Progressive Decomposition-enhanced Time-Varying Graph Neural Network for Traffic Forecasting

Jianuo Ji[1], Hongbin Dong[1](✉), and Xiaoping Zhang[2]

[1] College of Computer Science and Technology,
Harbin Engineering University, Harbin, China
{jijianuo, donghongbin}@hrbeu.edu.cn
[2] Traditional Chinese Medicine Data Center,
China Academy of Chinese Medical Sciences, Beijing, China
zhangxp@ndctcm.cn

Abstract. Traffic forecasting, a core technology in intelligent transportation systems, has a broad range of applications. The fundamental challenge in traffic prediction lies in effectively modeling the complex spatio-temporal dependencies inherent in traffic data. Spatio-temporal graph neural network (GNN) models have emerged as one of the most promising approaches to address this challenge. However, GNN-based models for traffic forecasting have two significant limitations: i) Most methods model spatial dependencies in a static manner (predefined or self-learning), which fails to capture the time-varying nature of spatial dependencies in real-world scenarios; ii) It is unreliable to capture temporal and spatial dependencies in entangled temporal patterns. To this end, we propose a **P**rogressive **D**ecomposition-enhanced **T**ime-**V**arying **G**raph **N**eural **N**etwork, namely PDTVGNN, for accurate traffic forecasting. Specifically, we design a time-varying graph generator that incrementally generates a series of adjacency matrices to capture the time-varying spatial relationships. Moreover, we adopt a novel progressive decomposition idea where the decomposition blocks are embedded as internal blocks to decouple the entangled temporal patterns gradually. The decoupled trend and seasonal parts are modeled via the proposed spatio-temporal normalization module and attention mechanism, respectively. Extensive experimental results on four real-world public traffic datasets demonstrate that the proposed method outperforms state-of-the-art baselines.

Keywords: Graph neural network · Traffic forecasting · Seasonal-trend Decomposition

1 Introduction

Traffic management systems are facing growing pressure as vehicles on road networks increase. There is a pressing need to develop Intelligent Transportation Systems (ITS) to achieve efficient traffic control. As a core technology and essential prerequisite for ITS implementation, traffic forecasting is pivotal in enabling effective traffic management [1].

The primary challenge in traffic forecasting is effectively capturing and modeling the complex and dynamic spatio-temporal dependencies in traffic data [2]. Over the years, researchers have explored various approaches, with deep learning techniques increasingly being adopted to uncover these spatio-temporal correlations. Graph neural networks (GNN) have gained popularity for spatial correlation modeling due to their strong capability in handling graph-structured data [3,4,5].

However, traffic forecasting has its characteristics in modeling temporal correlations within time series and capturing spatial dependencies between time series. Despite the effectiveness of existing methods, GNN-based models still have two significant limitations in traffic prediction. On the one hand, in real-world scenarios, the spatial dependencies between traffic sensors are highly dynamic, and the change of spatial relationships in a one-time step is closely related to the spatial relationships in the time step before it, which we call time-varying. For example, as shown in Fig. 1(b), the correlation between nodes A and B decreases in the morning and becomes more potent at other times. Existing methods mainly model spatial dependencies in a static way (predefined or self-learning), which obviously cannot handle such dynamic changes. Therefore, existing work does not fully harness the potential of graph neural networks on this problem. On the other hand, most existing methods capture potential dependencies directly from traffic sequences, which fails to uncover the intricate spatio-temporal dependencies masked under entangled temporal patterns. Specifically, they ignore that traffic data are composed of complex periodic patterns coupled with trend components in real-world scenarios, as shown in Fig. 1(c) and (d). This entanglement limits the capability of existing methods and is a performance bottleneck for traffic forecasting.

To address the above limitations, we propose a **P**rogressive **D**ecomposition-enhanced **T**ime-**V**arying **G**raph **N**eural **N**etwork, namely PDTVGNN, for traffic forecasting. Unlike previous methods, this paper introduces a novel progressive decomposition idea to disentangle complex temporal patterns by gradually decomposing the hidden sequences using the time series decomposition module throughout the forecasting process. For the seasonal part obtained after decoupling, we use the Fourier temporal attention to capture the temporal correlations. Additionally, we propose a time-varying graph generation module to construct a series of adjacency matrices that model the time-varying nature of the spatial structure, rather than maintaining a static graph structure. Subsequently, we employ a time-varying graph convolution module to extract the spatial dependencies. For the trend part obtained after decoupling, we propose a spatio-temporal normalization module to address the temporal and spatial non-stationarity information in the trend data. In summary, the main contributions are as follows:

– To cope with the intricate spatio-temporal dependencies masked by the entangled temporal patterns, we embed the decomposition blocks as internal blocks to progressively decouple the entangled temporal patterns. Moreover, the trend and seasonal parts obtained after decoupling are modeled separately.

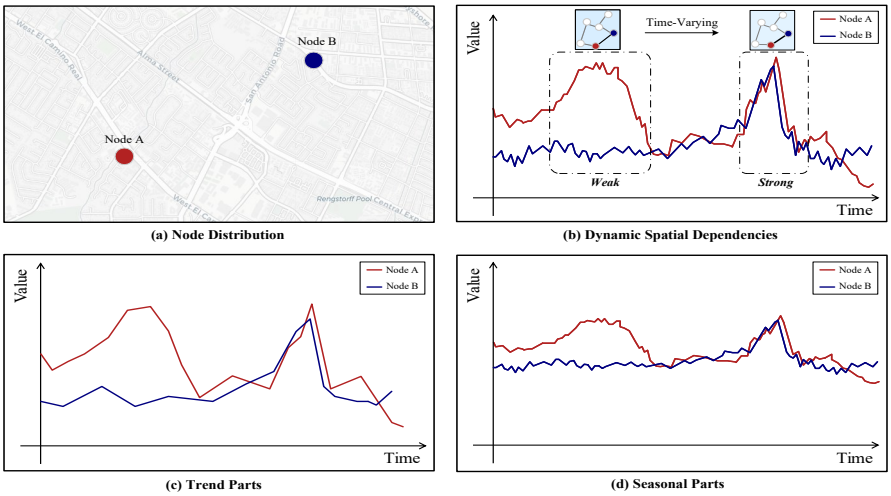


Fig. 1. The findings about traffic prediction.

