Pattern Recognition 142 (2023) 109670

Contents lists available at ScienceDirect

Pattern Recognition

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

A Decomposition Dynamic graph convolutional recurrent network for traffic forecasting

Wenchao Weng[a], Jin Fan[a,b,*], Huifeng Wu[a,b], Yujie Hu[a], Hao Tian[c], Fu Zhu[a], Jia Wu[d]

[a] Hangzhou Dianzi University, Hangzhou, Zhejiang, China
[b] Zhejiang Provincial Key Laboratory of Industrial Internet in Discrete Industries, Hangzhou, Zhejiang, China
[c] Zhejiang Scientific Research Institute of Transport, Hangzhou, Zhejiang, China
[d] School of Computing, Macquarie University, Sydney, Australia

ARTICLE INFO

Article history:
Received 14 February 2023
Revised 23 April 2023
Accepted 30 April 2023
Available online 1 May 2023

Keywords:
Traffic forecasting
Dynamic graph generation
Residual decomposition
Segmented learning
Graph convolution network

ABSTRACT

Our daily lives are greatly impacted by traffic conditions, making it essential to have accurate predictions of traffic flow within a road network. Traffic signals used for forecasting are usually generated by sensors along roads, which can be represented as nodes on a graph. These sensors typically produce normal signals representing normal traffic flows and abnormal signals indicating unknown traffic disruptions. Graph convolution networks are widely used for traffic prediction due to their ability to capture correlations between network nodes. However, existing approaches use a predefined or adaptive adjacency matrix that does not accurately reflect real-world relationships between signals. To address this issue, we propose a decomposition dynamic graph convolutional recurrent network (DDGCRN) for traffic forecasting. DDGCRN combines a dynamic graph convolution recurrent network with an RNN-based model that generates dynamic graphs based on time-varying traffic signals, allowing for the extraction of both spatial and temporal features. Additionally, DDGCRN separates abnormal signals from normal traffic signals and models them using a data-driven approach to further improve predictions. Results from our analysis of six real-world datasets demonstrate the superiority of DDGCRN compared to the current state-of-the-art. The source codes are available at: https://github.com/wengwenchao123/DDGCRN.

1. Introduction

In recent years, deep learning technology has done much to advance intelligent transportation systems. Areas like trajectory prediction [1] and object recognition for autonomous driving [2], reinforcement learning for railway train scheduling [3], traffic signal control [4], and big data analysis [5] for travel demand modeling [6,7] have all been benefitted greatly from emerging deep learning applications.

Due to their capability to capture spatio-temporal dependencies, deep spatio-temporal neural networks have become a key application backbone. For this reason, they have been widely used in travel time estimation [8], traffic demand prediction [6,7], and traffic speed and flow prediction [9–12]. In the early stages of modelling these tasks, the distribution of traffic states was simply divided into grids of equal size [13]. The spatial dependencies of different grids were learned by CNNs [13–15], while the temporal dependencies were learned by RNNs [6,11,16]. Later, with further research, it was found that most traffic indicators were strongly related to particular points in the traffic network. In other words, the sensed traffic speeds and flows only pertain to that specific section of road [11,17]. Furthermore, it was discovered that some of these points can be fixed, such as pick-up demand at a station or passenger check-in locations at subway stations [6,7]. Hence, it makes more sense to model the correlations in the traffic system as a graph, with recent studies focusing on spatio-temporal graph modeling.

Spatio-temporal models utilize both spatial and temporal dimensions to model data, and their excellent performance has been attracting increasing attention. Owing to its outstanding versatility, spatio-temporal graph modeling has gained wide adoption in addressing diverse time series forecasting problems [12,18–20], including traffic prediction. There are currently many models developed for traffic prediction [11,21–23]. For example, Graph Multi-Attention Network (GMAN) [24] integrated temporal and spatial attention mechanisms. Choi et al. designed Spatio-Temporal Graph Neural Controlled Differential Equation (STG-NCDE) [22] which used Neural Controlled Differential Equations (NCDEs) to capture the spatio-temporal features of traffic conditions. Multivari-

* Corresponding author.
E-mail address: fanjin@hdu.edu.cn (J. Fan).

https://doi.org/10.1016/j.patcog.2023.109670
0031-3203/

ate Time Graph Neural Network (MTGNN) was proposed by Wu et al. [20] for mining the dependencies between segments of roads.

However, despite the good work of these studies, there are two key issues that are seldom addressed. The first issue is how to capture spatial dependencies between roads dynamically without relying on prior knowledge. In earlier studies, researchers used some prior knowledge (e.g., road segment distance, POI similarity) to construct a graph structure representing spatial correlation. Diffusion Convolutional Recurrent Neural Network (DCRNN) [11], Spatio-Temporal Graph Convolutional Network (STGCN) [23] are the representatives of the. They used the distance between road segments to compute node similarity to constructed a adjacency matrix representing the pre-defined graph structure. However, the adjacency matrix constructed by prior knowledge is not directly related to the task, and completely depends on the prior knowledge and the rationality of the construction method, which leads to the limited expression ability of the adjacency matrix. To solve this problem, Graph WaveNet [12], Coupled Layer-wise Convolutional Recurrent Neural Network (CCRNN) [6] used adaptive adjacency matrix to better extract spatial features with some success, but they still needed predefined graph structure to perform best. Adaptive Graph Convolutional Recurrent Network (AGCRN) [9] and MTGNN [20] further improved the adaptive collar matrix without using the predefined matrix, and achieved the same effect as the predefined matrix. However, both the predefined matrix and the adaptive matrix have a drawback that their weights are static. Traffic condition is a dynamic process, and its spatial correlation is constantly changing. Traffic networks change over time in a complex way that neither pre-defined graph structures nor adaptive matrixes can convey. In addition, the difference of traffic signals will also affect the effect of prediction. The present research objective is to identify a method that can dynamically model road correlation without the dependence on prior knowledge. This approach can alleviate the limitations in the scope of application scenarios and also reduce the complexity of modeling.

The second issue is how to distinguish between normal and abnormal signals. Most urban transportation networks are large-scale, enormously complex, and dynamic. Traffic signals naturally contain two kinds of signals, that is, normal signals representing normal traffic flows and abnormal signals representing abnormal flows due to unknown reasons. Current traffic prediction methods commonly treat all traffic signals as equal and do not differentiate between normal and abnormal traffic conditions. In other words, these methods solely identify and model the spatio-temporal correlations present. However, abnormal signals are not the same as normal signals. They are usually different from normal signals due to random factors. For example, the daily traffic flow on a highway usually moves toward a clear destination. Hence, traffic generally flows along a predetermined trajectory. However, under abnormal conditions, such as a traffic accident [25], some vehicles may change their destination. These abnormal signals may have patterns and trajectories that are unlike those of the normal signals. Digging deeply into these abnormal patterns to understand how they affect traffic conditions is likely to improve the quality of any subsequent traffic predictions.

To solve the these issues, we propose a Decomposition Dynamic Graph Convolutional Recurrent Network (DDGCRN) for traffic forecasting. To address the first problem - dynamically modeling spatial dependencies without a priori knowledge - a method for dynamic graph generation is presented. Here, temporal information corresponding to the traffic signal is combined with the spatial embedding to generate a spatio-temporal embedding. Then, a dynamic graph embedding is combined with dynamic signals extracted from the traffic signals to produce a dynamic graph. This method fully considers the periodicity and dynamics of the traffic signals so that the generated dynamic graph captures the most realistic correlations between nodes. Lastly, the spatio-temporal dependencies in the traffic signals are extracted via the Dynamic Graph Convolution Recurrent Module (DGCRM), which is based on an RNN.

To manage the second problem distinguishing between normal and abnormal signals we use the residual decomposition mechanism to subtract original signal and the reverse prediction signal, which isolates any abnormal signals. These isolated signals can then be modeled using an additional DGCRM. By adding the predictions of each module, the final prediction result is obtained. Considering the deficiency of the model in efficiency and resource occupation, we designed a training strategy named segmented learning according to the characteristics of DDGCRN prediction through residual decomposition. In the early stages of training, this strategy can reduce training time and memory consumption significantly without compromising performance.

In summary, this paper includes the following contributions:

- A method of dynamic graph generation. This method first generates a spatio-temporal embedding based on the time information in the traffic signals. These spatio-temporal embeddings are then combined with the dynamic signals extracted from the traffic signals to generate dynamic graph embeddings, which are used to generate dynamic graphs. In this way, the method can generate dynamic graph structures to extract spatial features without any prior knowledge. To extract the spatio-temporal features of traffic signals, we use dynamic graphs to construct an RNN-based Dynamic Graph Convolution Recurrent Module (DGCRM).
- An efficient and effective framework DDGCRN for traffic prediction. Our framework distinguishes between normal and abnormal traffic signals and models each separately. The relationships between traffic conditions and time points can be studied by analyzing the spatio-temporal embeddings.
- A novel training strategy that overcomes the shortcomings of DDGCRN in terms of efficiency and resource use. This training strategy reduces time and memory consumption during the early stages of training without affecting the model's subsequent performance.
- A comprehensive analysis of six real-world datasets confirms the effectiveness of DDGCRN. As compared to the current state-of-the-art models, DDGCRN produces significantly lower prediction errors.

