# ECAI 2023

26th European Conference on Artificial Intelligence

September 30 – October 4, 2023, Kraków, Poland

Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023)

**Edited by**
Kobi Gal
Ann Nowé
Grzegorz J. Nalepa
Roy Fairstein
Roxana Rădulescu

IOS Press

*ECAI 2023*
*K. Gal et al. (Eds.)*


*doi:10.3233/FAIA230446*

# Dynamic Graph Convolutional Network with Attention Fusion for Traffic Flow Prediction

**Xunlian Luo[a], Chunjiang Zhu[b], Detian Zhang[a;*] and Qing Li[c]**

[a]Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, China.
[b]Department of Computer Science, University of North Carolina at Greensboro, USA.
[c]Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China.
20215227114@stu.suda.edu.cn, chunjiang.zhu@uncg.edu, detian@suda.edu.cn, qing-prof.li@polyu.edu.hk

**Abstract.** Accurate and real-time traffic state prediction is of great practical importance for urban traffic control and web mapping services. With the support of massive data, deep learning methods have shown their powerful capability in capturing the complex spatial-temporal patterns of traffic networks. However, existing approaches use pre-defined graphs and a simple set of spatial-temporal components, making it difficult to model multi-scale spatial-temporal dependencies. In this paper, we propose a novel dynamic graph convolution network with attention fusion to tackle this gap. The method first enhances the interaction of temporal feature dimensions, and then it combines a dynamic graph learner with GRU to jointly model synchronous spatial-temporal correlations. We also incorporate spatial-temporal attention modules to effectively capture long-range, multifaceted domain spatial-temporal patterns. We conduct extensive experiments in four real-world traffic datasets to demonstrate that our method surpasses state-of-the-art performance compared to 18 baseline methods.

## 1 Introduction

In urban computing, popular web mappings services such as Google Maps, and Bing Maps heavily rely on accurate traffic flow prediction as a backend for their frontend applications such as personalized route planning and closest restaurant/gas station recommendation. Similarly, departments of transportation in next-generation smart cities often need to allocate traffic resources and optimize traffic control plans, e.g., for an exposition, by blocking a minimum number of streets while avoiding serious traffic congestion. The plan needs to be generated in advance and thus requires precise traffic flow estimation. Recently, we have witnessed the bloom of online ride-hailing/sharing services, e.g., Uber, Lyft, and Didi. They often provide functions to request a ride on the Web directly with no need to download apps. These Web-based services, again, need a powerful traffic prediction engine. The traffic flow prediction model analyzes large amounts of historical traffic data to estimate future traffic conditions based on statistical or data-driven methods [23], providing a basis for decision-making for these applications. However, traffic flow is affected by complex spatial and temporal dependencies, which make accurate real-time traffic forecasting extremely challenging.

As shown in Fig. 1, traffic prediction differs from traditional time series in that it has significant spatial and temporal correlation and is also susceptible to other external factors [11]. The variation of traffic in time has proximity and trend. It is also influenced by the travel routine of urban residents, which shows a certain periodicity over a long period of time. Being connected by a non-Euclidean spatial road network, the traffic volume at the observation point is subject to the propagation of vehicles from neighboring roads upstream and downstream, exhibiting mobility and transience. Besides, the intervention of some external factors (weather, graduation ceremony, traffic accidents, etc.) also can cause drastic fluctuations in traffic flow. Additionally, the spatial and temporal correlations are typically multi-scale, which can be roughly divided into global and local in space and long-term and short-term in time.

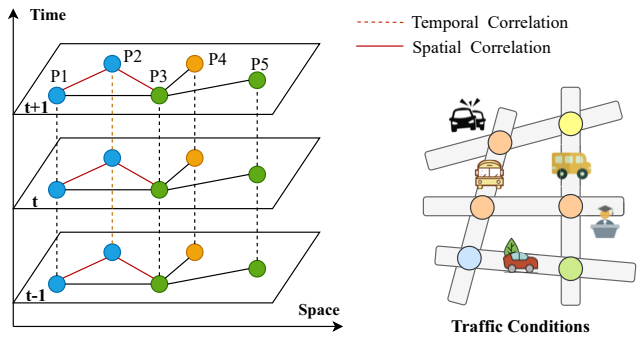


**Figure 1**: Complex spatial-temporal correlations.

Extensive research has been carried out to address these challenges. Statistical models (e.g., VAR [20], ARIMA [14]) and machine learning methods including SVR [29] and ANN [24], treat traffic sequences as independent data streams, considering only the temporal variability of the data and ignoring spatial correlation. The rapid development of deep learning techniques brings traffic prediction research to a new stage. In particular, CNN [2], RNNs (its variants LSTM, GRU) [5], and Transformer with self-attention [25] have been successful in modeling sequence tasks. A novel attempt is to use convolution neural networks to model the implicit spatial relationships of urban regions as they do for images or videos [38]. Later studies identify the advantages of graph neural networks over CNNs in extracting structure information of irregular topologies [30]. Inspired by the spatial-temporal characteristics of traffic data, an intuition naturally develops to combine graph convolution networks with sequence models to jointly model spatial-temporal dependencies.

* Corresponding Author. Email: detian@suda.edu.cn.

Spatial-temporal graph neural networks provide a feasible solution for traffic prediction. Some models such as DCRNN [17] combines diffusion convolution with GRU-based encoder-decoder architecture to capture the directionality of traffic propagation. STSGCN [21] proposes a local spatial-temporal graph to represent the transfer of information in the temporal and spatial directions, and adopts residual-connected GCN modules to model spatial-temporal heterogeneity. However, they ignore the fact that pre-defined graphs cannot fully reflect the reality of traffic connections. Although GWNET [32] and MTGNN [31] improve their practices with adaptive graph learning, the separated spatial-temporal components are unable to capture synchronized correlations. GMAN [39] based on spatial-temporal attention fusion shows promising results for long-term forecasting but performs poorly in the short term. STG-NCDE [4] and DSTAGNN [15] achieve good prediction performance, but bring large computational consumption and system overhead. These works have contributed to a tremendous improvement in performance from multiple perspectives. However, we observe that they are not combined in a cohesive way to address comprehensive features and their correlations, and rarely balance method performance and complexity.