– We propose a time-varying graph generator that produces a series of recurrent adjacency matrices to capture the dynamic and time-varying nature of spatial relationships.

– We validate the effectiveness of our method through extensive experiments on four real-world datasets, demonstrating its superior performance and significantly outpacing competitive baseline methods.

2 Related Work

2.1 Traffic Forecasting

Traffic forecasting has been extensively studied in intelligent transportation systems [6]. Early work focused on statistical methods like ARIMA [7] to predict traffic indicators. Later, machine learning models, such as VAR [8], were introduced with some success but were limited by assumptions of static conditions, hindering their ability to capture the complex nonlinear relationships in traffic data. Furthermore, these models neglected spatial dependencies, restricting their accuracy. Unlike earlier methods, deep neural networks, which capture both temporal and spatial features, have become popular. For example, FC-LSTM [9] combines CNNs with LSTM for traffic prediction. However, these models remain less effective in graph-based node data scenarios.

2.2 Spatio-Temporal Graph Neural Networks

In recent years, Graph Neural Networks have gained increasing attention due to their advanced performance. Therefore, researchers have begun to incorporate them into traffic prediction models to enhance predictive accuracy. Models

such as DCRNN [10] and STGCN [11] are among the most prominent works in this area. These models capture spatial dependencies between nodes through predefined graph structures and temporal dynamics through CNNs or RNNs. However, these approaches heavily rely on manually defined graph structures, with the quality of the predefined graph structure directly influencing model performance. To address this limitation, frameworks such as Graph WaveNet [12], MTGNN [3], and AGCRN [13] have been proposed. These frameworks generate graph structures adaptively in a data-driven manner and have demonstrated remarkable results as a result. In recent work, PDFormer [14] utilizes three attentions to form spatial-temporal feature extraction blocks, which can model local geographic and global semantic information from neighbors, thus improving prediction accuracy. However, maximizing its performance depends to some extent on these three features. STPGNN [5] defines pivotal nodes based on the aggregation and distribution capabilities of traffic nodes and proposes a pivotal graph convolutional network to predict traffic. However, the performance of the model depends on how accurately the pivotal nodes are identified. In addition, DGCRN [15] designs hypernetworks that utilize and extract dynamic features of node attributes and dynamic filters to generate dynamic graphs. In general, traffic data contains strong dynamic spatio-temporal correlations. Therefore, modeling dynamic nonlinear spatio-temporal correlation is crucial for accurately predicting traffic flow. The dynamic generation of dynamic graphs has become a new research direction.

3 Preliminaries

In this section, we first present the task definition and then briefly review the core idea of attention mechanism.

3.1 Definition and Problem Statement

Traffic Topology Graph. A traffic topology graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ within certain road network. Where $\mathcal{V}$ denotes the set of nodes($|\mathcal{V}| = N$) and each node corresponds to a road sensor; $\mathcal{E}$ denotes the set of edges, which represents the physical connectivity of the sensors; and $\mathcal{A}$ denotes the adjacency matrix of the graph, whose elements are the connectivity between any pair of nodes in the graph.

Traffic Signal Tensor. The traffic signal tensor of N nodes over P time steps is denoted as $\mathcal{X} = (\mathrm{X}_1, ..., \mathrm{X}_t, ..., \mathrm{X}_P) \in \mathbb{R}^{P \times N \times C}$. Here, $\mathrm{X}_t \in \mathbb{R}^{N \times C}$ denotes the traffic signal of N nodes in the road network at time step t, and C is the number of traffic features (e.g., $\mathrm{C} = 3$ for features traffic flow, speed, and occupy.). The traffic forecasting problem can be formalized as Eq. (1). Given a series of observations from N sensors in the graph $\mathcal{G}$ over the past P time steps, our goal is to predict the traffic signals at the next Q time steps via a mapping function $\mathcal{F}$:

$$[\mathrm{X}_{P+1}, ..., \mathrm{X}_{P+Q}] = \mathcal{F}([\mathrm{X}_1, ..., \mathrm{X}_P; \mathcal{G}]). \tag{1}$$

3.2 Attention Mechanism

The attention mechanism is a fundamental operation frequently employed in various modeling tasks. Its core principle involves assigning distinct weights to different segments of the input data, thereby emphasizing relevant information while disregarding less important details [16]. In essence, the attention mechanism operates by mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are represented as vectors. The output is computed as a weighted sum of the values, with each weight being determined by the interaction between the corresponding key and the query. These weights reflect the degree of association between the query and each key-value pair. In this work, we employ Scaled Dot-Product Attention [17], a widely used variant of the attention mechanism, specifically:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V. \tag{2}$$

where Q, K, V , and d_k are the query, key, value, and their dimensions, respectively.

4 Methodology

4.1 Model Overview

The framework of our PDTVGNN is illustrated in Fig. 2. This section provides a detailed discussion of its technical components. It consists of stacked L spatio-temporal layers and an output layer. For each layer, this paper adopts a novel progressive decomposition idea, where the series decomposition module gradually disentangles hidden series throughout the forecasting process. This enables the model to separate complex temporal patterns and generate predictions based on the more predictable seasonal and trend components. For the seasonal part, we extract temporal correlations using Fourier temporal attention, as shown in Fig. 2(b). Furthermore, we utilize the time-varying graph generator, as shown in Fig. 2(c), along with the time-varying graph convolution module to capture spatial correlations over time. For the trend component, we develop a spatio-temporal normalization module to address the temporal and spatial non-stationarity in the trend component. The extracted trend information is progressively accumulated and serves as the final trend representation learned by the model. Each spatio-temporal layer is skip-connected to the output layer. The computation process for layer $l \in \{1, ..., L\}$ can be formalized as follows:

$$\begin{aligned}
\mathcal{S}_1^l, \mathcal{T}_1^l &= \text{SeriesDecomp}(\mathcal{X}^{l-1}) \\
\mathcal{S}_2^l, \mathcal{T}_2^l &= \text{SeriesDecomp}(\text{FFT-Attention}(\mathcal{S}_1^l)) \\
\mathcal{S}_3^l, \mathcal{T}_3^l &= \text{SeriesDecomp}(\text{Time-VaryingGCN}(\mathcal{S}_2^l)) \\
\mathcal{S}_3^l &= \text{FeedForward}(\mathcal{S}_3^l) + \mathcal{S}_3^l \\
\mathcal{X}^l &= \mathcal{S}_3^l + \text{STNorm}(\mathcal{T}_1^l) + \text{STNorm}(\mathcal{T}_2^l) + \text{STNorm}(\mathcal{T}_3^l)
\end{aligned} \tag{3}$$

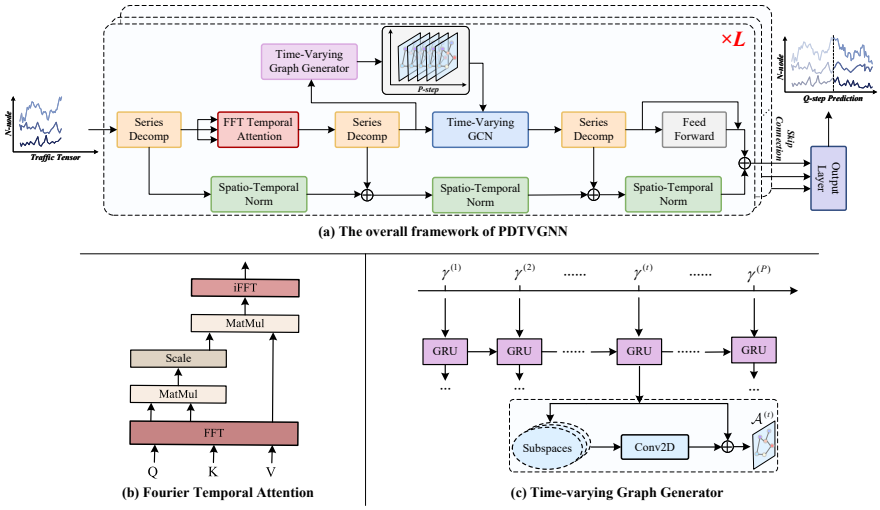


Fig. 2. Detailed framework of PDTVGNN.

where $\mathcal{X}^l$ denotes the output of layer l. $\mathcal{S}_i^l$ and $\mathcal{T}_i^l$ denote the seasonal and trend components obtained from the i-th time series decomposition module of layer l. The first layer input data $\mathcal{X}^0 \in \mathbb{R}^{P\times N\times D}$ is the high-dimensional spatial projection of the historical traffic observation data $\mathcal{X} \in \mathbb{R}^{P\times N\times C}$ through the linear layer, and D is the dimension of the hidden state.

4.2 Series Decomposition Module

Series decomposition [18], has been applied to time-series forecasting models such as the Autoformer [19] and FEDformer [20] to capture complex temporal patterns. To improve accuracy, we decompose the traffic series into trend and seasonal parts, which represent the long-term trend and potential periodicity of the traffic series, respectively. Specifically, moving averages is used to smooth out cyclical fluctuations, and the seasonal part is obtained by subtracting the trend from the original time series. As in Eq. (4), for input data $\mathcal{X} \in \mathbb{R}^{P\times N\times C}$ of length P, the trend and seasonal parts $\mathcal{X}_{sea}$, $\mathcal{X}_{tre} \in \mathbb{R}^{P\times N\times C}$ are obtained, respectively.

$$\begin{aligned} \mathcal{X}_{tre} &= AvgPool(padding(\mathcal{X}), w), \\ \mathcal{X}_{sea} &= \mathcal{X} - \mathcal{X}_{tre}. \end{aligned} \tag{4}$$

$AvgPool(\cdot)$ is a moving average operation with window size w and uses the operation *padding* to keep the length of the series constant.

4.3 Temporal Correlation Extraction

The series decomposition module decomposes the input of the P time steps into trend parts and seasonal parts. For the decomposed results, the seasonal part

and the trend part are modeled using Fourier temporal attention and spatio-temporal normalization module.

Fourier Temporal Attention for Seasonal Parts. The seasonal component of the time series corresponds to high-frequency values. The Softmax operation amplifies larger values and diminishes smaller ones, concentrating attention on dominant frequencies and improving seasonal information capture [21]. In contrast to time-domain attention, frequency-domain attention offers superior performance, as the primary frequency patterns are directly accessible in the frequency domain. Consequently, as illustrated in Eq. (5) and (6), Fourier attention is first computed in the frequency domain by transforming the query, key, and value through Fourier transformation. The result is then mapped back to the time domain through an inverse Fourier transformation.

$$\begin{aligned} Q_f &= FFT(Q) = FFT(\mathcal{X}_{sea}W_Q), \\ K_f &= FFT(K) = FFT(\mathcal{X}_{sea}W_K), \\ V_f &= FFT(V) = FFT(\mathcal{X}_{sea}W_V), \end{aligned} \tag{5}$$

$$\hat{\mathcal{X}}_{sea} = iFFT(softmax(Q_f K_f^T)V_f). \tag{6}$$

where W_Q, W_K, W_V are learnable parameters. With the help of Fast Fourier Transform (FFT), the computational complexity can be reduced from $O(n^2)$ to $O(n \log n)$.

Spatio-Temporal Normalization for Trend Parts. The attention mechanism fundamentally operates by modifying and adding to the contextual history, resulting in the inevitable loss of temporal information and poor generalization of trend data [21]. This limitation motivates the decomposition of time series into trend and seasonal parts. To address real-world scenarios where traffic data series exhibit non-stationarity and dynamic changes in data distribution, we propose a spatio-temporal normalization method (STNorm), which includes both temporal and spatial normalization. This method is designed to handle the temporal and spatial non-stationarity of the trend part, respectively.