2. Related work

2.1. Traffic forecasting

In the field of intelligent transportation systems, traffic forecasting has received extensive research. Traffic forecasting can be seen as a spatio-temporal prediction task that uses real-time data collected from road sensors to predict traffic conditions. Relevant technologies are crucial to building a smart city transportation system [26,27]. In early studies on traffic forecasting, traditional statistical methods (e.g., Auto-regressive Integrated Moving Average model (ARIMA)) [28] were used to predict traffic metrics. Later, machine learning methods (e.g., Vector AutoRegressive model (VAR)) [29,30] were applied to the problem with some success. However, these early machine learning methods relied heavily on assumptions related to stationarity [31], which meant they were largely ineffective for forecasting traffic data with highly nonlinear relationships. Additionally, these methods were only able to capture time information, while ignoring the complex correlations between time series, and this seriously limited their effectiveness for predicting traffic. In the field of traffic forecasting, deep neural network models [32] have emerged and are widely used. Unlike the earlier methods, DNNs are able to capture both temporal and spatial features. For example, Fully Connected Long Short-Term Memory (FC-LSTM) [32] modeled traffic data by combining CNN and LSTM. Spatio-Temporal Residual Network (ST-ResNet) [14] predicted crowd flows in cities by using a deep residual CNN network, reflecting the power of residual networks. However, although the above methods have achieved good results, they do not work well in scenarios with graph-based node data.

2.2. Spatio-temporal graph neural network

In recent years, spatio-temporal graph neural networks [6,9,16,21] have attracted more and more attention due to their advanced performance. As such, researchers have begun to introduce GCNs into traffic prediction to improve performance. At present, the most advanced works on traffic forecasting are largely based on GCNs. DCRNN [11], STGCN [23], and other models are the most representative works. These models captured spatial features between nodes through a predefined graph structure and captured the temporal features with a CNN or an RNN. However, these models rely heavily on manually predefined graph structures. The quality of predefined graph structures determines the pre performance of model predictions. To solve this problem, Graph Wavenet [12], MTGNN [20], and AGCRN [9] were invented. These frameworks adaptively generated graph structures in a data-driven way and have achieved excellent outcomes as a result. Recent research STG-NCDE [22] further improved performance by combining adaptive graphs and neural controlled differential equations. Generally, traffic data contain strong and dynamic spatio-temporal correlations. Hence, modeling dynamic nonlinear spatio-temporal correlation is crucial for accurate traffic prediction. The dynamic generation of dynamic graphs has emerged as a novel research direction. The Spatio-Temporal Graph Diffusion Network (ST-GDN) [33] leveraged multi-resolution traffic transition information and local-global inter-regional dependence to make predictions. The Dynamic Spatial-Temporal Aware Graph Neural Network (DSTAGNN) [34] captured the dynamic attribute of the spatial association between nodes by directly mining historical traffic flow data to extract temporal and spatial correlations. H. Peng et al. [35] proposed a traffic flow probability graph and used reinforcement learning to generate dynamic graphs for extracting temporal and spatial features.

3. Preliminaries

This section introduces fundamental concepts of traffic networks and traffic signals, and the prediction problems to be addressed. It also defines the GCN and its spatial and temporal properties.

Definition 1.**Traffic Network.** A traffic network can be represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}$ is a set of N nodes that represent sensors positioned at the corresponding locations in the road network. These sensors are responsible for recording traffic information at their locations. $\mathcal{E}$ is a set of edges, and $\mathcal{A}$ is a graph derived from the pairwise distances between nodes in the network, etc. Our framework involves a dynamic adjacency matrix, which is expressed as $A^d \in R^{N\times N}$.

Definition 2.**Traffic Signal.** A traffic signal $X_t \in R^{N\times C}$ represents the observed values of all sensors in the traffic network G at time step t, where C is the traffic features collected by the sensors.

Definition 3.**Traffic Forecasting.** Given historical traffic signals $\chi_P = [x_{t-P+1}, \ldots, x_{t-1}, x_t] \in R^{P\times N\times C}$, traffic forecasting aims to predict the future traffic signals $\gamma_Q = [y_{t+1}, y_{t-1}, \ldots, y_{t+Q}] \in R^{Q\times N\times C}$.

Definition 4.**GCN.** According to [36], a graph convolution operation can be well approximated by a first-order Chebyshev polynomial. This can be expressed as:

$$Z = (I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) X\Theta + b \tag{1}$$

where $A \in R^{N\times N}$ represents the graph structure, which is a semi-positive definite matrix, D is the degree matrix of A, and $X \in R^{N\times C}$ and $Z \in R^{N\times F}$ are the inputs and outputs of the GCN. $\Theta \in R^{C\times F}$ and $b \in R^F$ represent the weight and bias.

Definition 5.**Spatial and Temporal Properties.** Assuming that there are N nodes in the transportation network, and the frequency of sensor sampling data is N_d times per day, and there are 7 days in a week, the spatial and temporal features can be stored in three independent trainable embedding matrices, i.e., $E \in R^{N\times D}$, $T^D \in R^{N_d\times D}$, $T^W \in R^{N_w\times D}$. D represents the embedding dimension.

4. Model archithecture

This section introduces the details of DDGCRN. The architecture of DDGCRN can be seen in Fig. 1.

4.1. Dynamic graph generation

DDGCRN includes a dynamic graph generation method that captures the dynamic spatial dependencies in the sensor data. In this way, it does not rely on a predefined adjacency matrix, making it applicable to any scenario lacking prior knowledge.

To fully consider the dynamic spatial dependencies of road networks at different time points, a spatio-temporal embedding generator (STE generator) is designed. The core process of the generator finds the daily embedding T_P^D and the weekly embedding T_P^W of the time corresponding to the current traffic signal X_p. Then it performs an element-wise product operation between those embeddings and the spatial embedding E to obtain a new spatio-temporal embedding E_P^{st}, formulated as follows:

$$E_P^{st} = E \odot T_P^D \odot T_P^W \tag{2}$$

where $T_P^D, T_P^W \in R^{P\times N\times D}$ are the daily and weekly embeddings represented by the time step $P = [t-P+1, \ldots, t]$, and $\odot$ represents the element-wise product operation.

In time step t, the input of the current time step is passed through an MLP layer so as to extract the dynamic signals:

$$F_t = MLP(x_t) \tag{3}$$

where $F_t \in R^{N\times D}$ is the dynamic signal obtained after filtering. An element-wise multiplication operation is then performed on F_t and E_t^{st} to generate a dynamic graph embedding, expressed as follows:

$$E_t^d = tanh(F_t \odot E_t^{st}) \tag{4}$$

Next, similar to defining a graph according to node similarity, the spatial dependencies can be inferred by multiplying E_t^d and ${E_t^d}^T$. Then, to meet the requirements of Chebyshev polynomials, the generated dynamic matrix is normalized as follows:

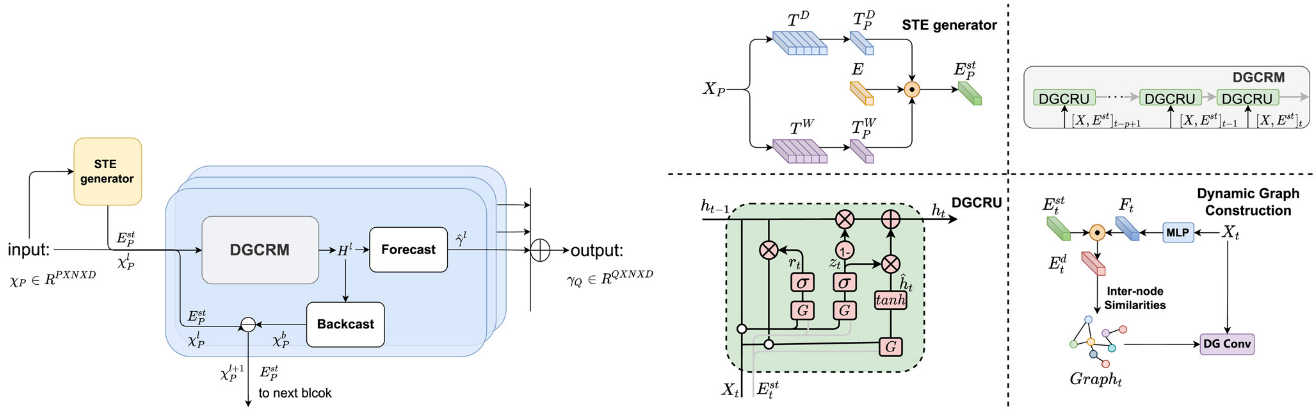
$$D_t^{-\frac{1}{2}} A_t^d D_t^{-\frac{1}{2}} = D_t^{-\frac{1}{2}} (ReLU(E_t^d {E_t^d}^T)) D_t^{-\frac{1}{2}} \tag{5}$$

where $A_t^d \in R^{N\times N}$ represents the dynamic graph at time step t. This is substituted into Eq. 1 to give the dynamic convolution formula, expressed as:

$$\begin{aligned} Z_t &= (I_N + D_t^{-\frac{1}{2}} A_t^d D_t^{-\frac{1}{2}}) X\Theta + b \\ &= (I_N + D_t^{-\frac{1}{2}} (ReLU(E_t^d {E_t^d}^T)) D_t^{-\frac{1}{2}}) X\Theta + b \end{aligned} \tag{6}$$