To address the shortcomings of existing methods, in this paper, we propose a Spatial-Temporal **A**ttention **F**usion **D**ynamic **G**raph **C**onvolution **N**etwork (**AFDGCN**) for traffic flow prediction. Specifically, We employ feature augmentation, dynamic graph convolution recurrent networks, and multiple attention fusion to capture complex traffic patterns from multi-scale spatial-temporal aspects. In summary, the main contributions of this paper are as follow:

- A novel and efficient spatial-temporal framework is proposed for traffic prediction. The traffic features are first augmented to improve the information interaction of the data on the feature channel and the temporal axis, rather than performing a basic channel ascending dimension.
- We propose a dynamic graph learner for capturing hidden spatial dependencies and combine adaptive graph convolution with gated recurrent units to build a dynamic graph convolution recurrent network for capturing local spatial-temporal correlations.
- We consolidate temporal attention and spatial attention to improve the model's ability that identifies changes in the spatial field of view and long-range time dependence. This indeed reduces the error of overall prediction in complex scenes and enhances the robustness of the model.
- We conduct extensive experiments in four benchmark datasets, and evaluate the performance compared with 18 baseline methods. The results show that our proposed method outperforms all other methods in three standard evaluation metrics with low computation costs.

## 2 Related Work

### 2.1 Graph Convolution Networks

Graph convolution networks process data as a set of graph structures consisting of nodes and edges [9], and have a wide range of applications in social network analysis, recommender systems and molecular biology. Its core idea is to aggregate the features of a node with those of its neighboring nodes to generate a new feature representation of the node [30]. Generally, there are two mainstream graph convolution networks, namely, the spectral-based methods and the spatial-based methods. Chebyshev graph convolution network (ChebNet) is a representative spectral method that transforms the graph signal into the frequency domain for filtering to achieve noise reduction and feature extraction [6]. Specifically, by computing the Chebyshev polynomial $T_k(x)$ to approximate the filter, a Laplacian matrix $\widetilde{L}$ can be obtained as follows:

$$\widetilde{L} = \frac{2}{\lambda_{max}} L - I_N = \sum_{k=0}^{K-1} \theta_k T_k(\widetilde{L}) \tag{1}$$

where $L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U \Lambda U^T$ is the normalized graph Laplacian matrix, $U$ is the eigenvalue matrix of $L$.
The ChebNet is expressed as follows:

$$\Theta \star_{\mathcal{G}} X = \Theta(L)X = \sigma\left(\sum_{k=0}^{K-1} \theta_k T_k(\widetilde{L}) x_i\right) \tag{2}$$

where $X \in \mathbb{R}^{N \times d}$ is the feature matrix of all nodes in the graph, $A \in \mathbb{R}^{N \times N}$ is the adjacency matrix of the graph and $k$ is the order of the ChebNet.

In traffic prediction, graph convolutional networks (GCNs) are generalized to high-dimensional GCNs by approximation of first-order Chebyshev polynomials [13] as follows:

$$X^{(l)} = (I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}) X^{(l-1)} W + b. \tag{3}$$

where $D$ is the degree matrix of $A$, $W$ and $b$ are learnable weights and biases, and $X^{(l)}$ denotes the hidden representation of the $l$-layer.

### 2.2 Deep Learning for Traffic Prediction

A basic assumption behind spatial-temporal graphs for traffic forecasting is that the future state of a node depends on its historical knowledge and the information of its neighbors [32]. It is natural to combine graph convolution networks with sequence models to construct spatial-temporal framework for modeling the spatial-temporal dependence of traffic data. The models can be classified as CNN-Based, RNN-Based and Attention-Based according to their temporal components [10]. The CNN-based approaches use dilation convolution to expand the receptive field of the model with high computational efficiency, but the fixed implicit time step sacrifices some flexibility. RNNs provide a universal architecture of encoder-decoder. Many works [17, 16] replace the fully connected layer of GRU with GCN to achieve synchronous spatial-temporal modeling. Compared to the previous two, the attention-based methods [39, 7] are relatively flexible and can learn to go further back in time patterns. The ongoing advancement of these efforts has made deep learning the primary method for spatial-temporal data mining. Meanwhile, the study of graph quality becomes a more general task. This has a direct impact on whether the model can effectively capture spatial dependencies. Adaptive graphs [1] and discrete sampling graphs [37] in many works have been shown to model information propagation in space better than pre-defined graphs. Compared to analyzing temporal correlations, it is much more challenging to model spatial relations in traffic with insufficient prior knowledge.

## 3 Problem Statement

Traffic forecasting refers to the use of historically observed traffic data to predict the traffic state in a future period based on road network knowledge. The traffic sensor distribution is represented as a graph $\mathcal{G} = (V, E, A)$. $V$ is the set of $N = |V|$ nodes in the spatial topology and $E$ is the set of edges connected by node pairs. $A \in \mathbb{R}^{N \times N}$ is the adjacency matrix of the graph $\mathcal{G}$, indicating the