Specifically, for the given trend input $\mathcal{X}_{tre} = \{\{\mathrm{X}_{tre(t,v)}\}_{t=1}^{P}\}_{v=1}^{N}$, the mean and standard deviation of the particular instance along the time axis are computed as follows:

$$\begin{aligned} \mu_T[\mathrm{X}_{tre(t,v)}] &= \frac{1}{P}\sum_{i=1}^{P}\mathrm{X}_{tre(i,v)}, \\ \sigma_T^2[\mathrm{X}_{tre(t,v)}] &= \frac{1}{P}\sum_{i=1}^{P}(\mathrm{X}_{tre(i,v)} - \mu_T[\mathrm{X}_{tre(t,v)}])^2, \end{aligned} \tag{7}$$

We then normalize the trend input across the time dimension as follows:

$$\mathrm{X}_{tre(t,v)}^{T} = \alpha_T \left(\frac{\mathrm{X}_{tre(t,v)} - \mu_T[\mathrm{X}_{tre(t,v)}]}{\sqrt{\sigma_T^2[\mathrm{X}_{tre(t,v)}] + \varepsilon}} \right) + \beta_T, \tag{8}$$

where α_T, β_T are learnable parameter vectors and ε is a small constant that maintains numerical stability. Similarly, we compute the mean and standard deviation of a particular instance along the spatial axis and then perform spatial normalization:

$$\begin{aligned}\mu_S[\mathrm{X}_{tre(t,v)}] &= \frac{1}{N}\sum_{j=1}^{N}\mathrm{X}_{tre(t,j)},\\ \sigma_S^2[\mathrm{X}_{tre(t,v)}] &= \frac{1}{N}\sum_{j=1}^{N}(\mathrm{X}_{tre(t,j)}-\mu_S[\mathrm{X}_{tre(t,v)}])^2,\end{aligned} \tag{9}$$

$$\mathrm{X}^S_{tre(t,v)} = \alpha_S\left(\frac{\mathrm{X}_{tre(t,v)}-\mu_S[\mathrm{X}_{tre(t,v)}]}{\sqrt{\sigma_S^2[\mathrm{X}_{tre(t,v)}]+\varepsilon}}\right)+\beta_S, \tag{10}$$

With spatio-temporal normalization, the model extracts non-stationary information while preserving a consistent distribution for forecasting. The two normalized inputs are then combined and passed through a two-layer MLP to predict future trends:

$$\hat{\mathcal{X}}_{tre} = MLP([\mathcal{X}^T_{tre}, \mathcal{X}^S_{tre}]). \tag{11}$$

4.4 Time-Varying Graph Structure Generator

In real-world scenarios, the spatial correlations between nodes are not always constant, and the graph structure changes smoothly over time. To account for this inherent property, we propose a time-varying graph structure generator (TVGG) to capture the dynamic correlations between nodes. This module takes into account both the dependencies on the current input values and the graph structure from the previous time step, which is modeled recurrently:

$$\mathcal{A}^{(t)} = \mathcal{F}^d(\mathcal{A}^{(t-1)}, \gamma^{(t)}). \tag{12}$$

where $\mathcal{A}^{(t)} \in \mathbb{R}^{N\times N}$ denotes the adjacency matrix representing the spatial correlations at time step t, $\gamma^{(t)}$ refers to the node features, and $\mathcal{F}^d$ is the function for extracting change correlations. For simplicity, we omit the subscript l, which denotes the layer number, both in this equation and throughout the rest of this subsection.

However, directly parameterizing the adjacency matrix $\mathcal{A}$ and the mapping function $\mathcal{F}^d$ introduces significant computational overhead. To address this issue, we assume that the nodes have a time-varying representation e over time, and the time-varying graph structure can be derived from this dynamic node representation.

We use the Gated Recurrent Unit (GRU), a simple but powerful variant of recurrent neural networks, to model the dynamics of time-varying representations. Define $e^{(t)} \in \mathbb{R}^{N\times D_e}$ as the hidden state and the update process of the

GRU as:

$$\begin{aligned} z^{(t)} &= \sigma(W_z[\gamma^{(t)}, e^{(t-1)}] + b_z), \\ r^{(t)} &= \sigma(W_r[\gamma^{(t)}, e^{(t-1)}] + b_r), \\ \tilde{e}^{(t)} &= tanh(W_e[\gamma^{(t)}, (r^{(t)} \odot e^{(t-1)})] + b_e), \\ e^{(t)} &= z^{(t)} \odot e^{(t-1)} + (1 - z^{(t)}) \odot \tilde{e}^{(t)}, \end{aligned} \tag{13}$$

where $r^{(t)}$ and $z^{(t)}$ denote the reset gate and update gate, respectively, $\odot$ represents the Hadamard product, W_z, W_r and W_e are the learnable parameters, and $\sigma$is the sigmoid function.

Inspired by [22], we adopt a spatial embedding method to extract node features as the initial state of the GRU. First, an additional embedding vector is assigned to each node, and then a graph convolution layer is applied for Laplace smoothing. This process enhances the representation of each node by incorporating information from its neighbors, explicitly modeling the spatial structure while reflecting the graph structure information. The resulting spatial representation, $e \in \mathbb{R}^{N \times D_e}$, encapsulates the rich features of the nodes, which are then combined with a fully connected layer to form the initial hidden state $e^{(0)}$.

$$\begin{aligned} \hat{\mathcal{A}}_{ij,v}^{(t)} &= \frac{(W_v e_i^{(t)} \cdot W_v e_j^{(t)})}{\delta}, \\ \mathcal{A}_{ij,v}^{(t)} &= \text{conv2D}(\hat{\mathcal{A}}_{ij,v}^{(t)}) + \hat{\mathcal{A}}_{ij,v}^{(t)}, \end{aligned} \tag{14}$$

where $\mathcal{A}_{ij,v}^{(t)}$ denotes the value of the graph structure learned by the v-th subspace at time step t in the i-th row and j-th column. W_v is the learnable matrix that maps the spatial correlation information to the v-th subspace. The operation $\cdot$ represents the inner product of vectors, and δ is used to prevent the correlation values from deviating excessively. In the above formulation, we introduce a 2D convolutional layer along with residual connections to enhance the information exchange between different subspaces, where the number of channels in the convolutional kernel corresponds to the number of subspaces. Finally, the matrices of all subspaces are summed:

$$\mathcal{A}_{ij}^{(t)} = f(\sum_{v=1}^{V} \mathcal{A}_{ij,v}^{(t)}). \tag{15}$$

where $f(\cdot)$ represents a normalization process designed to prevent training instability, it yields the time-varying graph structure $\mathcal{A}^{(t)}$ at time step t.