4.2. Node adaptive parameter learning

The GCN is optimized through a module called the node adaptive parameter learning (NAPL) module. Based on matrix decom-



(a) The architecture of DDGCRN (b) The modules of DDGCRN

Fig. 1. The architecture and modules of DDGCRN.

position, this module lets the model learn a unique traffic pattern for each node. Here, the weight matrix is set to $\theta \in R^{N\times c\times F}$. Then, to optimize the GCN while preventing overfitting, the weight matrix is decomposed into a node parameter matrix $E_g \in R^{N\times d}$, and two weight matrices $W_g \in R^{d\times c\times F}$ and $b_g \in R^{d\times F}$, where $d <= N$ and $\theta = E_g W_g$, $b = E_g b_g$. The optimized GCN formula is as follows:

$$Z = (I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) X E_g W_g + E_g b_g \tag{7}$$

4.3. Dynamic graph convolution recurrent module

Traffic forecasting involves complex temporal correlations as well as spatial correlations. In RNNs, gated recurrent units (GRUs) [37] capture temporal and spatial features effectively. Based on the work of [11], a dynamic graph convolution gated recurrent unit (DGCRU) is obtained by replacing the matrix product in a GRU with a combination of a dynamic graph convolution method and an NAPL module. The DGCRU can be expressed as follows:

$$\begin{aligned} r_t &= \sigma\left(\theta\left[x_t \middle\| H_{t-1}, E_t^{st}\right] E W_r + E b_r\right) \\ u_t &= \sigma\left(\theta\left[x_t \middle\| H_{t-1}, E_t^{st}\right] E W_u + E b_u\right) \\ \hat{h}_t &= tanh\left(\theta\left[x_t \middle\| r_t \odot H_{t-1}, E_t^{st}\right] E W_c + E b_c\right) \\ H_t &= u_t \odot H_{t-1} + (1 - u_t) \odot \hat{h}_t \end{aligned} \tag{8}$$

where x_t and H_t are the input and output at time step t, σ denotes the sigmoid activation, θ represents the dynamic graph generation, $\|$ represents the concatenation operation. W_r, W_u, W_c and b_r, b_u, b_c are learnable parameters.

The DGCRM, which is composed of DGCRUs, are used to extract the spatio-temporal characteristics from the traffic signal sequences. The hidden state H_P of the last DGCRU is taken as the output H^l of the DGCRM. Note that all DGCRMs share a parameter matrix E.

4.4. Residual decomposition and multi-step prediction

To implement multi-step prediction and signal decomposition, the framework includes an output sublayer consisting of linear layers after the DGCRM. This output sublayer has two outputs:

$$\begin{aligned} \hat{y}^l &= Linear_{l,f}(H^l) \\ x_b^l &= Linear_{l,b}(H^l) \end{aligned} \tag{9}$$

The forecasting output $\hat{y}^l \in R^{Q\times N\times C}$ represent the forecasting of y by the l^{th} block through x^l, while the backcasting output $x_b^l \in R^{P\times N\times C}$ represents the reverse forecasting of x^l by the l^{th} block. The model's final output is:

$$\begin{aligned} y \; &= \textstyle\sum_n^l \hat{y}^l \\ x_P^{l+1} &= x_P^l - x_b^l \end{aligned} \tag{10}$$

Reverse prediction can be thought of a process of traffic signal decomposition. Specifically, x_b^l can be regarded as the reconstruction of the input signal x_P^l, which contains the information H^l learns from x_P^l. Through residual decomposition, the information that has been learned in x_P^l is removed, and only the part x_P^{l+1} that have not been learned is reserved for modeling in the next block. Each block's output values are summed to form the final prediction.

4.5. Segmented learning training strategy

Improving the accuracy of a model often means complicating the model to achieve better results. However, this usually results in greatly increased time and hardware requirements to train the model. As such, ways to effectively reduce the memory and time consumed during training is a hot topic in deep learning. Drawing inspiration from MTGNN [20] and Dynamic Graph Convolutional Recurrent Network (DGCRN) [20], an efficient and effective general training strategy is designed based on the features of decomposing the models predictions. The scheme is called segmented learning. Early in the training phase, only the first L blocks are trained, not all the blocks. Then, as training continues, more blocks are gradually added. Thus, the model is trained gradually, substantially reducing the time and memory needed in the early stages of training. An overview of the segmented learning scheme can be found in Algorithm 1.

Algorithm 1 Training algorithm of DDGCRN.

Input: The traffic signals χ_P,time embeddings T^D and T^W,node embeddings E,step size s,the number of blocks K.

$epoch = 1$, $L = 1$;
repeat
  if $eopch\%s == 0$ and $L < K$ **then**
    $L = L + 1$;
  end if
  for $l = 1$; $l < L$; $l++$ **do**
    Calculate $\hat{y}^l$, $\hat{x}_b^l$ according to Eq.9, Eq.10
  end for
  sum $\hat{\gamma}_Q = \sum_L \hat{y}^l$
  Calculate $L = loss(\hat{\gamma}_Q, \gamma_Q)$
  Back propagation and update parameters according to L
  $epoch = epoch + 1$
until convergence of the model is achieved

Output: learned model.

Table 1
Description and statistics of the datasets.

Datasets	nodes	rate	Time steps	Time Range	Tpye
PEMSD3	358	26208	5min	2018.09–2018.11	Volume
PEMSD4	307	16992	5min	2018.01–2018.02	Volume
PEMSD7	883	28224	5min	2017.05–2017.08	Volume
PEMSD8	170	17856	5min	2016.07–2016.08	Volume
PEMS07(M)	223	12672	5min	2012.05–2012.06	Velocity
PEMS07(L)	1026	12672	5min	2012.05–2012.06	Velocity

5. Experimental evaluation

5.1. Datasets

To evaluate the performance of DDGCRN, a series of large-scale experiments are conducted using six real-world datasets: PeMSD3, PeMSD4, PeMSD7, PeMSD8, PeMS07(M), PeMS07(L). These datasets contain sensor information collected every 30 seconds by the Caltrans Performance Measure System (PeMS) [38]. They have been widely used in many previous studies. For more details on the dataset, see Table 1, where Tpye represents the prediction target of the dataset.

5.2. Experimental settings

The datasets are split into training, validation, and test sets at a ratio of 6:2:2, consistent with the benchmarks used in previous studies [17,21,22]. The prediction of traffic conditions for the next hour is based on the conditions observed over the previous hour. Specifically, Q is set to 12 and P is set to 12.

All the experiments are carried out on a Win11 computer equipped with a NVIDIA GeForce GTX 3090 graphics card. With regards to hyperparameters, the number of hidden units in the DCGRU is set to 64 and the number of blocks K to 2. The embedding dimensions D are set as follows: 12 for PEMSD3, 10 for PEMSD4, 12 for PEMSD7, 5 for PEMSD8, 8 for PEMS07(M), and 15 for PEMS07(L).

The model is trained using the Adam optimizer. The training epoch is set to 100. With the PEMSD7 and PEMS07(L) dataset, the batch size is set to 16 and the learning rate is set to 0.0075 due to the large amount of memory used by the graph convolution operation. With the other datasets, the batch size is set to 64 and the learning rate is 0.03. Models are trained using MAE as the loss function.

To evaluate performance, three evaluation metrics are utilized: (1) mean absolute error (MAE), (2) root mean square error (RMSE), and (3) mean absolute percentage error (MAPE).

5.3. Baselines

DDGCRNs performance is compared to 21 baselines, including traditional models and state-of-the-art works:

History Average Model (HA) [28] calculates the next value based on the average of the last 12 time steps.

ARIMA [39] combines auto-regression with a moving average model to predict the future by fitting time series data.

VAR[28] captures correlations in traffic sequence data.

Temporal Convolutional Network (TCN)[40] uses causal convolutions to model temporal features.

FC-LSTM[32] model the data by combining a CNN and a LSTM.

GRU-ED[32] is a simple and effective time series forecasting model.

Dual Self-Attention Network (DSANet)[41] uses a CNN and a dual attention mechanism for prediction.

