0.9690	0.9467	0.8870	0.9283	0.9135	0.9077	0.9119	0.9735	0.9658	0.9511	0.9354	0.8721	0.8690	0.8614	0.8588
TPA-LSTM	RRSE	0.1803	0.2347	0.3234	0.4389	0.0823	0.0916	0.0964	0.1006	0.0174	0.0241	0.0341	0.0444	0.4487	0.4658	0.4641	0.4765
	CORR	0.9850	0.9742	0.9487	0.9081	0.9439	0.9337	0.9250	0.9133	0.9790	0.9709	0.9564	0.9381	0.8812	0.8717	0.8717	0.8629
MTGNN	RRSE	0.1778	0.2348	0.3109	0.4270	0.0745	0.0878	0.0916	0.0953	0.0194	0.0259	0.0349	0.0456	0.4162	0.4754	0.4461	0.4535
	CORR	0.9852	0.9726	0.9509	0.9031	0.9474	0.9316	0.9278	0.9234	0.9786	0.9708	0.9551	0.9372	0.8963	0.8667	0.8794	0.8810
Informer	RRSE	0.2134	0.2701	0.4331	0.7017	0.1524	0.1932	0.1748	0.2110	0.1392	0.1548	0.1793	0.1998	0.5175	0.5258	0.5533	0.5883
	CORR	0.9715	0.9549	0.8985	0.7921	0.8858	0.8660	0.8585	0.8347	0.9473	0.9207	0.8817	0.7715	0.8515	0.8465	0.8279	0.8033
Autoformer	RRSE	0.1935	0.2604	0.3959	0.6064	0.1458	0.1555	0.1541	0.1754	0.0400	0.0481	0.0638	0.0651	0.5368	0.5462	0.5623	0.6020
	CORR	0.9784	0.9651	0.9111	0.8180	0.9032	0.8957	0.8907	0.8732	0.9458	0.9197	0.9054	0.8952	0.8268	0.8191	0.8082	0.7757
GTS	RRSE	0.1842	0.2691	0.3259	0.4796	0.0790	0.0884	0.0957	0.0951	0.0180	0.0260	0.0333	0.0442	0.4665	0.4779	0.4792	0.4766
	CORR	0.9842	0.9645	0.9481	0.8678	0.9291	0.9187	0.9135	0.9234	0.9798	0.9724	**0.9601**	**0.9418**	0.8693	0.8582	0.8589	0.8573
ESG	RRSE	0.1708	0.2278	0.3073	0.4101	**0.0718**	0.0844	0.0898	0.0962	0.0181	0.0246	0.0345	0.0468	0.4329	0.4429	0.4566	0.4622
	CORR	0.9865	0.9743	0.9519	0.9100	**0.9494**	0.9372	**0.9321**	**0.9279**	0.9792	0.9717	0.9564	0.9392	0.8891	0.8833	0.8757	0.8732
SCINet	RRSE	0.1775	0.2301	0.2997	**0.4081**	0.0740	0.0845	0.0929	0.0967	**0.0171**	**0.0240**	**0.0331**	**0.0436**	0.4216	0.4414	0.4495	0.4453
	CORR	0.9853	0.9739	0.9550	**0.9112**	**0.9494**	**0.9387**	0.9305	0.9270	0.9787	0.9704	0.9553	0.9396	0.8920	0.8809	0.8772	0.8825
DWISTGNN	RRSE	**0.1629**	**0.2190**	**0.2982**	0.4264	0.0726	**0.0823**	**0.0896**	**0.0944**	0.0172	**0.0240**	0.0333	0.0440	**0.4183**	**0.4350**	**0.4394**	**0.4434**
	CORR	**0.9877**	**0.9768**	**0.9551**	0.9018	0.9450	0.9321	0.9202	0.9164	**0.9819**	**0.9735**	0.9580	0.9399	**0.9006**	**0.8902**	**0.8870**	**0.8831**