4.5 Time-Varying Graph Convolution Module

Formally, the output γ_l of the l-th layer of temporal attention is passed into the l-th TVGG, as presented in Section 4.4, generating a series of adjacency matrices as follows:

$$[\mathcal{A}_l^{(1)}, \mathcal{A}_l^{(2)}, ..., \mathcal{A}_l^{(P)}] = \mathcal{F}_l^a(\gamma_l). \tag{16}$$

where $\mathcal{A}_l = [\mathcal{A}_l^{(1)}, \mathcal{A}_l^{(2)}, ..., \mathcal{A}_l^{(P)}]$, P is the historical time step.

After learning the adjacency matrices, spatial dependencies are extracted using a graph convolution-based spatial propagation method. First, initial residual connections [23] are employed in graph convolution. By emphasizing the initial features, the learned features become more discriminative after multiple graph convolution layers. The layer-wise propagation rule can be expressed as:

$$H^{(m)} = (\partial H^{(0)} + (1 - \partial)\mathcal{A}H^{(m-1)})W^{(m-1)}, \tag{17}$$

where $H^{(0)} = \gamma_l$ denotes the initial node features and ∂ is a hyperparameter that controls the proportion of initial information. Then, the component achieves the changing locality property by adaptively aggregating information from neighbors at different hops [24]. The output of the graph convolution is the combination of the representations obtained from all previous layers:

$$H^{(M)} = \sum_{m=0}^{M-1} H^{(m)}. \tag{18}$$

5 Experiments

This section presents a comprehensive evaluation of our proposed PDTVGNN framework. Specifically, we aim to address the following research questions:

- **RQ1.** How does PDTVGNN perform in the traffic prediction task?
- **RQ2.** How does each component of PDTVGNN contribute to the prediction?
- **RQ3.** How do key hyperparameters influence model performance?
- **RQ4.** Does PDTVGNN provide interpretability in the spatial dimension?

5.1 Setting

Datasets & Processing. To evaluate the performance of the PDTVGNN method, we conducted comparative experiments on four real-world datasets. Table 1 provides detailed information on the selected datasets, including the Los Angeles PEMS series (PEMS-BAY, PEMS04, PEMS08) and the California Metro Traffic Los Angeles (METR-LA). These datasets were chosen due to their distinct characteristics. Given the differences in sampling periods and regions, each dataset has unique time spans and node sizes, making them representative in their own right. Traffic flow data is aggregated at 5-minute intervals, resulting in 12 sample points per hour. The datasets are divided into training, validation, and test sets. In line with the most contemporary solution, METR-LA and PEMS-BAY are split in a 7:1:2 ratio, while PEMS04 and PEMS08 are split in a 6:2:2 ratio. Additionally, we utilize data from the previous hour (12 time steps) to predict traffic flow for the following hour (12 time steps), thereby performing multi-step prediction.

Table 1. Datasets description.

Datasets	Time steps	Nodes	Time windows	Data Type
METR-LA	34272	207	5 min	speed
PEMS-BAY	52116	325	5 min	speed
PEMS04	16992	307	5 min	flow
PEMS08	17856	170	5 min	flow

Baselines. We selected 13 baselines and categorized them into 4 classes: **(1)** Methods that do not consider spatial correlation: VAR [8], SVR, FC-LSTM [9]. **(2)** Methods based on predefined graphs: DCRNN [10], STGCN [11], STSGCN [4]. **(3)** Methods considering dynamic spatial correlation : GWnet [12], AGCRN [13], MTGNN [3], DGCRN [15]. **(4)** Other superior methods based on spatio-temporal graphs: GMAN [25], PDFormer [14], STPGNN [5].

Evaluative Metrics. Three commonly used evaluation metrics are employed: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE). All experiments are repeated five times, and the results were averaged.

Implementation Details. The implementation environment for PDTVGNN is Python 3.8 and PyTorch 2.2.1. The evaluation environment is a server equipped with a V100-PCIe-32. To train our model, an Adam optimization is used with an initial learning rate of 0.001, a batch size of 16, and a maximum of 60 epochs. Hyperparameters and their optimal values are determined on the validation set: the model dimension is 64, the number of attention heads is 8, and the number of model layers is [3, 3, 5, 5], respectively.

5.2 Predictive Performance (RQ1)

Table 2 and 3 show the results, where the bolded values are optimal and the underlined values are suboptimal.

We find the following observations. **(1)** PDTVGNN outperforms all other state-of-the-art baselines on all datasets. We note that PDTVGNN's MAPE results on PEMS04 are slightly lower, but all other metrics are more favorable, such as the 0.38% decrease under MAE metrics. **(2)** The traditional statistical model performs poorly because it only considers temporal correlation and ignores spatial dependence. **(3)** In GNN-based models, those that account for dynamic spatial correlations in time series, such as MTGNN and DGCRN, demonstrate more competitive performance than static graph embeddings. However, these models still rely on static graph structures and do not consider the spatial structure as time-varying and dynamic. In contrast, our PDTVGNN model incorporates the time-varying graph convolution module, which effectively captures the dynamic correlations among nodes by generating a series of time-varying graph structures,

Table 2. Performance comparison of different methods on PEMS-BAY and METR-LA.