DCRNN[11] is a baseline based on diffusion graph convolution that uses SeqtoSeq framework for traffic prediction.

STGCN[23] captures spatio-temporal features by using GCN and CNN.

Graph WaveNet[12] extracts hidden spatial features by using an adaptive matrix.

Spatial-temporal Graph to Sequence Model (STG2Seq)[42] uses gated long-term and short-term encoders to fuse long-term and short-term temporal features.

Long Short-term Graph Convolutional Network (LSGCN)[43] combines a new graph attention network cosAtt and GCN to accurately capture spatial features, while GLU is used to capture temporal features.

Spatial-Temporal Synchronous Graph Convolutional Network (STSGCN)[44] uses local spatio-temporal graph convolution to synchronously capture spatio-temporal correlation.

Spatial-Temporal Fusion Graph Neural Network (STFGNN)[45] captures spatio-temporal correlation through the fusion of multiple spatial and temporal graphs.

Spatial-Temporal Graph ODE Network (STGODE)[46] applies CGNN (continous graph neural network) to deal with the problem of GCN over-smooth and extract spatio-temporal dependence.

AGCRN[9] captures spatio-temporal features by using an adaptive graph and a GRU.

Attention Based Spatial-Temporal Graph Convolutional Network (ASTGCN)[17] combines an attention mechanism with graph convolution to model traffic data.

MSTGCN[17] is a variant of ASTGCN.

Time Zigzags at Graph Convolutional Networks (Z-GCNETs)[21] incorporate the concept of zigzag persistence into GCNs to enhance performance.

STG-NCDE[22] predicts traffic using two neural controlled differential equations.

6. Experimental results

6.1. Forecasting performance comparison

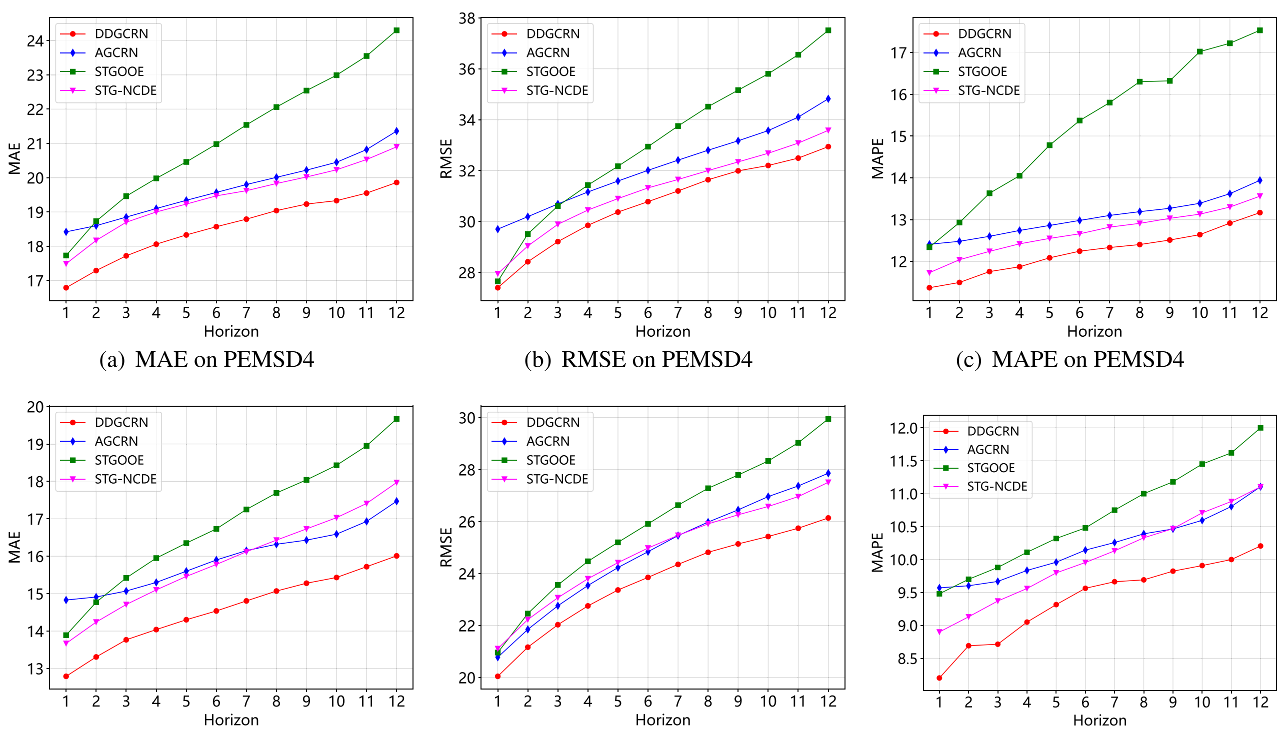
Table 2 compares each model's performance based on the six datasets. Overall, our model achieves optimum accuracy within 12 time steps. As traditional methods and machine learning models only model temporal correlations, and they have strong requirements for the stationarity of the data, it can be difficult for traffic data to meet these requirements. Based on Table 2, it is evident that these approaches exhibited the poorest performance. The non-graph-based models, such as TCN and FC-LSTM, use deep learning methods to capture capture temporal features in traffic data, achieving good results. However, these models ignore the correlation in the spatial dimension, so their accuracy falls short of graph-based models, which also proves why it is so important to also model the spatial correlations.

In terms of the graph-based models, DCRNN uses predefined graph structures to capture spatial correlations. However, the quality of the predefined graph structures greatly affects the models final output. AGCRN, STG-NCDE and other models model the spatial relationships by generating adaptive matrices. Generally, they achieve excellent results. However, these models still model spatial relationships through static graph structures, and do not consider dynamic changes in the data. Compared to other models, DDGCRN extracts dynamic signals from the sensor data to construct a more suitable adjacency matrix. The result is better predictions.

The prediction error metrics at each step are visualized using the PEMSD4 and PEMSD8 datasets to facilitate a better comparison of DDGCRN with the other baselines. As shown in Fig. 2, although STGODE performs well at ultra-short-term predictions, its single-step error increases much faster than that of the other models. In fact, its average error rate is the worst among all the latest models. While AGCRN outperforms STG-NCDE in long-term prediction

Table 2
Comparison of DDGCRN and baselines on six traffic datasets.

Model	PEMSD3			PEMSD4			PEMSD7			PEMSD8			PEMS07(M)			PEMS07(L)		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
HA [28]	31.58	52.39	33.78%	38.03	59.24	27.88%	45.12	65.64	24.51%	34.86	59.24	27.88%	4.59	8.63	14.35%	4.84	9.03	14.90%
ARIMA [39]	35.41	47.59	33.78%	33.73	48.80	24.18%	38.17	59.27	19.46%	31.09	44.32	22.73%	7.27	13.20	15.83%	7.51	12.39	15.83%
VAR [28]	23.65	38.26	24.51%	24.54	38.61	17.24%	50.22	75.63	32.22%	19.19	29.81	13.10%	4.25	7.61	10.28%	4.45	8.09	11.62%
FC-LSTM [32]	21.33	35.11	23.33%	26.77	40.65	18.23%	29.98	45.94	13.20%	19.19	29.81	13.10%	4.16	7.51	10.10%	4.66	8.20	11.69%
TCN [40]	19.32	33.55	19.93%	23.22	37.26	15.59%	32.72	42.23	14.26%	22.72	35.79	14.03%	4.36	7.20	9.71%	4.05	7.29	10.43%
TCN(w/o causal)	18.87	32.24	18.63%	22.81	36.87	14.31%	30.53	41.02	13.88%	21.42	34.03	13.09%	4.43	7.53	9.44%	4.58	7.77	11.53%
GRU-ED [32]	19.12	32.85	19.31%	23.68	39.27	16.44%	27.66	43.49	12.20%	22.00	36.22	13.33%	4.78	9.05	12.66%	3.98	7.71	10.22%
DSANet [41]	21.29	34.55	23.21%	22.79	35.77	16.03%	31.36	49.11	14.43%	17.14	26.96	11.32%	3.52	6.98	8.78%	3.66	7.20	9.02%
STGCN [23]	17.55	30.42	17.34%	21.16	34.89	13.83%	25.33	39.34	11.21%	17.50	27.09	11.29%	3.86	6.79	10.06%	3.89	6.83	10.09%
DCRNN [11]	17.99	30.31	18.34%	21.22	33.44	14.17%	25.22	38.61	11.82%	16.82	26.36	10.92%	3.83	7.18	9.81%	4.33	8.33	11.41%
GraphWaveNet [12]	19.12	32.77	18.89%	24.89	39.66	17.29%	26.39	41.50	11.97%	18.28	30.05	12.15%	3.19	6.24	8.02%	3.75	7.09	9.41%
ASTGCN(r) [17]	17.34	29.56	17.21%	22.92	35.22	16.56%	24.01	37.87	10.73%	18.25	28.06	11.64%	3.14	6.18	8.12%	3.51	6.81	9.24%
MSTGCN [17]	19.54	31.93	23.86%	23.96	37.21	14.33%	29.00	43.73	14.30%	19.00	29.15	12.38%	3.54	6.14	9.00%	3.58	6.43	9.01%
STG2Seq [42]	19.03	29.83	21.55%	25.20	38.48	18.77%	32.77	47.16	20.16%	20.17	30.71	17.32%	3.48	6.51	8.95%	3.78	7.12	9.50%
LSGCN [43]	17.94	29.85	16.98%	21.53	33.86	13.18%	27.31	41.46	11.98%	17.73	26.76	11.20%	3.05	5.98	7.62%	3.49	6.55	8.77%
STSGCN [44]	17.48	29.21	16.78%	21.19	33.65	13.90%	24.26	39.03	10.21%	17.13	26.80	10.96%	3.01	5.93	7.55%	3.61	6.88	9.13%
AGCRN [9]	15.98	28.25	15.23%	19.83	32.26	12.97%	22.37	36.55	9.12%	15.95	25.22	10.09%	2.79	5.54	7.02%	2.99	5.92	7.59%
STFGNN [45]	16.77	28.34	16.30%	20.48	32.51	16.77%	23.46	36.60	9.21%	16.94	26.25	10.60%	2.90	5.79	7.23%	2.99	5.91	7.69%
STGODE [46]	16.50	27.84	16.69%	20.84	32.82	13.77%	22.59	37.54	10.14%	16.81	25.97	10.62%	2.97	5.66	7.36%	3.22	5.98	7.94%
Z-GCNETs [21]	16.64	28.15	16.39%	19.50	31.61	12.78%	21.77	35.17	9.25%	15.76	25.11	10.01%	2.75	5.62	6.89%	2.91	5.83	7.33%
STG-NCDE [22]	15.57	27.09	15.06%	19.21	31.09	12.76%	20.53	33.84	8.80%	15.45	24.81	9.92%	2.68	5.39	6.76%	2.87	5.76	7.31%
DDGCRN	**14.63**	**25.07**	**14.22%**	**18.45**	**30.51**	**12.19%**	**19.79**	**33.11**	**8.35%**	**14.40**	**23.75**	**9.40%**	**2.59**	**5.21**	**6.48%**	**2.79**	**5.68**	**7.06%**