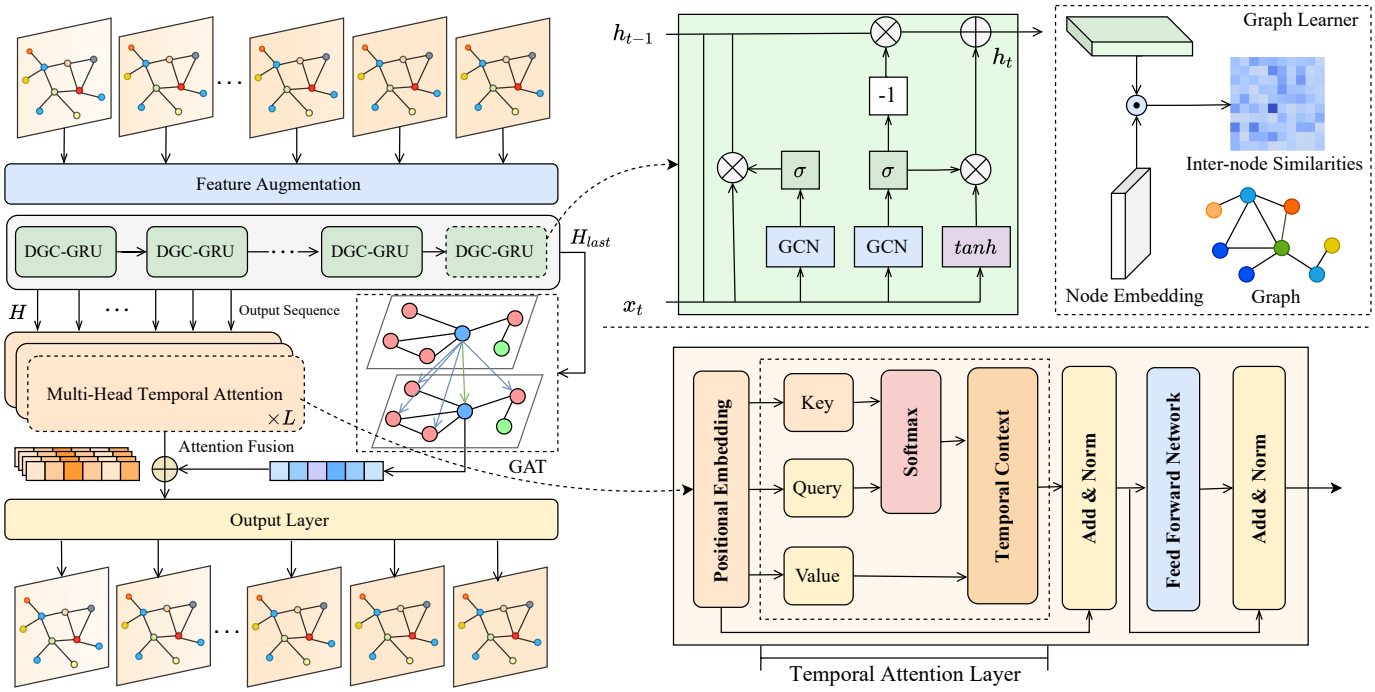


**Figure 2**: The model overview of dynamic graph convolution network with spatial-temporal attention fusion.

proximity of connections or distances based on the Gaussian kernel function. Denote the traffic flow observed on $\mathcal{G}$ at time step $t$ as a graph signal matrix $X_t \in \mathbb{R}^{N\times C}$, where $C$ is the feature channel (e.g. flow, speed, demand) of each node.

Similarly, $X_{t-T:t-1} \in \mathbb{R}^{T\times N\times C}$ denotes the traffic statistics of the full road network at time intervals of $T$. The aim of traffic forecasting is to learn a function $f$ that is able to forecast $Q$ future signal tensor given $P$ historical signal tensor and the graph $\mathcal{G}$:

$$[X_{t-P-1}, \cdots, X_t; \mathcal{G}] \xrightarrow[\Theta]{f} [\widehat{X}_{t+1}, \cdots, \widehat{X}_{t+Q}] \tag{4}$$

where $\Theta$ denotes the model parameters to be optimized.

## 4 Proposed Model

We propose a novel spatial-temporal graph neural network called AFDGCN. It consists of four main modules, namely feature augmentation layer, dynamic graph convolutional recurrent network, multi-head temporal attention module, and graph attention module. The structure of the model is illustrated in Fig. 2.

### 4.1 Feature Augmentation Layer

To gain a comprehensive understanding of the intrinsic characteristics of time series data, we develop a feature augmentation layer that is inspired by channel-wise attention [28]. The layer is based on a "squeeze-excitation" structure [18] that enhances the information interaction between the channels and the temporal dimension. It improves the expressiveness of the feature map by automatically calibrating the importance of the feature space.

Specifically, the feature augmentation layer consists of two substructures in series, namely the channel calibration block and the temporal calibration block. The channel block is a fully connected network with two layers, which first performs a squeezing operation on the feature map $X \in \mathbb{R}^{T\times N\times D}$. This helps the model recalibrate the features with a low-dimensional distribution. It then analyzes the influence of each channel through a single activation process and a self-threshold mechanism. The channel-wise computation is formalized as follows:

$$s^{(c)} = \sigma(W_2\delta(W_1X^{(c)} + b_1) + b_2), \tag{5}$$

$$X_1^{(c)} = X^{(c)} \circ s^{(c)}. \tag{6}$$

where $W_1, W_2$ and $b_1, b_2$ are the weight parameters and bias of the fully connected layer. $\circ$ is the Hamada product. $\sigma$ and $\delta$ are the Sigmoid and ReLU activation functions, respectively.

Similarly, to focus on the structural information (significant time points in $P$) of the time series, the temporal calibration block utilizes a temporal convolution layer for feature extraction. After channel calculation and dimensional exchange, the feature map $X_1$ served as the input to the temporal-wise block. That is:

$$s^{(t)} = \sigma(\Theta_{1,k} \star \delta(\Theta_{1,k} \star X_1^{(t)}), \tag{7}$$

$$X_2^{(t)} = X_1^{(t)} \circ s^{(t)}. \tag{8}$$

where $\Theta_{1,k}$ is the $1 \times k$ convolution kernel. After preprocessing by feature augmentation layer, the reweighted feature maps are used as the output to feed in the training of the subsequent structures.

### 4.2 Dynamic Graph Convolutional Recurrent Network

The performance of traditional GCNs relies on a pre-defined graph structure. However, the connections built during the study based on distance proximity are often incomplete and biased [32, 31]. And the nodes' own factors (e.g., POI, road structure, and road type) are difficult to represent [8]. These make general GCNs unable to capture spatial dependencies effectively. In view of these problems, we propose a dynamic graph learner for mining the potential spatial properties of traffic data. It generates the similarity matrix by initializing the node embeddings and the GCN takes into account the unique

patterns of each node during feature aggregation. The parameters are dynamically updated during the training stage to adapt to the traffic data in an end-to-end manner.

All nodes are predefined with the embedding vector $E_G \in \mathbb{R}^{N\times d}$, where $d$ denotes the node embedding matrix, $d \ll N$. Then, we infer the implicit correlation between node pairs by $E_G \cdot E_G^T \in \mathbb{R}^{N\times N}$. The normalized adjacency matrix is defined as:

$$\widetilde{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = softmax(ReLU(E_G \cdot E_G^T)), \tag{9}$$

where $E_G^{(i)} \in \mathbb{R}^d$ is the node embedding vector of $v_i$. The ReLU activation function is used for matrix sparsification and to eliminate the effect of negative values.