Datasets		Methods												
		VAR	SVR	LSTM	DCRNN	STGCN	ASTGCN	GWN	STSGCN	DSTAGNN	D2STGNN	STWave	STIDGCN	DWISTGNN
PEMS03	MAE	23.65	21.97	21.33	17.99	17.55	17.34	19.12	17.48	15.57	14.88	14.93	14.76	**14.74**
	RMSE	38.26	35.29	35.11	30.31	30.42	29.56	32.77	29.21	27.21	26.01	26.50	**24.59**	25.45
	MAPE%	24.51	21.51	23.33	18.34	17.34	17.21	18.89	16.78	**14.68**	15.12	15.05	15.28	15.24
PEMS04	MAE	24.54	28.70	26.77	21.22	21.16	22.93	24.89	21.19	19.30	18.34	18.50	18.16	**18.01**
	RMSE	38.61	44.56	40.65	33.44	34.89	35.22	39.66	33.65	31.46	29.93	30.39	29.77	**29.53**
	MAPE%	17.24	19.20	18.23	14.17	13.83	16.56	17.29	13.90	12.70	12.81	12.43	12.24	**12.13**
PEMS07	MAE	50.22	32.49	29.98	25.22	25.33	24.01	26.39	24.26	21.42	19.68	19.94	19.26	**18.91**
	RMSE	75.63	50.22	45.94	38.61	39.34	37.87	41.50	39.03	34.51	33.19	33.88	32.51	**32.39**
	MAPE%	32.22	14.26	13.20	11.82	11.21	10.73	11.97	10.21	9.01	8.43	8.38	8.11	**7.88**
PEMS08	MAE	19.19	23.25	23.09	16.82	17.50	18.25	18.28	17.13	15.67	14.35	13.42	13.45	**13.21**
	RMSE	29.81	36.16	35.17	26.36	27.09	28.06	30.05	26.80	24.77	24.18	23.40	23.28	**22.88**
	MAPE%	13.10	14.64	14.99	10.92	11.29	11.64	12.15	10.96	9.94	9.33	8.90	8.77	**8.67**

Table 5: **Multi-step Forecasting Results**

5.2.2. Visualization Analysis of Results

To facilitate understanding of the performance, visualizations of the results are presented with detailed analysis.

Fig.3 displays the evolution of metrics with different prediction horizons on the PeMS04/08 datasets. A deterioration of all models is noted as the horizon extends from 1 to 12 steps. DWISTGNN consistently outperforms STWave and STIDGCN across all horizons and metrics, demonstrating its robustness and effectiveness.