(a) MAE on PEMSD4 (b) RMSE on PEMSD4 (c) MAPE on PEMSD4
(d) MAE on PEMSD8 (e) RMSE on PEMSD8 (f) MAPE on PEMSD8

Fig. 2. Prediction error at each horizon on PEMSD4 and PEMSD8.

accuracy on the PEMSD8 dataset, STG-NCDE still has a better average effect than AGCRN. Finally, DDGCRN has significantly lower error rates than that of the other baselines over all time horizons, verifying the superior performance of our model.

6.2. Computation cost

We also compare the efficiency of DDGCRN against the other baselines with the PEMSD4 and PEMSD8 datasets in terms of both training time and inference time per epoch. More specifically, we compare the speed of DDGCRN, AGCRN, STFODE, and STG-NCDE without segmented learning. For all models, the batch size is set to 64.

As shown in Table 3, DDGCRN and STGODE have the same computation cost, second only to AGCRN. However, the accuracy is much better than the other baselines. Thanks to its simple adaptive graph construction method, AGCRN has the fastest calculation speed, but, since it does not consider time points and dynamic changes, its scores are not as good as STG-NCDE. STG-NCDE works well, but because it uses ordinary differential equations (Odes) for its modelling, it has a relatively long computation time, so it is far less fast and efficient than DDGCRN.

Table 3
The computation time on PEMSD4 and PEMSD8 datasets.

Dataset	Models	Computation Time	
		Training (s/epoch)	Inference (s)
PEMSD4	AGCRN	6.5	1.1
	STGODE	35.2	4.1
	STG-NCDE	118.6	12.3
	DDGCRN	38.1	3.8
PEMSD8	AGCRN	3.9	0.5
	STGODE	22.3	2.1
	STG-NCDE	43.2	4.3
	DDGCRN	18.6	1.7

7. Ablation study

This section aims to demonstrate the effectiveness of the key components and course learning of our model through a series of experiments. The following variants are designed to verify the effects of different model compositions:

w/o DG: eliminates the dynamic graph and only uses the node adaptive parameter learning module for modeling. This variant tests the contribution of dynamic graph convolution.

w/o SL: does not use the segmented learning strategy to train the model. This variant is designed to test whether the learning strategy affects model performance.

w/o NA: eliminates the node adaptive parameter learning module and only uses basic matrix multiplication operation. This variant tests the contributions of the node adaptive parameter learning module.

Each experiment is repeated five times, and the resulting mean MAE, RMSE, and MAPE values for each variant on PEMSD4 and PEMSD8 are presented in Table 4.

As shown, DDGCRN outperforms **w/o DG**, indicating that the dynamic graph plays a positive role in the framework. **w/o SL** performs similarly to DDGCRN, demonstrating that segmented learning can reduce memory requirements and training time without affecting the model's performance. Finally, **w/o NA** does not perform as well as DDGCRN, proving that the node adaptive parameter learning module makes a solid contribution to DDGCRNs overall performance.

7.1. Effect analysis of different types of graph structure

The purpose of this subsection is to analyze the impact of different graph structures on the performance of the model. To this end, four variants have been designed which each explore the utility of the dynamic graphs:

DDGCRN-dg: This is DDGCRN with dynamic graph convolution.

DDGCRN-ag: Replace the dynamic graph with the adaptive graph from AGCRN.

DDGCRN-c: Use the processed adjacency matrix attached to the dataset as the predefined graph for graph convolution. The weights between the connected nodes of the predefined graph are all set to 1, and it is Laplacian matrix normalized.

Table 4
Ablation experiments on PEMSD4 and PEMSD8.

PEMSD4	DDGCRN	**w/o DG**	**w/o SL**	**w/o NA**
MAE	18.45	22.90	18.44	18.64
RMSE	30.51	36.66	30.88	30.56
MAPE	12.19%	15.57%	12.24%	12.34%
PEMSD8	DDGCRN	**w/o DG**	**w/o SL**	**w/o NA**
MAE	14.4	17.83	14.39	14.52
RMSE	23.75	28.70	23.72	23.75
MAPE	9.40%	11.54%	9.50%	9.53%

Table 5
The effect of different types of graph structures.

PEMSD4	**DDGCRN-dg**	**DDGCRN-ag**	**DDGCRN-c**	**DDGCRN-d**
MAE	18.45	18.97	20.42	21.23
RMSE	30.51	31.28	33.23	34.15
MAPE	12.19%	12.60%	13.47%	14.11%
PEMSD8	**DDGCRN-dg**	**DDGCRN-ag**	**DDGCRN-c**	**DDGCRN-d**
MAE	14.4	15.15	15.88	16.37
RMSE	23.75	24.51	25.16	26.24
MAPE	9.40%	9.79%	10.29%	10.62%

DDGCRN-d: This variable uses the inverse of the distance between the sensors represented by the nodes as the weight of the edges.

The results of the four variants are shown in Table 5. There is no doubt that **DDGCRN-dg** and **DDGCRN-ag** perform significantly better than other variants using pre-defined graphs, proving the limitations of artificially designed, predefined graphs in representing spatial correlations on transportation networks. The superior performance of **DDGCRN-dg** over **DDGCRN-ag** also indicates that the traffic conditions are not fixed, and so can only be improved by considering the dynamic changes in the network. In addition, the **DDGCRN-c** performes significantly better than **DDGCRN-d**, which also shows that even if the graph structure is the same, an unreasonable weight design will greatly affect graph convolution. This also proves the limitations of artificially constructed graph structures.

7.2. Analysis of the node adaptive parameter learning module

To test the practicability of the NAPL module, three different module variants are designed:

use NA: This is the DDGCRN with using the NAPL module.

use ND: This variant uses a node dynamic parameter learning module. Its corresponding GCN formula is:

$$Z^t = (I_N + D_t^{-\frac{1}{2}} A_t^d D_t^{-\frac{1}{2}}) X^t E_t^d W_g + E_t^d b_g \tag{11}$$

This module replaces the node embedding E in the dynamic parameter learning module with the dynamic graph embedding E_t^d, so as to achieve dynamic adjustment.

no NA: This variant does not use the node adaptive parameter learning module. Rather, it simply uses the simplest linear multiplication for graph convolution.