We observe that the weight and bias used by the traditional GCN for feature linear transformation are shared by all nodes. This ignores the differences in the traffic patterns of the nodes themselves. Inspired by the matrix decomposition, we perform matrix multiplication of $W_G \in R^{d\times C\times F}$ ($b_G \in R^{d\times F}$) with the node embedding $E_G \in R^{N\times d}$ to generate new model parameters $W = E_G W_G$ ($b = E_G b_G$, respectively). In this way, the dynamic generation of graph convolution $\Theta_x \star \mathcal{G}$ is expressed as:

$$Z = (I_N + softmax(ReLU(E_G \cdot E_G^T)))X E_G W_G + E_G b_G. \tag{10}$$

To capture synchronous temporal and spatial dependencies, we extend the fully connected operation in GRU with GCN to obtain the Dynamic Graph Convolutional Recurrent Network (DGCGRU). Due to the superior performance of GRU in handling sequential tasks, we embed graph convolution operations in the original GRU structure to capture sequential information while merging spatial relationships. Structurally, in addition to the present data input, each time step receives a hidden representation from the previous time step [17], which is used to control the memorization and forgetting of information. Here, the GRU operation is applied to each node in the graph and the parameters are shared with each other.

Specifically, given the previously hidden representation $H_{t-1}$ and the input data $X_{:,t}$ at time step $t$, we represent the computation of a single-step gated recurrent unit as:

$$\begin{aligned} z_t &= \sigma\left(\Theta_z \star \mathcal{G}[X_{:,t}, H_{t-1}] + b_z\right) \\ r_t &= \sigma\left(\Theta_r \star \mathcal{G}[X_{:,t}, H_{t-1}] + b_r\right) \\ \hat{H}_t &= tanh\left(\Theta_h \star \mathcal{G}[X_{:,t}, (r \odot H_{t-1})] + b_h\right) \\ H_t &= z \odot H_{t-1} + (1 - z) \odot \hat{H}_t, \end{aligned} \tag{11}$$

where $b_z, b_r$ and $b_h$ are learnable parameters of the recurrent neural network and $H_t$ is the hidden output at time $t$.

The combination of dynamic graph convolution and GRU enhances the model's ability to handle fine-grained spatial-temporal patterns. This dynamically generated graph can be freed from the constraints of pre-defined graphs because it adaptively builds connection relationships of node pairs using training data. However, GRU does not rigidly memorize all fixed-length sequences, but selectively stores information of historical time steps through hidden states due to the forgetting characteristic of gating. Although it can effectively capture local temporal information from traffic sequence data, it is insensitive to long-distance temporal relationships.

### 4.3 Multi-head Temporal Attention Module

A multi-head temporal attention module is used to extract the global context information of the sequence [27, 34]. This module consists of a positional embedding, a multi-head temporal attention layer, and a residual network. The temporal attention enables the model to observe longer-term temporal trends and focus on highly correlated information to compensate for the deficiencies of GRU in modeling long-range temporal dependence.

The hidden layer sequence $\{H_1[i,:], H_2[i,:], \cdots, H_T[i,:]\}$ from DGCGRU is fed to the temporal attention module. Given that the computation of the multi-head attention mechanism [25] ignores the relative positions of the sequences, we add a position encoding to the feature data at each time step. That is:

$$\hat{H}_t[i,:] = H_t[i,:] + e_t, \text{where } H_t \in \mathbb{R}^{N\times D} \tag{12}$$

where the position token $e_t$ is defined as:

$$e_t = \begin{cases} \sin(t/10000^{2i/d_{model}}), & \text{if } t = 0, 2, 4, \ldots \\ \cos(t/10000^{2i/d_{model}}), & \text{otherwise} \end{cases} \tag{13}$$

The embedded features $\hat{H} \in \mathbb{R}^{T\times N\times D}$ are first projected into a high-dimensional latent subspace to generate the Query, Key, and Value matrix as shown in Eq. (14). The mappings are realized with the feed-forward neural networks. Then, Q and $K^T$ are matrix products and normalized to obtain the attention distribution of each time step. Finally, the attention matrix is superimposed on $V$ to generate an implicit representation with temporal context as Eq. (15).

$$Q = \hat{H} W_q^T, K = \hat{H} W_k^T, V = \hat{H} W_v^T \tag{14}$$

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V, \tag{15}$$

where $W_q^T, W_k^T$, and $W_v^T$ are the projection matrix to be learned, $\sqrt{d_k}$ is the weight scaled factor, and the softmax function is used to normalize the weight scores.

To improve the perception level of temporal semantics in different subspaces, we adopt a multi-head temporal attention module to enrich the representation of information. Multiple sets of self-attentions act independently on the sequence. Their results are concatenated and then linearly transformed to obtain the outputs $H_{Attn}$. The mathematical formula is as follows:

$$\begin{aligned} &MultiHead(Q, K, V) = Concat(head_1, head_2, \cdots, head_h) W^O, \\ &head_i = Attention(\hat{H} W_i^Q, \hat{H} W_i^K, \hat{H} W_i^V). \end{aligned} \tag{16}$$

where $W_i^Q$, $W_i^K$, and $W_i^V \in \mathbb{R}^{d_{model}\times d_k}$ are all the projection matrices for the linear transformation. $D = h \times d_{model}$, $h$ is the number of heads.

Finally, $H_{Attn}$ is passed into the residual network, with each sublayer followed by a Feed-Forward Network and a Layer Normalization. We define the representation of the temporal attention module output as $H_S \in \mathbb{R}^{T\times N\times D}$.

### 4.4 Graph Attention Module

Adaptive graph learner and recurrent graph convolution layer can capture spatial heterogeneity and synchronous spatial-temporal correlations. However, the spatial information of the traffic context is often dynamically evolving [19]. For example, a major event (e.g., concerts, traffic accidents) at a location provokes a sudden increase or decrease in vehicles within the neighboring area. To respond to complex and variable spatial patterns, our model applies a graph attention module [26]. It dynamically assigns different weights based on the feature similarity of the target node and adjacent nodes.

The input to the graph attention module is the set of all node hidden vectors, denoted as $H_{last} = \{h_1, h_2, \cdots, h_N\}$, where $h_i \in \mathbb{R}^D$. Suppose the feature vectors of $v_i$ and $v_j$ be $h_i$ and $h_j$, and $N_i$ be the set of neighbors to $v_i$. The equation to calculate the attention scores between node pairs is as follows:

$$e_{ij} = a(Wh_i, Wh_j), j \in N_i, \tag{17}$$

where $(\cdot, \cdot)$ is the concatenation operation, $W \in \mathbb{R}^{F \times D}$ is a learnable linear matrix, $a : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}$ as same as $W$ maps the combined parameter matrix into a scalar.