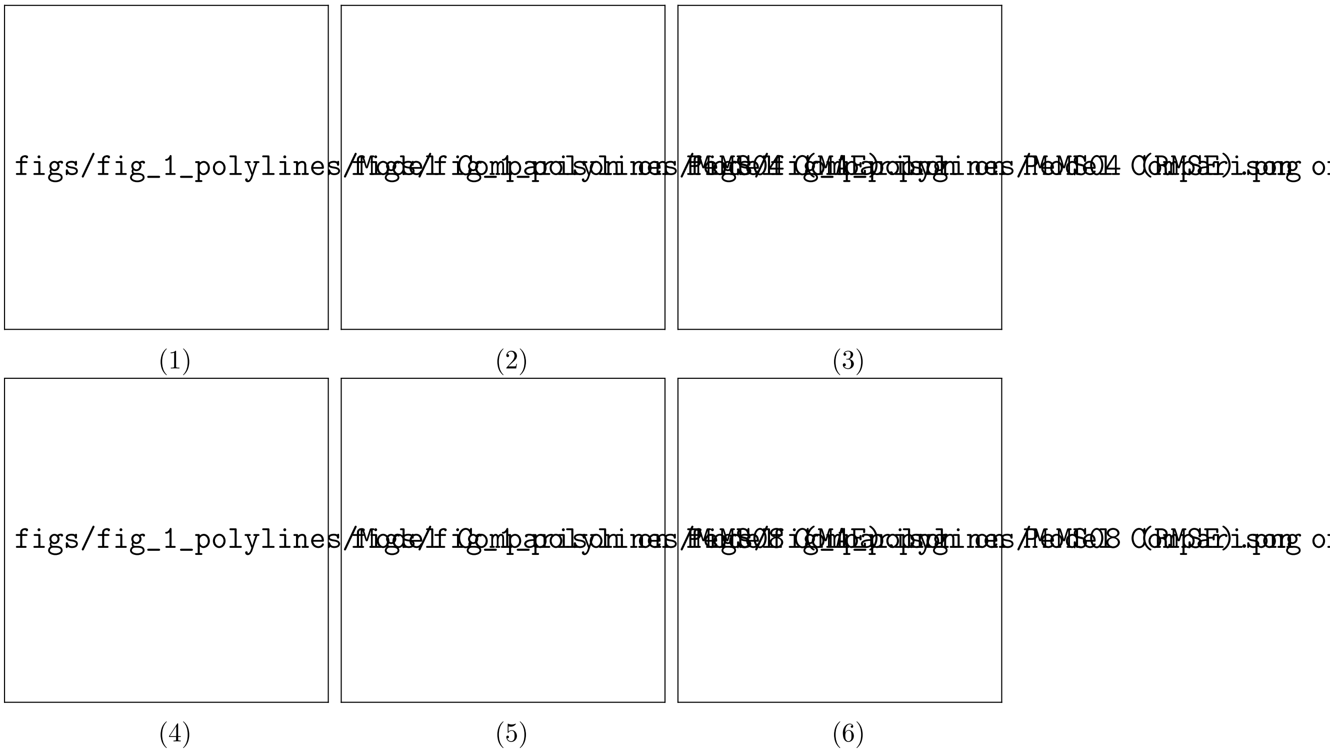
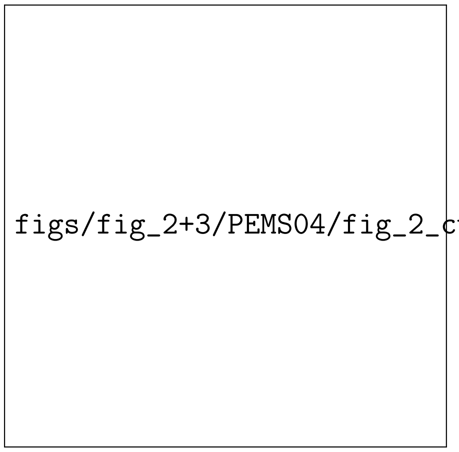
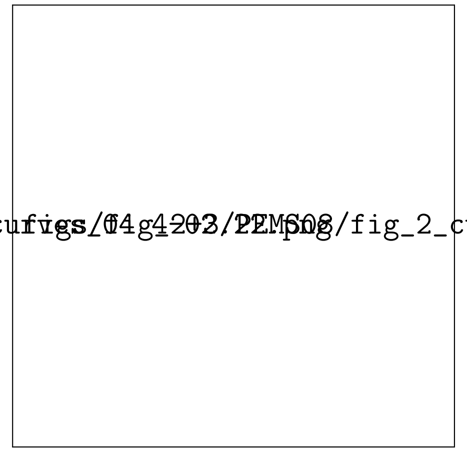


Figure 3: Horizon Metrics on PeMS04/08

Fig.4 illustrates the forecasting results during rush hours on the PeMS04/08 datasets. **Fig.4**(1) corresponds to the time period from 10:30 to 13:30 on 2018-02-22 for Node 4 in PeMS04, which describes the increasing traffic flow around noon. **Fig.4**(2) depicts the fluctuation during morning and evening rush hours on 2016-08-29 for Node 89 in PeMS08. In both cases, DWISTGNN consistently provides more accurate predictions than STWave and STIDGCN during peak periods, highlighting its capability to capture complex patterns.