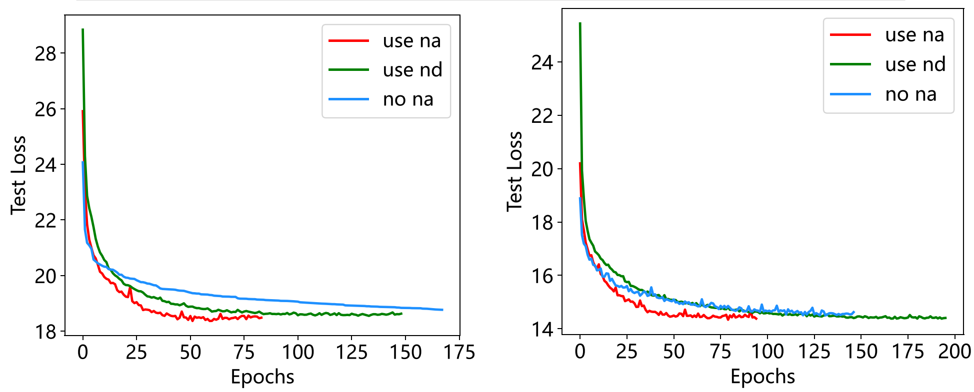
These experiments are conducted with the PEMSD4 and PEMSD8 datasets. For the convenience of comparison, our batchsizes for PEMSD4 and PEMSD8 are 16 and 32 respectively, and our learning rates for PEMSD4 and PEMSD8 are 0.00075 and 0.0015, respectively.

Table 6 shows the computation time and GPU cost required for each of the three variants. It can be seen from Table 6 that **use NA** and **no NA** do not have a big gap between them. However, **use ND's** computation time and GPU cost are much larger than the other two variants. This is because the node dynamic parameter learning module generates a five-dimensional tensor during its process, which greatly increases the complexity of the model. Because this leads to such high resource requirements, it is not conducive to practically deploying the model.

Fig. 3 compares the test loss curves for the test sets of the PEMSD4 and PEMSD8 datasets. Although the loss curves decrease rapidly in the early stage, the accuracy of **no NA** is not as good as that of **use NA** or **use ND**. Further, although the accuracy of **use ND** is comparable to that of **use NA**, its loss curve decreases slowly. In fact, **use ND** requires twice as many epochs as **use NA** to reach its peak accuracy. Additionally, the training time per epoch for **use ND** is more than twice as long as **use NA**. As a

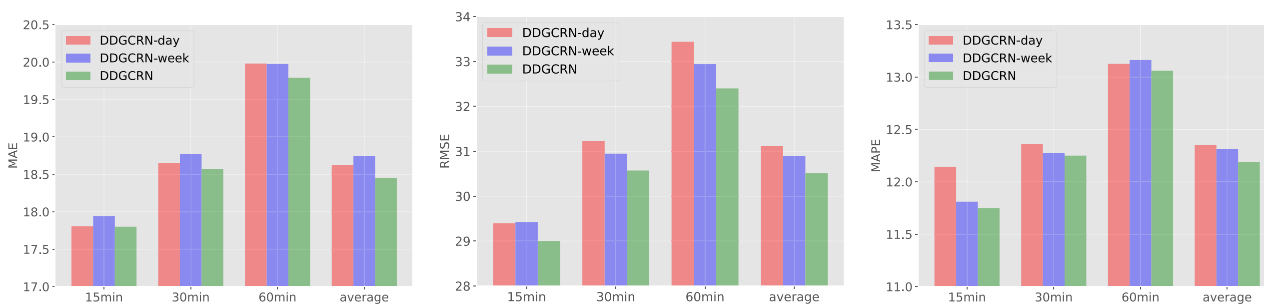
Table 6
Calculating cost of the three variants.

Dataset	variant	Computation Time		GPU cost
		Training (s/epoch)	Inference (s)	GPU memory (GB)
PEMSD4(16)	**use NA**	65.2	6.7	6.7
	use ND	148.8	17.9	16.3
	no NA	57.1	6.1	4.9
PEMSD8(32)	**use NA**	28.3	2.8	4.5
	use ND	80.8	9.8	16.7
	no NA	25.4	2.6	4.2

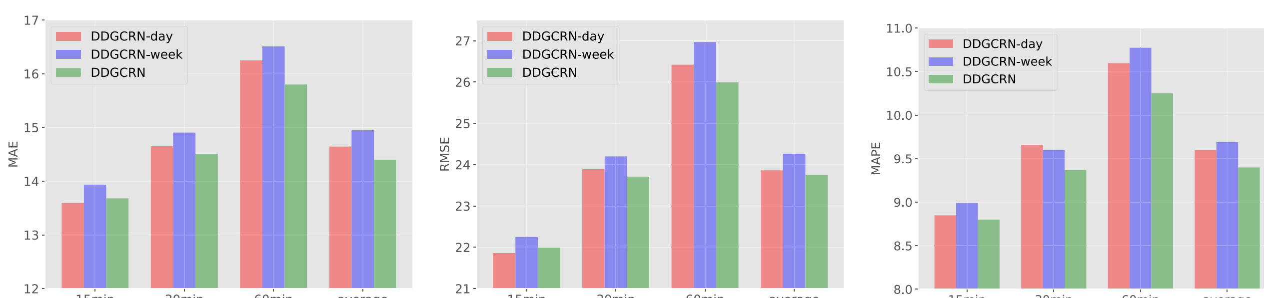


(a) Testset Loss Curve on PEMSD4. (b) Testset Loss Curve on PEMSD8.

Fig. 3. Testset Loss Curve for Three Variations.



(a) MAE on PEMSD4 (b) RMSE on PEMSD4 (c) MAPE on PEMSD4



(d) MAE on PEMSD8 (e) RMSE on PEMSD8 (f) MAPE on PEMSD8

Fig. 4. Ablation experiment of time embedding.

result, it take three to four time as long for **use ND** to reach the same level of performance as **use NA** through training. This is not cost-effective. Overall, **use NA** reaches optimum performance at a lower cost and with a faster training speed than the other two variants. The NAPL module is chosen to optimize the GCN due to this reason.

7.3. Time embedding

To verify the effectiveness of using two different time embeddings - daily and weekly, two DDGCRN variants are prepared for the next set of ablation experiments.

DDGCRN-day: this variant removes the weekly embedding and uses only the daily embedding to generate the dynamic embedding.

DDGCRN-week: This variant uses only the weekly embedding to generate the dynamic embedding.

Experiments are conducted with the PEMSD4 and PEMSD8 datasets, and the mean values for all metrics are reported. Fig. 4 illustrates the results. In short, both kinds of time embedding are effective. Specifically, DDGCRN-day appears to be better than DDGCRN over the short-term, but is not as good overall. Real-world traffic signals usually contain daily and weekly periodicities and

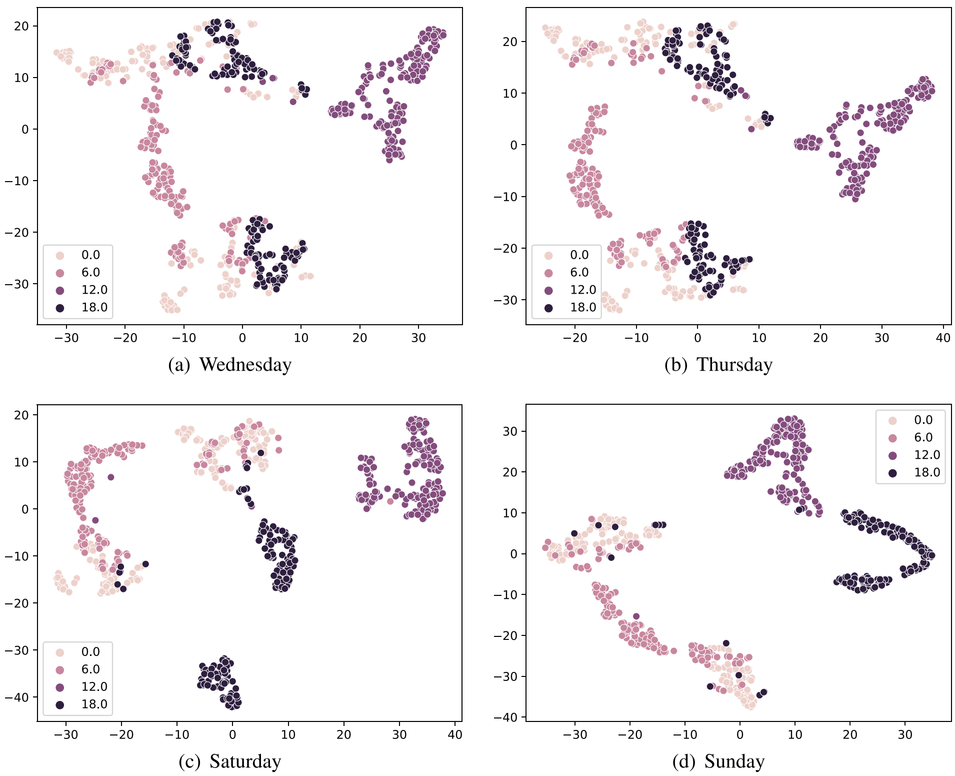


Fig. 5. Visualization of spatio-temporal embeddings at different times.