We use the nonlinear activation function LeakyReLU to eliminate minor dependencies. Then the attention scores of $v_i$'s all the adjacent nodes are normalized by the softmax. Formally, the formula is calculated as below:

$$\alpha_{ij} = softmax(e_{ij}) = \frac{exp(LeakyReLU(e_{ij}))}{\sum_{k \in N_i} exp(LeakyReLU(e_{ik}))}, \tag{18}$$

Message aggregation of attention is applied to each node to obtain its output representation. That is:

$$h_i' = \sigma(\sum_{j \in N_i} \alpha_{ij} W h_j). \tag{19}$$

where $\alpha_{ij}$ is the normalized attention score. Generally, we also dropout a certain percentage of it. Here we represent the above steps uniformly in matrix calculations as the following:

$$H_S = ELU\left((M \odot A)H_{last}W\right). \tag{20}$$

where $A$ is the adjacency matrix based on the distance relationship. The ELU is also an activation function. The elements in $M \in \mathbb{R}^{N \times N}$ are the dynamic attention factors. $H_S \in \mathbb{R}^{N \times D}$ is the graph embedding of the output in the graph attention module.

**The prediction layer** is a typical convolution layer that uses the hidden features of spatial-temporal attention fusion for multi-step prediction. Its input is a graph embedding $H_S$ and temporal tensor $H_T$, which is mapped to the output space after broadcast summation. Formally, it is as follows:

$$\widehat{X}_{t+1:t+Q} = Conv(H_T + H_S). \tag{21}$$

where $\hat{X} \in \mathbb{R}^{Q \times N \times D}$ is the final prediction result of the model.

## 5 Experiment

In this section, we first introduce the experimental settings, including the datasets used, baseline models, and evaluation metrics. We then provide a detailed analysis of the experimental results and visualizations. Finally, we present a series of studies, including module ablation and hyperparameter tuning, to illustrate the effects of model components and parameters on the overall results.

### *5.1 Experimental Settings*

**Dataset.** We use four traffic flow datasets (i.e., PeMSD3, PeMSD4, PeMSD7, and PeMSD8) collected by the California Department of Transportation sensors on highways [3]. They record traffic statuses such as flow, speed, and occupancy at 5-minute time intervals. The statistical details of the datasets are provided in Table 1.

The raw data are standardized using Z-Score [21] and then divided into the training set, validation set, and test set with a ratio of 6: 2: 2. The training set is disrupted before training and the validation set is used to control the early stopping of the training process. In the multi-step prediction, we set both the input sequence $P$ and the output sequence $Q$ to 12.

**Table 1**: The statistics of the tested real-world datasets.

| Dataset | Sensors | Edges | Samples | Time Range |
|---|---|---|---|---|
| PeMSD3 | 358 | 547 | 26,208 | Sept - Nov, 2018 |
| PeMSD4 | 307 | 340 | 16,992 | Jan - Feb, 2018 |
| PeMSD7 | 883 | 866 | 28,224 | May - Aug, 2017 |
| PeMSD8 | 170 | 296 | 17,856 | Jul - Aug, 2016 |

**Baseline Methods.** To evaluate the proposed model, we compare it with 18 baseline models, including traditional statistical, deep learning, and graph neural network models. These baselines serve as a benchmark to demonstrate the effectiveness of our approach. Below is a brief overview of the critical baseline:

- **VAR** [29]: Vector Auto-Regression assumes that the historical time series is stationary and forecasts by estimating the relationship between the time series and its lagged values.
- **ARIMA** [14]: Auto-regressive Integrated Moving Average with Kalman filter is a widely used statistical model for time series.
- **FC-LSTM** [22]: A recurrent neural network with fully connected LSTM hidden units.
- **STGCN** [36]: Spatial-Temporal Graph Convolutional Network, which com- bines graph convolution with 1D gated convolution.
- **DCRNN** [17]: Diffusion convolution recurrent neural network, which combines graph convolution networks with recurrent neural networks in an encoder- decoder architecture.
- **GWNET** [33]: Graph WaveNet introduces an adaptive adjacency matrix and combines diffuse graph convolution with TCN instead of 1D convolution.
- **AGCRN** [1]: Adaptive Graph Convolutional Recurrent Network, which augments traditional graph convolution with adaptive graph generation and node adaptive parameter learning, and is integrated into a recurrent neural network to capture more complex spatial-temporal correlations.
- **STG-NCDE** [4]: Spatial-Temporal Graph Neural Controlled Differential Equation, which designs two NCDEs for temporal processing and spatial processing and integrates them into a single framework.
- **DSTAGNN** [15]: Dynamic Spatial-Temporal Aware Graph Neural Network, which captures dynamic spatial and temporal dependencies between nodes and receptive field features through improved multi-headed attention and multi-scale gated convolution.

**Evaluation Metrics.** Throughout the experiments, we employ Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) [27] as evaluation metrics to compare the overall performance of the different models on the test dataset. A lower MAE, RMSE, and MAPE value indicates a better prediction performance.

**Parameters Setup.** Our task is to predict the traffic state in the next hour based on the historical traffic data of the previous hour. In our experiments, we set all the hidden dimensions to 64, the head of attention to 4, and the order of graph convolution to 2. We use the Adam [12] optimizer for training the model. We select SmoothL1Loss as the loss function. The batch size of the data is 64, the number of epochs is 300, and the initialized learning rate is 0.003. An early stopping strategy is used with a patience of 15 iterations on the validation dataset to prevent overfitting.