Datasets	Methods	15 min			30 min			60 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PEMS-BAY	VAR	1.74	3.16	3.60%	2.32	4.25	5.00%	2.93	5.44	6.50%
	SVR	1.85	2.59	3.80%	2.48	5.18	5.50%	3.28	7.08	8.00%
	FC-LSTM	2.05	4.19	4.80%	2.20	4.55	5.20%	2.37	4.96	5.70%
	DCRNN	1.38	2.95	2.90%	1.74	3.97	3.90%	2.07	4.74	4.90%
	STGCN	1.36	2.96	2.90%	1.81	4.27	4.17%	2.49	5.69	5.79%
	STSGCN	1.44	3.01	3.04%	1.83	4.18	4.17%	2.26	5.21	5.40%
	GWNet	1.30	2.74	2.73%	1.63	3.70	3.67%	1.95	4.52	4.63%
	AGCRN	1.37	2.87	2.94%	1.69	3.85	3.87%	1.96	4.54	4.64%
	MTGNN	1.32	2.79	2.77%	1.65	3.74	3.69%	1.94	4.49	4.53%
	GMAN	1.34	2.91	2.86%	1.63	3.76	3.68%	1.86	4.32	4.37%
	DGCRN	1.28	2.69	2.66%	1.59	3.63	3.55%	1.89	4.42	4.43%
	PDFormer	1.32	2.83	2.78%	1.64	3.79	3.71%	1.91	4.43	4.51%
	STPGNN	1.35	2.88	2.85%	1.72	3.83	3.90%	2.10	4.72	5.03%
	PDTVGNN	**1.26**	**2.67**	**2.62%**	**1.55**	**3.53**	**3.48%**	**1.85**	**4.30**	**4.34%**
	Improv.	1.59%	0.75%	1.53%	2.58%	2.83%	2.01%	0.54%	0.47%	0.69%
METR-LA	VAR	4.42	7.89	10.20%	5.41	9.13	12.70%	6.52	10.11	15.80%
	SVR	3.99	8.45	9.30%	5.05	10.87	12.10%	6.72	13.76	16.70%
	FC-LSTM	3.44	6.30	9.60%	3.77	7.23	10.90%	4.37	8.69	13.20%
	DCRNN	2.77	5.38	7.30%	3.15	6.45	8.80%	3.60	7.60	10.50%
	STGCN	2.88	5.74	7.62%	3.47	7.24	9.57%	4.59	9.40	12.70%
	STSGCN	3.31	7.62	8.06%	4.13	9.77	10.29%	5.06	11.66	12.91%
	GWNet	2.69	5.15	6.90%	3.07	6.22	8.37%	3.53	7.37	10.01%
	AGCRN	2.87	5.58	7.70%	3.23	6.58	9.00%	3.62	7.51	10.38%
	MTGNN	3.31	7.62	8.06%	4.13	9.70	10.29%	5.06	11.66	12.91%
	GMAN	2.80	5.55	7.41%	3.12	6.49	8.73%	3.44	7.35	10.07%
	DGCRN	2.62	5.01	6.63%	2.99	6.05	8.19%	3.44	7.19	9.73%
	PDFormer	2.83	5.45	7.77%	3.20	6.46	9.19%	3.62	7.47	10.91%
	STPGNN	2.83	5.52	7.73%	3.25	6.59	8.91%	3.72	7.61	10.66%
	PDTVGNN	**2.59**	**4.93**	**6.51%**	**2.96**	**6.02**	**8.10%**	**3.39**	**7.15**	**9.62%**
	Improv.	1.16%	1.62%	1.84%	1.02%	0.50%	1.11%	1.47%	0.56%	1.14%

Table 3. Performance comparison of different methods on PEMS04 and PEMS08.

Datasets	Metric	VAR	SVR	FC-LSTM	DCRNN	STGCN	STSGCN	GWnet	AGCRN	MTGNN	GMAN	DGCRN	PDFormer	STPGNN	OURS	Improv.
PEMS04	MAE	24.44	26.18	23.60	24.42	23.90	21.52	19.91	19.36	19.50	19.25	18.80	18.32	18.34	**18.25**	0.38%
	RMSE	37.76	38.91	37.11	37.48	36.43	34.14	31.06	31.28	32.00	30.85	30.65	29.97	29.64	**29.59**	0.17%
	MAPE	17.27%	22.84%	16.17%	16.86%	13.67%	14.50%	13.62%	12.81%	14.04 %	13.00 %	12.82 %	**12.10%**	12.49%	12.14%	-0.33%
PEMS08	MAE	19.83	20.92	21.18	18.49	18.79	17.88	15.57	15.65	15.31	14.87	14.60	13.58	13.90	**13.50**	0.59%
	RMSE	29.94	31.23	31.88	27.30	28.20	27.36	24.32	24.99	24.42	24.06	24.16	23.51	23.05	**22.59**	2.04%
	MAPE	13.08%	14.24%	13.72%	11.69%	10.55%	11.71%	10.32%	10.17%	10.70 %	9.77%	9.33%	9.05%	9.01%	**8.98%**	0.33%

resulting in improved performance. **(4)** Most models utilizing attention mechanisms improve significantly over traditional time series and GNN-based models. Among them, GMAN and PDFormer outperform other baselines across different dataset metrics. However, our PDTVGNN model surpasses both, achieving superior performance. We attribute this to the intrinsic progressive decomposition capability provided by the series decomposition module in PDTVGNN, which effectively untangles entangled temporal patterns and enhances the model's ability to capture spatio-temporal dependencies.

5.3 Ablation Study (RQ2)

To further evaluate the effectiveness of modules of PDTVGNN, we compare it with the following variants on the METR-LA dataset.