Table 7
Hyper-parameter Analysis: the number of blocks K.

Dataset	blocks	15min			30min			60min			average		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PEMSD4	1	18.01	29.51	11.99%	18.73	31.03	12.43%	20.06	33.23	13.24%	18.77	31.00	12.45%
	2	17.80	29.33	11.75%	18.57	30.93	12.25%	19.79	32.92	13.06%	18.45	30.51	12.19%
	3	17.75	29.27	11.66%	18.61	30.93	12.13%	19.88	33.06	12.93%	18.57	30.97	12.13%
PEMSD8	1	14.07	22.44	9.15%	14.80	24.07	9.86%	16.30	26.27	11.17%	14.87	24.00	10.16%
	2	13.68	21.99	8.80%	14.51	23.71	9.37%	15.80	25.99	10.25%	14.40	23.75	9.40%
	3	13.62	21.94	8.87%	14.50	23.84	9.42%	15.83	26.22	10.34%	14.47	23.81	9.43%

this is reflected in the contribution of both embeddings to the final result.

7.4. Visualization

To further examine the difference between the spatial correlations in traffic conditions at different time points on different days, we visualize the daily spatio-temporal embeddings for the hours 0, 6, 12, and 18, and the weekly embeddings for Wednesday, Thursday, Saturday, and Sunday. It is worth noting that t-SNE [47] is utilized to reduce the dimensions of the embedding to two dimensions for this analysis.

As can be clearly seen from Fig. 5, the embeddings at each time point have obvious clusterings. On Wednesday and Thursday, many embeddings cluster together at 0:00 and 18:00, indicating that their traffic conditions are similar. However, the same cannot be said of Saturday and Sunday. The difference here is workdays vs. weekends. In addition, the embedding for 12:00 does not cluster with other time points either on weekdays or weekends. This is consistent with the actual traffic conditions at 12:00, which are congested but rather smooth at other times.

8. Parameter analysis

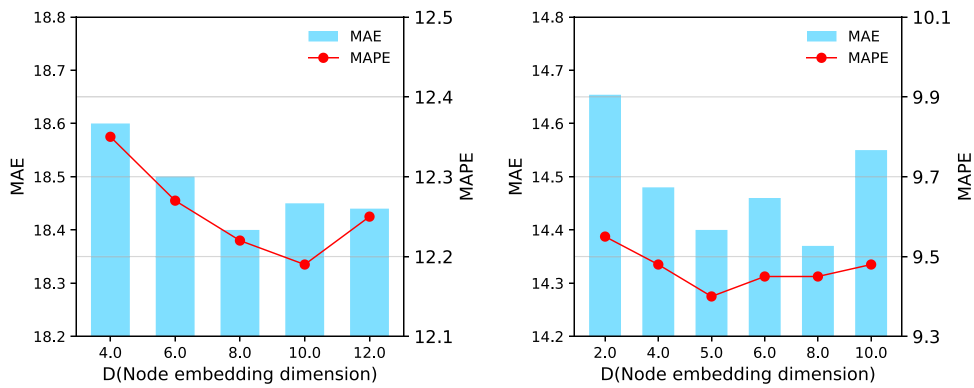
The influence of the two hyperparameters, block number K and the number of embedding dimensions D, is analyzed. Through experiments on the PEMSD4 and PEMSD8 datasets, we vary the value of K from 1 to 3 and vary D from 4 to 12 on PEMSD4 and from 2 to 10 on PEMSD8.

8.1. Sensitivity to K

The results for our tests with K are shown in Table 7. Although the single-layer DDGCRN without residual decomposition outperforms latest baselines, the result is not as good as the multi-layer DDGCRN with residual decomposition. This proves that decomposing the traffic signals is very effective. It can be observed that DDGCRN(2nd) and DDGCRN(3rd) exhibit comparable performance. This suggests that abnormal signals only account for a small proportion of the overall traffic signals. Modeling abnormal signals once can achieve sufficient performance improvement, so K is set as 2.

8.2. Sensitivity to D

Fig. 6 presents the results for the different embedding dimensions. Here, the model's performance is optimal only when D lays in a specific range. This indicates some kind of Goldilocks zone for the dimensionality of the embedding too small, and the spatial and temporal information will be hard to encode; too large, and overfitting might occur.



(a) Sensitivity to D in PEMSD4 (b) Sensitivity to D in PEMSD8

Fig. 6. Sensitivity analysis of parameter D.

9. Conclusion

This paper presents a novel decomposition dynamic graph model named DDGCRN for traffic forecasting. The proposed model utilizes spatio-temporal embedding and traffic signals to generate dynamic graphs. One module, DGCRM captures spatio-temporal correlations, while another, DGCRM, isolates abnormal signals from normal ones using a residual decomposition mechanism. Furthermore, a novel training strategy called segmented learning is developed to significantly reduce the amount of time and resources needed for initial training. DDGCRN consistently outperforms othe methods in extensive experiments on six datasets.

The dynamic graph generation approach proposed by DDGCRN achieves excellent performance, reduces reliance on prior knowledge, and further enhances the versatility of spatio-temporal graphs. However, despite its superior performance, there is still room to improve DDGCRN. For example, DDGCRN simply decomposes traffic signals into normal signals and abnormal signals for modeling. If multiple traffic patterns in traffic signals can be decomposed for modeling, it would lead to further improvements in both the performance and interpretability of the model. Therefore, our current focus is on finding a better approach to decompose the traffic signals.

Declaration of Competing Interest

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

Data availability

We provide a Github link in the paper.

Acknowledgments

This work is supported by a Grant from The National Natural Science Foundation of China (No. U21A20484); Science and Technology Program of Zhejiang Province under Grant No.2022C01016.

References

[1] S. Zamboni, Z.T. Kefato, S. Girdzijauskas, C. Norén, L. Dal Col, Pedestrian trajectory prediction with convolutional neural networks, Pattern Recognit. 121 (2022) 108252.
[2] A. Boukerche, Z. Hou, Object detection using deep learning methods in traffic scenarios, ACM Comput. Surv. (CSUR) 54 (2) (2021) 1–35.
[3] Z. Jiang, J. Gu, W. Fan, W. Liu, B. Zhu, Q-Learning approach to coordinated optimization of passenger inflow control with train skip-stopping on a urban rail transit line, Comput. Ind. Eng. 127 (2019) 1131–1142.
[4] L. Wu, M. Wang, D. Wu, J. Wu, Dynstgat: dynamic spatial-temporal graph attention network for traffic signal control, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, in: CIKM '21, 2021, pp. 2150–2159.
[5] S. Xue, J. Lu, J. Wu, G. Zhang, L. Xiong, Multi-instance graphical transfer clustering for traffic data learning, in: 2016 International Joint Conference on Neural Networks (IJCNN), 2016, pp. 4390–4395.
[6] J. Ye, L. Sun, B. Du, Y. Fu, H. Xiong, Coupled layer-wise graph convolution for transportation demand prediction, in: Proceedings of the AAAI conference on artificial intelligence, volume 35, 2021, pp. 4617–4625.
[7] K.C. Roy, S. Hasan, A. Culotta, N. Eluru, Predicting traffic demand during hurricane evacuation using real-time data from transportation systems and social media, Transp. Res. Part C: Emerg. Technol. 131 (2021) 103339.
[8] Y. Fan, J. Xu, R. Zhou, J. Li, K. Zheng, L. Chen, C. Liu, Metaer-tte: an adaptive meta-learning model for en route travel time estimation, in: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, volume 23, 2022, pp. 2023–2029.
[9] L. Bai, L. Yao, C. Li, X. Wang, C. Wang, Adaptive graph convolutional recurrent network for traffic forecasting, Adv. Neural. Inf. Process. Syst. 33 (2020) 17804–17815.
[10] S. Cao, L. Wu, J. Wu, D. Wu, Q. Li, A spatio-temporal sequence-to-sequence network for traffic flow prediction, Inf. Sci. (Ny) 610 (2022) 185–203.
[11] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: data-driven traffic forecasting, arXiv preprint arXiv:1707.01926 (2017).
[12] Z. Wu, S. Pan, G. Long, J. Jiang, C. Zhang, Graph wavenet for deep spatial-temporal graph modeling, arXiv preprint arXiv:1906.00121 (2019).
[13] Z. Lin, J. Feng, Z. Lu, Y. Li, D. Jin, Deepstn+: context-aware spatial-temporal neural network for crowd flow prediction in metropolis, in: Proceedings of the AAAI conference on artificial intelligence, volume 33, 2019, pp. 1020–1027.
[14] J. Zhang, Y. Zheng, D. Qi, Deep spatio-temporal residual networks for citywide crowd flows prediction, Thirty-first AAAI conference on artificial intelligence, 2017.
[15] H. Yao, X. Tang, H. Wei, G. Zheng, Z. Li, Revisiting spatial-temporal similarity: a deep learning framework for traffic prediction, in: Proceedings of the AAAI conference on artificial intelligence, volume 33, 2019, pp. 5668–5675.
[16] F. Li, J. Feng, H. Yan, G. Jin, F. Yang, F. Sun, D. Jin, Y. Li, Dynamic graph convolutional recurrent network for traffic prediction: benchmark and solution, ACM Trans. Knowl. Discov. Data (TKDD) (2021).
[17] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, in: Proceedings of the AAAI conference on artificial intelligence, volume 33, 2019, pp. 922–929.
[18] S. Wang, Y. Li, J. Zhang, Q. Meng, L. Meng, F. Gao, Pm2. 5-gnn: a domain knowledge enhanced graph neural network for pm2. 5 forecasting, in: Proceedings of the 28th international conference on advances in geographic information systems, 2020, pp. 163–166.
[19] N. Rathore, P. Rathore, A. Basak, S.H. Nistala, V. Runkana, Multi scale graph wavenet for wind speed forecasting, in: 2021 IEEE International Conference on Big Data (Big Data), IEEE, 2021, pp. 4047–4053.
[20] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, C. 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, 2020, pp. 753–763.
[21] Y. Chen, I. Segovia, Y.R. Gel, Z-gcnets: time zigzags at graph convolutional networks for time series forecasting, in: International Conference on Machine Learning, PMLR, 2021, pp. 1684–1694.
[22] J. Choi, H. Choi, J. Hwang, N. Park, Graph neural controlled differential equations for traffic forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, 2022, pp. 6367–6374.
[23] B. Yu, H. Yin, Z. Zhu, Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting, arXiv preprint arXiv:1709.04875 (2017).