**Table 2**: Performance comparison of AFDGCN and other baseline models. AFDGCN achieves the best performance for all datasets.

| Model | PEMSD3 | | | PEMSD4 | | | PEMSD7 | | | PEMSD8 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MAE | RMSE | MAPE | MAE | RMSE | MAPE | MAE | RMSE | MAPE | MAE | RMSE | MAPE |
| HA | 31.58 | 52.39 | 33.78% | 38.03 | 59.24 | 27.88% | 45.12 | 65.64 | 24.51% | 34.86 | 59.24 | 27.88% |
| ARIMA | 35.41 | 47.59 | 33.78% | 33.73 | 48.80 | 24.18% | 38.17 | 59.27 | 19.46% | 31.09 | 44.32 | 22.73% |
| VAR | 23.65 | 38.26 | 24.51% | 24.54 | 38.61 | 17.24% | 50.22 | 75.63 | 32.22% | 19.19 | 29.81 | 13.10% |
| FC-LSTM | 21.33 | 35.11 | 23.33% | 26.77 | 40.65 | 18.23% | 29.98 | 45.94 | 13.20% | 23.09 | 35.17 | 14.99% |
| TCN | 19.32 | 33.55 | 19.93% | 23.22 | 37.26 | 15.59% | 32.72 | 42.23 | 14.26% | 22.72 | 35.79 | 14.03% |
| STGCN | 17.55 | 30.42 | 17.34% | 21.16 | 34.89 | 13.83% | 25.33 | 39.34 | 11.21% | 17.50 | 27.09 | 11.29% |
| DCRNN | 17.99 | 30.31 | 18.34% | 21.22 | 33.44 | 14.17% | 25.22 | 38.61 | 11.82% | 16.82 | 26.36 | 10.92% |
| GWNET | 19.12 | 32.77 | 18.89% | 24.89 | 39.66 | 17.29% | 26.39 | 41.50 | 11.97% | 18.28 | 30.05 | 12.15% |
| STG2Seq | 19.03 | 29.83 | 21.55% | 25.20 | 38.86 | 13.18% | 32.77 | 47.16 | 20.16% | 20.17 | 30.71 | 17.32% |
| LSGCN | 17.94 | 29.85 | 16.98% | 21.53 | 33.86 | 13.18% | 27.31 | 41.46 | 11.98% | 17.73 | 26.76 | 11.20% |
| ASTGCN | 17.34 | 29.56 | 17.21% | 22.93 | 35.22 | 16.56% | 24.01 | 37.87 | 10.73% | 18.25 | 28.06 | 11.64% |
| STSGCN | 17.48 | 29.21 | 16.78% | 21.19 | 33.65 | 13.90% | 24.26 | 39.03 | 10.21% | 17.13 | 26.80 | 10.96% |
| STFGNN | 16.77 | 28.34 | 16.30% | 20.48 | 32.51 | 16.77% | 23.46 | 36.60 | 9.21% | 16.94 | 26.25 | 10.60% |
| STGODE | 16.50 | 27.84 | 16.69% | 20.84 | 32.82 | 13.77% | 22.59 | 37.54 | 10.14% | 16.81 | 25.97 | 10.62% |
| AGCRN | 15.98 | 28.25 | 15.23% | 19.83 | 32.26 | 12.97% | 22.37 | 36.55 | 9.12% | 15.95 | 25.22 | 10.09% |
| Z-GCNETs | 16.64 | 28.15 | 16.39% | 19.50 | 31.61 | 12.78% | 21.77 | 35.17 | 9.25% | 15.67 | 25.11 | 10.01% |
| STG-NCDE | 15.57 | 27.09 | 15.06% | 19.21 | 31.09 | 12.76% | 20.53 | 33.84 | 8.80% | 15.45 | 24.81 | 9.92% |
| DSTAGNN | 15.57 | 27.21 | 14.68% | 19.30 | 31.46 | 12.70% | 21.67 | 34.51 | 9.01% | 15.67 | 24.77 | 9.94% |
| **AFDGCN** | **14.97** | **25.81** | **14.18%** | **19.09** | **31.01** | **12.62%** | **20.22** | **33.80** | **8.52%** | **15.02** | **24.37** | **9.68%** |

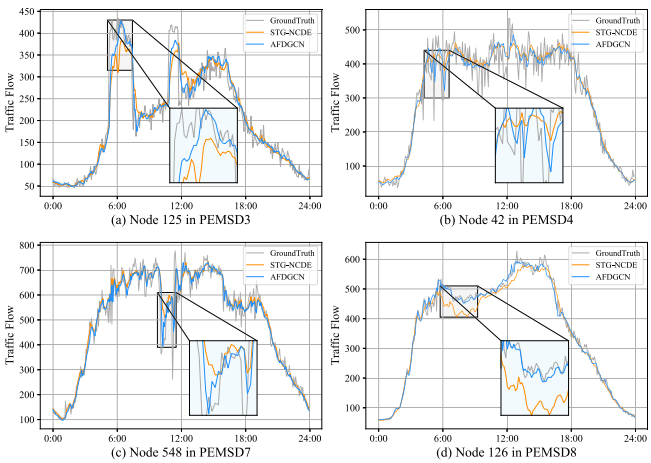


**Figure 3**: Traffic forecasting within one day in different datasets.

## 5.2 *Experimental Results*

Table 2 presents the evaluation results of AFDGCN and major baseline methods on the four tested datasets. The bold values and the underlined values are the best and second-best prediction performances, respectively. Our method (AFDGCN) always achieves the best performance and is in bold. Compared with the best baseline (underlined), the performance of AFDGCN in four datasets (PeMSD3, PeMSD4, PeMSD7, and PeMSD8) is improved by 3.85%, 0.62%, 1.51%, and 2.78% on MAE and 3.41%, 0.63%, 3.50%, and 2.42% on MAPE respectively. In particular, the improvements are more significant for the PeMSD3 and PeMSD8 datasets.

We observe that statistical methods such as HA, ARIMA, and VAR have difficulty in handling nonlinear, non-stationary time series data and the model prediction errors are high. Although deep learning methods such as FC-LSTM and TCN have advantages over statistical models, they only consider temporal correlations and cannot exploit spatial dependencies, thus having a limited ability to model spatial-temporal data. Spatial-temporal graph networks such as STGCN [35], DCRNN [17], and GWNET [33] are designed with spatial-temporal components, so they generally have better performance compared with temporal-only-based methods. AGCRN and DSTAGNN are free from the limitation of fixed graph structure and use self-adaptive or dynamic properties of spatial association between nodes in historical data to construct the graph structure. They further improve the prediction performance compared with the previous methods. In contrast, our method not only considers the effects of synchronous spatial-temporal correlation but also mines the multi-scale and long-distance traffic fluctuation relationships from the perspective of spatial-temporal attention.