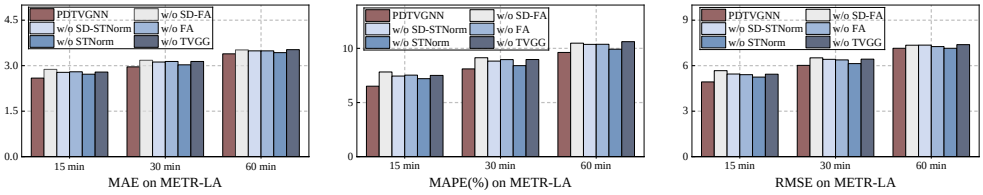


Fig. 3. Ablation study on METR-LA.

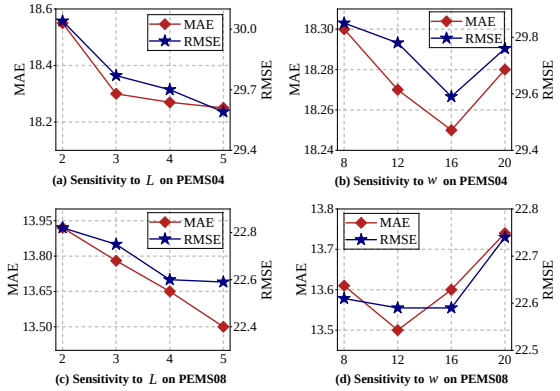


Fig. 4. The effect of model layers L and window size w on PEMS04 and PEMS08.

– **w/o SD-FA** removes the series decomposition and uses only the STNorm module to extract temporal features from the entangled traffic series.

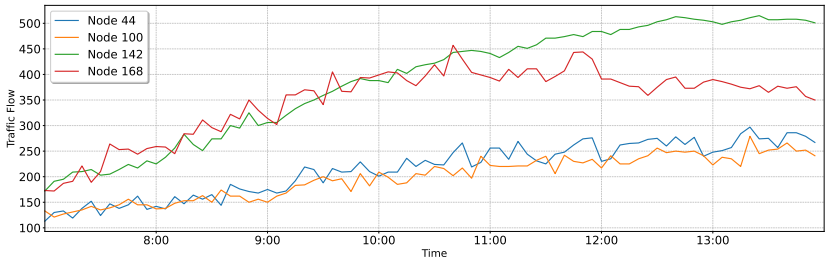
– **w/o SD-STNorm** removes the series decomposition module and uses only the Fourier temporal attention to extract temporal features from the entangled traffic series.

– **w/o FA** replaces the Fourier temporal attention with the STNorm for seasonal parts.

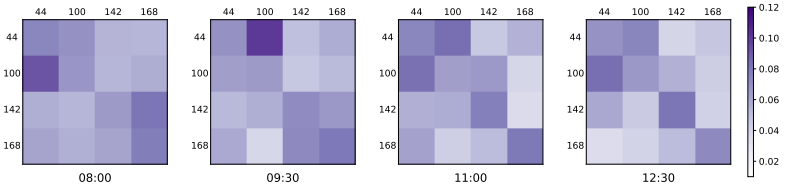
– **w/o STNorm** replaces the STNorm with the Fourier temporal attention for trend parts.

– **w/o TVGG** removes time-varying graph generator and directly uses a predefined geographic adjacency matrix for graph convolution.

From the results presented in Fig. 3, all components contribute to the final result to some extent, and we draw the following conclusions. **(1)** The significant drop in performance for w/o SD-FA and w/o SD-STNorm emphasizes the importance of progressive decomposition and separately modeling trend and seasonal parts. **(2)** PDTVGNN outperforms w/o FA, demonstrating the effectiveness of the STNorm module in modeling trend components accurately. **(3)** PDTVGNN further improves performance over w/o STNorm, suggesting that the Fourier temporal attention aids the model in capturing seasonal information better. **(4)** w/o TVGG, which relies on static geographic distance-based graphs, exhibits large standard deviations, highlighting the need for dynamic spatial dependencies modeling.



(a) The original time series curve



(b) Weight visualization of several adjacency matrices

Fig. 5. Case study of time-varying graph (from traffic flow on PEMS08 dataset).

5.4 Parameter Sensitivity (RQ3)

A key advantage of the PDTVGNN model is its ability to disentangle complex temporal patterns progressively, thanks to the progressive decomposition capabilities provided by its embedded decomposition blocks. The model ensures robustness and accuracy by modeling the different components separately. In the series decomposition module, the parameter w represents the window size for the moving average operation. Smaller window sizes tend to overfit short-term fluctuations and neglect long-term trends, while larger window sizes may excessively smooth the data, diminishing sensitivity to rapid fluctuations. In our experiments, we set the values of w to 8, 12, 16, and 20 on the PEMS04 and PEMS08 datasets, respectively. The specific experimental results are shown in Fig. 4(b) and (d). We found that the optimal performance on PEMS04 is achieved with $w = 16$ and on PEMS08 with $w = 12$.

We also examined the impact of the number of layers L on model performance by varying L from 2 to 5. The experimental results in Fig. 4(a) and (c) show that performance improves as the model depth increases, with the best results achieved when $L = 5$ for both datasets.

5.5 Case Study (RQ4)

To further validate the effectiveness of the time-varying graph structure generator and provide spatial interpretability, we selected four nodes numbered 44, 100, 142, and 168 from the PEMS08 dataset on August 14, 2016, for a case study. Fig. 5(b) shows the adjacency matrices at 8:00, 9:30, 11:00, and 12:30, visualized as heatmaps. The intensity of the purple grids indicates larger weights. Fig. 5(a)

also presents the original time series curves. As the relationships are one-way, we did not enforce symmetry in the adjacency matrix.

For node 142, before 11:00 and especially at 9:00, we observe a strong correlation between node 168 (represented by the red line) and node 142 (represented by the green line), with similar magnitudes of change. However, after 11:00, node 168 shows a decreasing trend while node 142 continues to increase. Adjacency matrixes effectively capture this shift in correlation, with corresponding matrix values decreasing over time. This supports our hypothesis that the spatial correlation of traffic data is time-varying. In contrast, when comparing the series trends of nodes 44 and 100, the correlation between them remains relatively stable and generally stronger, as reflected in the matrices. These observations provide compelling evidence for the effectiveness of the time-varying graph structure generator.