[24] C. Zheng, X. Fan, C. Wang, J. Qi, Gman: a graph multi-attention network for traffic prediction, in: Proceedings of the AAAI conference on artificial intelligence, volume 34, 2020, pp. 1234–1241.
[25] P. Trirat, J.-G. Lee, Df-tar: a deep fusion network for citywide traffic accident risk prediction with dangerous driving behavior, in: Proceedings of the Web Conference 2021, 2021, pp. 1146–1156.
[26] D. Sun, S. Xue, H. Wu, J. Wu, A data stream cleaning system using edge intelligence for smart city industrial environments, IEEE Trans. Ind. Inf. 18 (2) (2022) 1165–1174.
[27] Z. Xia, S. Xue, J. Wu, Y. Chen, J. Chen, L. Wu, Deep reinforcement learning for smart city communication networks, IEEE Trans. Ind. Inf. 17 (6) (2021) 4188–4196.
[28] J.D. Hamilton, Time Series Analysis, Princeton university press, 2020.
[29] Y.-S. Jeong, Y.-J. Byon, M.M. Castro-Neto, S.M. Easa, Supervised weighting-online learning algorithm for short-term traffic flow prediction, IEEE Trans. Intell. Transp. Syst. 14 (4) (2013) 1700–1707.
[30] J. Van Lint, C. Van Hinsbergen, Short-term traffic and travel time prediction models, Artif. Intell. Appl. Crit. Transp. Iss. 22 (1) (2012) 22–41.
[31] W.K.A. Wan Ahmad, S. Ahmad, Arima model and exponential smoothing method: a comparison, in: AIP Conference Proceedings, volume 1522, American Institute of Physics, 2013, pp. 1312–1321.
[32] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, Adv. Neural Inf. Process. Syst. 27 (2014).
[33] X. Zhang, C. Huang, Y. Xu, L. Xia, P. Dai, L. Bo, J. Zhang, Y. Zheng, Traffic flow forecasting with spatial-temporal graph diffusion network, in: Proceedings of the AAAI conference on artificial intelligence, volume 35, 2021, pp. 15008–15015.
[34] S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, P. Li, Dstagnn: dynamic spatial-temporal aware graph neural network for traffic flow forecasting, in: International Conference on Machine Learning, PMLR, 2022, pp. 11906–11917.
[35] H. Peng, B. Du, M. Liu, M. Liu, S. Ji, S. Wang, X. Zhang, L. He, Dynamic graph convolutional network for long-term traffic flow prediction with reinforcement learning, Inf. Sci. (Ny) 578 (2021) 401–416.
[36] M. Welling, T.N. Kipf, Semi-supervised classification with graph convolutional networks, in: J. International Conference on Learning Representations (ICLR 2017), 2016.
[37] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).
[38] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, Z. Jia, Freeway performance measurement system: mining loop detector data, Transp. Res. Rec. 1748 (1) (2001) 96–102.
[39] B.M. Williams, L.A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal arima process: theoretical basis and empirical results, J. Transp. Eng. 129 (6) (2003) 664–672.
[40] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122(2015).
[41] S. Huang, D. Wang, X. Wu, A. Tang, Dsanet: dual self-attention network for multivariate time series forecasting, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 2129–2132.
[42] L. Bai, L. Yao, S.S. Kanhere, X. Wang, Q.Z. Sheng, Stg2seq: spatial-temporal graph to sequence model for multi-step passenger demand forecasting
[43] R. Huang, C. Huang, Y. Liu, G. Dai, W. Kong, Lsgcn: long short-term traffic prediction with graph convolutional networks, in: IJCAI, 2020, pp. 2355–2361.
[44] C. Song, Y. Lin, S. Guo, H. 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, 2020, pp. 914–921.
[45] M. Li, Z. Zhu, Spatial-temporal fusion graph neural networks for traffic flow forecasting, in: Proceedings of the AAAI conference on artificial intelligence, volume 35, 2021, pp. 4189–4196.
[46] Z. Fang, Q. Long, G. Song, K. Xie, Spatial-temporal graph ode networks for traffic flow forecasting, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 364–373.
[47] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, J. Mach. Learn. Res. 9 (11) (2008).

Wenchao Weng received the B.S. degree in Information and Computing Science from Zhejiang Wanli University in 2019. He is currently pursuing the M.S. degree in computer technology at Hangzhou Dianzi University, Hangzhou, China. His research interests include time series data analytics, graph Neural Network and traffic prediction.

Jin Fan received her B.Sc. degree from Xi'an Jiaotong University, China, and M.Sc. and Ph.D. degrees from Loughborough University, U.K. She is currently an Associate Professor with the Department of Computer Science and Technology, Hangzhou Dianzi University, China. She has published over 20 technical papers in international journals and conferences in the past five years. Her research interests lie in the general areas of data analytics, wireless sensor networks, mobile computing and related aspects.

Huifeng Wu (Member, IEEE) holds a Ph.D. degree in computer science and technology from Zhejiang University, Hangzhou, China, earned in 2006. He is currently a Professor with the Institute of Intelligent and Software Technology, Hangzhou Dianzi University, Hangzhou. His current research interests include the industrial internet of things, software development methods and tools, software architecture, embedded system, and intelligent control and automation.

Yujie Hu received the B.S. degree degree in Communication Engineering from Soochow University in 2021. He is currently pursuing the MS degree in electronic Information Technology at Hangzhou Dianzi University in China. His research interest covers autonomous driving, graph neural networks and topology.

Hao Tian received his Ph.D. degree in civil engineering from Tongji University, China. He was a postdoctoral fellow at ATLSS National Engineering Research Center of Lehigh University, USA. He is currently the vice president of Zhejiang Scientific Research Institute of Transport, Young Science and Technology talents of the Ministry of Transport, and 151 Talent of Zhejiang Province. His research activity concerns structural health monitoring data analysis and safety assessment of long-span bridges, key technologies for operation and maintenance of cross-sea bridges, structural reinforcement, and performance optimization. By 2021, He has won the third prize of the Zhejiang Provincial Science and Technology Progress Award and the second prize of the science and technology progress award in China Highway and Transportation Society. He has published 50+ refereed papers in SCI and EI magazines.

Fu Zhu received his B.S. degree in Software Engineering from Zhejiang University of Science and Technology, Hangzhou, China, in 2022. He is currently pursuing the M.S. degree in computer technology at Hangzhou Dianzi University,Hangzhou. His research interests include deep learning,time series analysis and traffic prediction.

Jia Wu (SM'21) received his Ph.D. degree in computer science from the University of Technology Sydney, Australia. He is currently an ARC DECRA Fellow with the Department of Computing, Macquarie University, Sydney, Australia. His current research interests include data mining and machine learning. Since 2009, he has published 100+ refereed journal and conference papers in titles such as IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), IEEE Transactions on Knowledge and Data Engineering (TKDE), ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), The Web Conference (WWW), and Neural Information Processing Systems (NeurIPS). Dr Wu was the recipient of the SDM'18 Best Paper Award in the Data Science Track, the IJCNN'17 Best Student Paper Award, and the ICDM'14 Best Paper Candidate Award. He is the Associate Editor of ACM Transactions on Knowledge Discovery from Data (TKDD) and Neural Networks (NN).