Furthermore, we quantify the difference between our model and the best baseline model using the relative error rate. Across the four datasets, the average MAE, RMSE, and MAPE values for AFDGCN are 17.33, 28.75, and 11.25%, respectively; the corresponding average values of DSTAGNN [15] are 18.05 (104.2%), 29.48 (102.5%), and 11.58% (103.0%), and the average error performance of STG-NCDE [4] are 17.69 (102.1%), 29.21 (101.6%) and 11.64% (103.5%). Taken together, the improved gain of our model compared to these two models ranges from 1.6% to 4.2%.

To clearly examine the difference in prediction performance between our method and the baseline methods, we plot in Fig. 3 the predicted and ground truth of our method (AFDGCN) versus STG-NCDE for several stations/nodes within a given day. AFDGCN and STG-NCDE well fit the real conditions of traffic variations in many time periods, but our model shows more accurate prediction performance in challenging traffic scenarios, e.g., during traffic peak hours or when the magnitude of fluctuations is drastic. In particular, our method adapts well to changes in traffic trends, e.g., 5:00-8:00 in (a), 4:00-7:00 in (b), 10:00-12:00 in (c), and 6:00-9:00 in (d). In contrast, the prediction curves of STG-NCDE significantly deviate from the ground truth because of its limited predictive power.

## 5.3 *Additional Experimental Studies*

**Efficiency study.** We first compare the computational cost of our model AFDGCN with several competitive baseline models in the PEMSD4 using the same Graphic Card NVIDIA Tesla V100. As shown in Table 3, AFDGCN has a comparable training and infer-

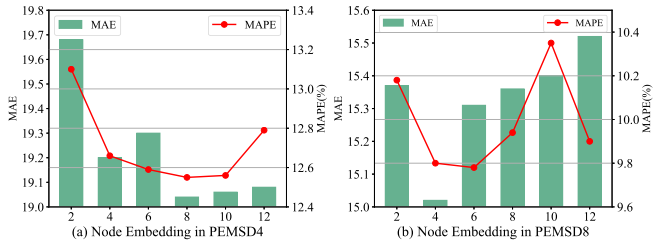


**Figure 4**: The effects of the Node Embedding Dimension.

ence time as AGCRN but attains much better predictive performance. Compared with STGODE, the training time and inference time of AFDGCN are reduced by 56.13% and 64.10%. Although STG-NCDE has fewer parameters than AFDGCN, the complex model structure and operators make the computational cost of STG-NCDE significantly higher than AFDGCN. The training time and inference time of AFDGCN are 4-5x and 4-6x faster than DSTAGNN and STG-NCDE, respectively. Remarkably, our model outperforms all the baseline methods in the predictive performance by only using highly competitive training and inference time.

**Ablation study.** We perform an ablation study on different modules of our model to investigate their individual contributions in the PeMSD4 and PeMSD8. Let **DGCGRU** denotes the dynamic graph convolution recurrent network, and **DGCGRU with Attn** denote DGCGRU with multi-head temporal attention. We denote the full model AFDGCN with the feature augmentation removed and AFDGCN with the graph attention layer removed as **w/o FAL** and **w/o GAT**, respectively. The prediction results of these methods for PeMSD4 and PeMSD8 are summarized in Table 4. DGCGRU performs normally as expected. After adding the multi-headed temporal attention mechanism, DGCGRU with Attn achieves significant improvements in the three evaluation metrics. This demonstrates the effectiveness of the temporal attention mechanism in modeling global temporal dependence. In addition, AFDGCN with either the feature augmentation layer or the graph attention layer removed has a noticeable loss in the prediction performance.

**Hyperparametric study.** An essential parameter in AFDGCN is the node embedding size, which not only affects the quality of the learning graph but also determines the diversity of parameters in the DGCGRU layer. Fig. 4 show the effects of different node embedding dimensions on the model prediction results. It can be found that the optimal node embedding dimensions for the PeMSD4 and PeMSD8 datasets are 8 and 4, respectively. On the one hand, node embedding with a larger value contains more parameter information, which improves the expressiveness of the model to infer more complete spatial correlations. On the other hand, the larger the number of parameters, the more likely the model is prone to overfitting.

**Interpretability study.** Now we provide a shred of evidence on the benefit of using an adaptive graph structure. We visualize the correlation between some node pairs under adaptive and pre-defined graphs using heatmaps in PeMSD4, as shown in Fig. 5. The darker the color, the stronger the correlation between the nodes. The pre-defined graph (right) relies only on geographic distance, and the information presented by the heatmap is sparse and single. In contrast, the information in the dynamically generated graph (left) learned with parameters is dense and diverse. We take two points in the road network as an example (as shown by the star in the figure), which do not exhibit correlation in the pre-defined static graph. But in fact, the two points (Node 117, Node 118) are very close to each other in terms of the period and trend changes of traffic as in the bottom figure. Therefore,

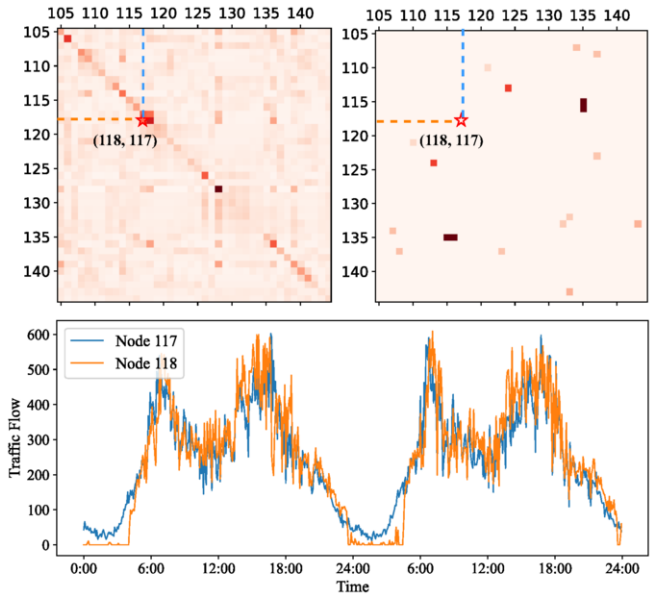


**Figure 5**: The heatmap visualization for dynamic generation graph (Left) and pre-defined graph (Right) in PeMSD4.

the dynamic adjacency matrix can implicitly learn the dynamic representation of the road network and provide an effective complement to the static adjacency matrix.