6 Conclusion

In this work, we propose a novel traffic forecasting model named PDTVGNN. Specifically, we adopt a progressive decomposition idea throughout the forecasting process, where decomposed blocks are embedded as internal modules to gradually decouple the entangled temporal patterns. The trend and seasonal parts obtained after decoupling are then modeled separately. Additionally, we introduce a time-varying graph generation module that constructs a series of adjacency matrices that process the current inputs and retain hidden information from the historical graph structure, capturing the time-varying nature of spatial relationships. Extensive experiments conducted on four real-world datasets demonstrate our proposed model's superiority and highlight each module's effectiveness.

Acknowledgments. This work is supported by National Natural Science Foundation of China No. 82374621.

Ethics Statement. The data used in this paper (PEMS04, PEMS08, PEMS-BAY, and METR-LA) are public open-sourced benchmark datasets, which are wildly used in academic research. No personally identifiable information was obtained and people can not infer personal information through the data. This work is not potentially a part of policing or military work. The authors of this paper are committed to ethical principles and guidelines in conducting research and have taken measures to ensure the integrity and validity of the data. The use of the data in this study is in accordance with ethical standards and is intended to advance knowledge in the field of traffic forecasting.

References

1. Tedjopurnomo, D.A., Bao, Z., Zheng, B., Choudhury, F., Qin, A.K.: IEEE Trans. Knowl. Data Eng. 34(4), 1544-1561 (2020)

2. Yin, X., Wu, G., Wei, J., Shen, Y., Qi, H., Yin, B.: Deep Learning on Traffic Prediction: Methods, Analysis, and Future Directions. IEEE Trans. Intell. Transp. Syst. 23(6), 4927-4943 (2022)
3. Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., Zhang, C.: Connecting the dots: multivariate time series forecasting with graph neural networks. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 753-763 (2020)
4. Song, C., Lin, Y., Guo, S., Wan, H.: Spatial-temporal synchronous graph convolutional networks: a new framework for spatial-temporal network data forecasting. In: AAAI Conference on Artificial Intelligence, pp. 914-921 (2020)
5. Kong, W., Guo, Z., Liu, Y.: Spatio-Temporal Pivotal Graph Neural Networks for Traffic Flow Forecasting. In: AAAI Conference on Artificial Intelligence, 8627-8635 (2024)
6. Jin, G., Liang, Y., Fang, Y., Shao, Z., Huang, J., Zhang, J., Zheng, Y.: Spatio-Temporal Graph Neural Networks for Predictive Learning in Urban Computing: A Survey. IEEE Trans. Knowl. Data Eng. 36(10), 1-20 (2024)
7. Williams, B.M., Hoel, L.A.: Modeling and forecasting vehicular traffic flow as a seasonal Arima process: theoretical basis and empirical results. J. Transp. Eng. 129(6), 664-672 (2003)
8. Zivot, E., Wang, J.: Vector autoregressive models for multivariate time series. In: Modeling Financial Time Series with S-Plus®, pp. 385-429. Springer, New York (2006). https://doi.org/10.1007/978-0-387-21763-5_11
9. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NeurIPS, pp. 3104-3112 (2014)
10. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: data-driven traffic forecasting. In: ICLR, pp. (2018)
11. Yu, B., Yin, H., Zhu, Z.: Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: IJCAI, pp. 3634-3640 (2017)
12. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Graph WaveNet for deep spatial-temporal graph modeling. In: IJCAI, pp. 1907-1913 (2019)
13. BAI, L., Yao, L., Li, C., Wang, X., Wang, C.: Adaptive graph convolutional recurrent network for traffic forecasting. In: NeurIPS, pp. 17804-17815 (2020)
14. Jiang, J., Han, C., Zhao, W.X., Wang, J.: PDFormer: Propagation Delay-Aware Dynamic Long-Range Transformer for Traffic Flow Prediction. In: AAAI Conference on Artificial Intelligence, pp. 4365-4373 (2023)
15. Li, F., Feng, J., Yan, H., Jin, G., Yang, F., Sun, F., Jin, D., Li, Y.: Dynamic Graph Convolutional Recurrent Network for Traffic Prediction: Benchmark and Solution. ACM Trans. Knowl. Discov. Data. 17(1), 1-21 (2023)
16. Bahdanau, D., Cho, K., Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate. arXiv preprint arXiv:1409.0473 (2014)
17. Vaswani, A., et al.: Attention is all you need. In: NeurIPS, pp. 5998-6008 (2017)
18. Robert B Cleveland, William S Cleveland, Jean E McRae, and Irma Terpenning: STL: A seasonal-trend decomposition. J. Off. Stat. 6(1), 3-73(1990)
19. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS) (2021)
20. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: Fedformer: frequency enhanced decomposed transformer for long-term series forecasting. In: Proceedings of the International Conference on Machine Learning (ICML) (2022)

21. Zhang, X., Jin, X., Gopalswamy, K., Gupta, G., Park, Y., Shi, X., Wang, H., Maddix, D.C., Wang, Y.: First De-Trend then Attend: Rethinking Attention for Time-Series Forecasting. arXiv preprint arXiv:2212.08151 (2022)
22. Guo, S., Lin, Y., Wan, H., Li, X., Cong, G.: Learning Dynamics and Heterogeneity of Spatial-Temporal Graph Data for Traffic Forecasting. IEEE Trans. Knowl. Data Eng. 34(11), 5415-5428 (2021)
23. Chen, M.,Wei, Z., Huang, Z., Ding, B., Li, Y.: Simple and deep graph convolutional networks. In: International Conference on Machine Learning, pp. 1725-1735 (2020)
24. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K., Jegelka, S.: Representation Learning on Graphs with Jumping Knowledge Networks. In: Proceedings of the International Conference on Machine Learning (ICML), pp. 5453-5462. (2018).
25. Zheng, C., Fan, X., Wang, C., Qi, J.: GMAN: a graph multi-attention network for traffic prediction. In: AAAI Conference on Artificial Intelligence, pp. 1234-1241 (2020)