(1)



(2)

Figure 4: Rush Hours Forecasting on PeMS04/08

Fig.5 unites the real values and predicted results and presents them in scatter plots. The linear fits represent the regression lines of the scatter plots. The closer the scatters lie to the diagonal line (Ground Truth), the higher the forecasting performance. Across all sub-figures **Fig.5**(1) to **Fig.5**(4), which correspond to specific days and the entire time range of the PeMS04/08 datasets respectively, the regression line of DWISTGNN is the closest to the diagonal line, demonstrating its superior predictive performance.

Fig.6 visualizes the dynamic process of graph evolution and reveals the essence of the DGC module in DWISTGNN. **Fig.6**(1) and (2) show the adjacency matrices at two different time points. **Fig.6**(3) to **Fig.6**(6) illustrate the increment of the adjacency matrices over different time windows, which correspond to the compensatory graph adjacency matrices in the DGC module. These visualizations demonstrate how the DGC module adapts to evolutions of complex dependencies.

5.3. Ablation Study

An ablation study is performed on the PeMS04/08 datasets to prove the necessity of the proposed modules in DWISTGNN. The results are presented in **Table 6**. Four variants of DWISTGNN are created by removing one key component at a time:

- **w/o DWT**: This variant excludes the DWT module, which is responsible for decomposing the input into different frequency bands. To ensure the training stability, the original input interfaces of all components are fed with the same raw series.
- **w/o DGC**: This variant replaces the graph structure constructed by the DGC module with the initial specific graph to evaluate the capability of capturing graph dynamicity.
- **w/o GCN**: This variant eliminates the entire Graph Convolutional Network part, which captures spatial dependencies among nodes to validate its contribution to model performance.
- **w/o FI**: This variant omits the FI module, which models the deep interactions between different frequency components. Instead, the outputs from the previous modules are directly passed to the subsequent modules.

Table 6: **Ablation Study**

Datasets	PEMS04			PEMS08		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE
w/o DWT	18.12	29.69	12.17	13.29	23.08	8.75
w/o DGC	18.39	29.92	12.35	13.43	23.32	8.77
w/o GCN	18.36	29.94	12.29	13.65	23.46	8.94
w/o FI	18.13	29.78	12.18	13.28	22.90	8.77
DWISTGNN	**18.01**	**29.53**	**12.13**	**13.21**	**22.88**	**8.67**

Fig.7 shows the histogram of metrics from the ablation study. On both datasets, full DWISTGNN model outperforms all its variants. The w/o DGC variant exhibits the greatest drop in accuracy, indicating the necessity of capturing dynamic spatial dependencies over static graphs. The w/o GCN variant also deteriorates heavily, proving the importance of modeling variates' spatial relations. The w/o FI variant and the w/o DWT variant show moderate performance decreases. Better performance of the w/o FI variant is observed when more frequency components are considered and interacted. Similarly, the w/o DWT variant performs better when more wavelet patterns

are extracted and utilized. Overall, this study validates that each module is integral to the performance of DWISTGNN.

6. Conclusion

In this work, we propose a Dynamic Spatio-Temporal Graph Neural Network with Wavelet Interaction for Multivariate Time Series Forecasting (DWISTGNN). We propose a Data Transform (DT) module that leverages Discrete Wavelet Transform (DWT) to obtain various frequency bands from raw data to capture multi-scale temporal patterns. A Dynamic Graph Construction (DGC) module is introduced to capture dynamic spatial dependencies by fusing an initial specific graph component, a gradual changing graph component and a compensatory graph component. Additionally, we introduce a Frequency Interaction (FI) module to model the deep interactions between decomposed frequency components. Abundant experiments on eight benchmark datasets showcase the advantage capability of DWISTGNN. Nevertheless, there still exist some constraints in this work. First, the DWT used in DT module may perform differently on datasets with different patterns. Second, the entire model remains non-lightweight, requiring sufficient computational resources when employed on large datasets and tending to overfit on small datasets. In future work, the priority is to explore more delicate architectures to further improve performance, reduce model complexity and adapt to more scenarios.

References

[1] Bin Wang, Junrui Shi, Binyu Tan, Minbo Ma, Feng Hong, Yanwei Yu, and Tianrui Li. Deepwind: a heterogeneous spatio-temporal model for wind forecasting. *Knowledge-Based Systems*, 286:111385, 2024.

[2] Zhewen Xu, Xiaohui Wei, Jieyun Hao, Junze Han, Hongliang Li, Changzheng Liu, Zijian Li, Dongyuan Tian, and Nong Zhang. Dgformer: a physics-guided station level weather forecasting model with dynamic spatial-temporal graph neural network. *GeoInformatica*, 28(3):499–533, 2024.

[3] Yujie Yang, Junhong Chen, Wenbin Zheng, and Fan Zhang. Long-term wind power forecasting with series decomposition and spatio-temporal graph neural network. *International Journal of Green Energy*, 21(15):3470–3484, 2024.

[4] Xiaojun Shen, Chao Ji, Xiaohan Lai, Lei Zhao, and Shouyi Jiang. Energy-stgnn: A dynamic forecasting approach for energy consumption prediction. In *2024 10th IEEE International Conference on Intelligent Data and Security (IDS)*, pages 75–79. IEEE, 2024.

[5] Wei Zhuang, Jili Fan, Min Xia, and Kedong Zhu. A multi-scale spatial–temporal graph neural network-based method of multienergy load forecasting in integrated energy system. *IEEE Transactions on Smart Grid*, 15(3):2652–2666, 2023.

[6] Mengda Xing, Weilong Ding, Tianpu Zhang, and Han Li. Stcgcn: a spatio-temporal complete graph convolutional network for remaining useful life prediction of power transformer. *International journal of web information systems*, 19(2):102–117, 2023.

[7] Xiangjie Kong, Wenfeng Zhou, Guojiang Shen, Wenyi Zhang, Nali Liu, and Yao Yang. Dynamic graph convolutional recurrent imputation network for spatiotemporal traffic missing data. *Knowledge-Based Systems*, 261:110188, 2023.

[8] Shuhao Li, Yue Cui, Libin Li, Weidong Yang, Fan Zhang, and Xiaofang Zhou. St-abc: Spatio-temporal attention-based convolutional network for multi-scale lane-level traffic prediction. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1185–1198. IEEE, 2024.

[9] Weiyang Kong, Ziyu Guo, and Yubao Liu. Spatio-temporal pivotal graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 8627–8635, 2024.

[10] George Box and GM Jenkins. Analysis: Forecasting and control. *San francisco*, 1976.

[11] Harris Drucker, Christopher J Burges, Linda Kaufman, Alex Smola, and Vladimir Vapnik. Support vector regression machines. *Advances in neural information processing systems*, 9, 1996.

[12] Eric Zivot and Jiahui Wang. Vector autoregressive models for multivariate time series. In *Modeling financial time series with S-PLUS®*, pages 369–413. Springer, 2006.

[13] Sheng Chen, XX Wang, and Chris J Harris. Narx-based nonlinear system identification using orthogonal least squares basis hunting. *IEEE Transactions on Control Systems Technology*, 16(1):78–84, 2007.

[14] Roger Frigola-Alcalde. *Bayesian time series learning with Gaussian processes*. University of Cambridge (United Kingdom), 2015.

[15] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[16] Nonparametric Regression. An introduction to kernel and nearest-neighbor. *The American Statistician*, 46(3):175–185, 1992.

[17] G Peter Zhang. Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50:159–175, 2003.

[18] Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pages 37–45, 2012.

[19] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[20] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

[21] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 95–104, 2018.

[22] Shun-Yao Shih, Fan-Keng Sun, and Hung-yi Lee. Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 108(8):1421–1441, 2019.

[23] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[24] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[25] TN Kipf. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[26] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE transactions on knowledge and data engineering*, 36(10):5388–5408, 2023.

[27] Flavio Corradini, Flavio Gerosa, Marco Gori, Carlo Lucheroni, Marco Piangerelli, and Martina Zannotti. A systematic literature review of spatio-temporal graph neural network models for time series forecasting and classification. *arXiv preprint arXiv:2410.22377*, 2024.

[28] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

[29] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*, 2019.

[30] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 753–763, 2020.

[31] Xuxiang Ta, Zihan Liu, Xiao Hu, Le Yu, Leilei Sun, and Bowen Du. Adaptive spatio-temporal graph neural network for traffic forecasting. *Knowledge-based systems*, 242:108199, 2022.

[32] Shengnan Guo, Youfang Lin, Huaiyu Wan, Xiucheng Li, and Gao Cong. Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 34(11):5415–5428, 2021.

[33] Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting. In *International conference on machine learning*, pages 11906–11917. PMLR, 2022.

[34] Junchen Ye, Zihan Liu, Bowen Du, Leilei Sun, Weimiao Li, Yanjie Fu, and Hui Xiong. Learning the evolutionary and multi-scale graph structure for multivariate time series forecasting. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pages 2296–2306, 2022.

[35] Kaiwen Xia, Li Lin, Shuai Wang, Qi Zhang, Shuai Wang, and Tian He. Prost: Prompt future snapshot on dynamic graphs for spatio-temporal prediction. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 1645–1656, 2025.

[36] Minhao Liu, Ailing Zeng, Muxi Chen, Zhijian Xu, Qiuxia Lai, Lingna Ma, and Qiang Xu. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022.

[37] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in neural information processing systems*, 34:22419–22430, 2021.

[38] Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and Jun Zhou. Timemixer: Decomposable multiscale mixing for time series forecasting. *arXiv preprint arXiv:2405.14616*, 2024.

[39] Qinyao Luo, Silu He, Xing Han, Yuhan Wang, and Haifeng Li. Lsttn: A long-short term transformer-based spatiotemporal neural network for traffic flow forecasting. *Knowledge-Based Systems*, 293:111637, 2024.

[40] Hyunwoo Seo and Chiehyeon Lim. St-mtm: Masked time series modeling with seasonal-trend decomposition for time series forecasting. *arXiv preprint arXiv:2507.00013*, 2025.

[41] Defu Cao, Yujing Wang, Juanyong Duan, Ce Zhang, Xia Zhu, Congrui Huang, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, et al. Spectral temporal graph neural network for multivariate time-series forecasting. *Advances in neural information processing systems*, 33:17766–17778, 2020.

[42] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, pages 27268–27286. PMLR, 2022.

[43] Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain mlps are more effective learners in time series forecasting. *Advances in Neural Information Processing Systems*, 36:76656–76679, 2023.

[44] Zhijian Xu, Ailing Zeng, and Qiang Xu. Fits: Modeling time series with $10k$ parameters. *arXiv preprint arXiv:2307.03756*, 2023.

[45] Tian Tian, Chunyan Miao, and Hangwei Qian. Frera: A frequency-refined augmentation for contrastive learning on time series classification. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 2835–2846, 2025.

[46] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE transactions on intelligent transportation systems*, 21(9):3848–3858, 2019.

[47] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.

[48] Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S Jensen. Decoupled dynamic spatial-temporal graph neural network for traffic forecasting. *arXiv preprint arXiv:2206.09112*, 2022.

[49] Yuchen Fang, Yanjun Qin, Haiyong Luo, Fang Zhao, Bingbing Xu, Liang Zeng, and Chenxing Wang. When spatio-temporal meet wavelets: Dis-

entangled traffic forecasting via efficient spectral graph attention networks. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 517–529. IEEE, 2023.

[50] Aoyu Liu and Yaying Zhang. Spatial–temporal dynamic graph convolutional network with interactive learning for traffic forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 25(7):7645–7660, 2024.

[51] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[52] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11106–11115, 2021.

[53] Chao Shang, Jie Chen, and Jinbo Bi. Discrete graph structure learning for forecasting multiple time series. *arXiv preprint arXiv:2101.06861*, 2021.

[54] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 922–929, 2019.

[55] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan. Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 914–921, 2020.

figs/fig_2+3/PEMS04/fig_3_regression_04.png

(1)

figs/fig_2+3/PEMS04/fig_3_regression_04 (3375).png

(2)

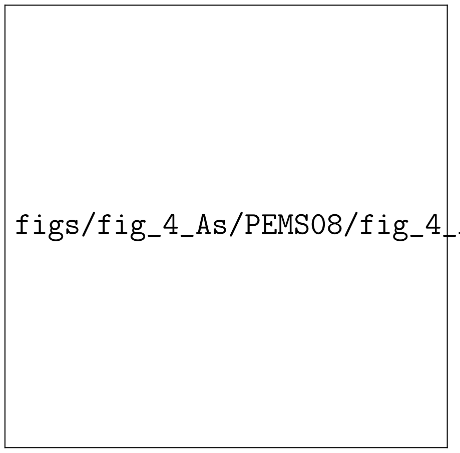
figs/fig_2+3/PEMS08/fig_3_regression_08.png

(3)

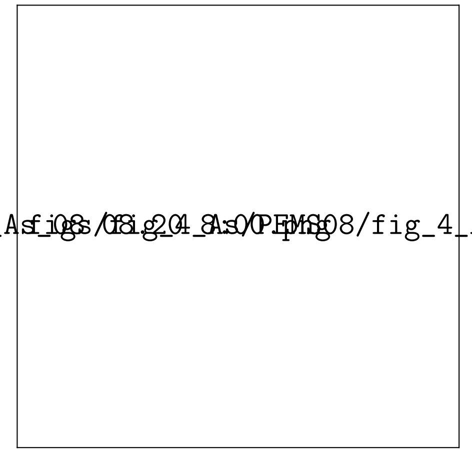
figs/fig_2+3/PEMS08/fig_3_regression_08 (3548).png

(4)

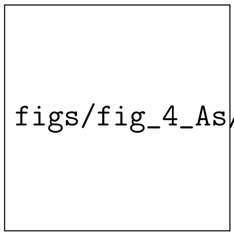
Figure 5: Scatter Plots with Linear Fits on PeMS04/08



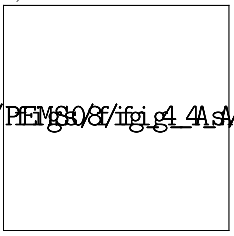
(1)



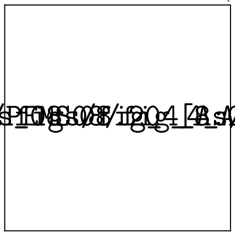
(2)



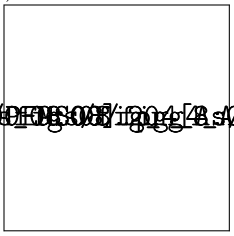
(3)



(4)



(5)



(6)

Figure 6: Compensatory Graph Adjacency Matrix on PeMS08

figs/fig_5_ablation/Ablation-PEMS04-MAE.png figs/fig_5_ablation/Ablation-PEMS04-RMSE.png figs/fig_5_ablation/Ablation-PEMS04-MAP

(1) (2) (3)

figs/fig_5_ablation/Ablation-PEMS08-MAE.png figs/fig_5_ablation/Ablation-PEMS08-RMSE.png figs/fig_5_ablation/Ablation-PEMS08-MAP

(4) (5) (6)

Figure 7: Histogram of Ablation Study