**Table 3**: The training and inference time in PEMSD4 dataset under a Tesla V100 GPU(s/epoch).

| Model | #Params | #Training | #Inference |
|---|---|---|---|
| DSTAGNN | 3,579,728 | 134.72 | 14.94 |
| STG-NCDE | 322,588 | 92.04 | 10.94 |
| STGODE | 714,504 | 57.38 | 7.27 |
| AGCRN | 748,810 | 21.68 | 2.88 |
| AFDGCN | 435,121 | 25.17 | 2.61 |

**Table 4**: The results of an ablation study in AFDGCN.

| Module | PEMSD4 | | | PEMSD8 | | |
|---|---|---|---|---|---|---|
| | MAE | RMSE | MAPE | MAE | RMSE | MAPE |
| DGCGRU | 19.80 | 32.20 | 13.18% | 16.01 | 25.51 | 10.35% |
| DGCGRU with Attn | 19.32 | 31.87 | 12.79% | 15.53 | 24.91 | 10.01% |
| AFDGCN w/o FAL | 19.11 | 31.65 | 12.67% | 15.35 | 24.49 | 9.98% |
| AFDGCN w/o GAT | 19.11 | 31.48 | 12.61% | 15.29 | 24.68 | 9.71% |
| AFDGCN | 19.04 | 31.11 | 12.55% | 15.02 | 24.36 | 9.80% |

## 6 Conclusion

In this paper, we propose a novel spatial-temporal graph neural network for traffic prediction. The model enhances the traditional GCN by adopting a dynamic generation graph with node parametric learning, and combines the improved GCN with GRU to capture synchronous spatial-temporal correlation. To handle long-range and multi-view changes in complex traffic scenes, we introduce a spatial-temporal attention fusion module to effectively improve the model's performance. We conduct experiments on four publicly available traffic datasets, and the results demonstrate the effectiveness and superiority of AFDGCN. In future studies, we plan to extend AFDGCN to other spatial-temporal prediction tasks and explore modeling dynamic spatial dependencies with time-varying properties.

## Acknowledgements

Detian Zhang is supported by the Collaborative Innovation Center of Novel Software Technology and Industrialization, the Priority Academic Program Development of Jiangsu Higher Education Institutions. Chunjiang Zhu is supported by UNCG Start-up Funds and Faculty First Award.

## References

[1] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang, 'Adaptive graph convolutional recurrent network for traffic forecasting', in *Advances in Neural Information Processing Systems*, (2020).

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun, 'An empirical evaluation of generic convolutional and recurrent networks for sequence modeling', *arXiv preprint*, **arXiv:1803.01271**, (2018).

[3] Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia, 'Freeway performance measurement system: mining loop detector data', *Transportation Research Record*, 96–102, (2001).

[4] Jeongwhan Choi, Hwangyong Choi, Jeehyun Hwang, and Noseong Park, 'Graph neural controlled differential equations for traffic forecasting', in *AAAI*, pp. 6367–6374, (2022).

[5] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio, 'Empirical evaluation of gated recurrent neural networks on sequence modeling', *arXiv preprint arXiv:1412.3555*, (2014).

[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst, 'Convolutional neural networks on graphs with fast localized spectral filtering', in *Advances in Neural Information Processing Systems*, pp. 3837–3845, (2016).

[7] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan, 'Attention based spatial-temporal graph convolutional networks for traffic flow forecasting', in *AAAI*, pp. 922–929, (2019).

[8] Shengnan Guo, Youfang Lin, Huaiyu Wan, Xiucheng Li, and Gao Cong, 'Learning dynamics and heterogeneity of spatial-temporal graph data for traffic forecasting', *IEEE Trans. Knowl. Data Eng.*, **34**(11), 5415–5428, (2022).

[9] Mikael Henaff, Joan Bruna, and Yann LeCun, 'Deep convolutional networks on graph-structured data', *CoRR*, **abs/1506.05163**, (2015).

[10] Renhe Jiang, Du Yin, Zhaonan Wang, Yizhuo Wang, Jiewen Deng, Hangchen Liu, Zekun Cai, Jinliang Deng, Xuan Song, and Ryosuke Shibasaki, 'Dl-traff: Survey and benchmark of deep learning models for urban traffic prediction', in *CIKM*, pp. 4515–4525, (2021).

[11] Weiwei Jiang and Jiayun Luo, 'Graph neural network for traffic forecasting: A survey', *Expert Syst. Appl.*, **207**, 117921, (2022).

[12] Diederik P. Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', in *ICLR*, (2015).

[13] Thomas N. Kipf and Max Welling, 'Semi-supervised classification with graph convolutional networks', in *ICLR*, (2017).

[14] S Vasantha Kumar and Lelitha Vanajakshi, 'Short-term traffic flow prediction using seasonal arima model with limited input data', *European Transport Research Review*, 1–9, (2015).

[15] Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li, 'Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting', in *ICML*, pp. 11906–11917, (2022).

[16] Fuxian Li, Jie Feng, Huan Yan, Guangyin Jin, Depeng Jin, and Yong Li, 'Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution', *CoRR*, **abs/2104.14917**, (2021).

[17] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu, 'Diffusion convolutional recurrent neural network: Data-driven traffic forecasting', in *ICLR*, (2018).

[18] Yichao Liu, Zongru Shao, and Nico Hoffmann, 'Global attention mechanism: Retain information to enhance channel-spatial interactions', *CoRR*, **abs/2112.05561**, (2021).

[19] Bin Lu, Xiaoying Gan, Haiming Jin, Luoyi Fu, and Haisong Zhang, 'Spatiotemporal adaptive gated graph convolution network for urban traffic flow forecasting', in *CIKM*, pp. 1025–1034, (2020).

[20] Zheng Lu, Chen Zhou, Jing Wu, Hao Jiang, and Songyue Cui, 'Integrating granger causality and vector auto-regression for traffic prediction of large-scale wlans', *KSII Transactions on Internet and Information Systems (TIIS)*, **10**(1), 136–151, (2016).

[21] Chao Song, Youfang Lin, Shengnan Guo, and Huaiyu Wan, 'Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting', in *AAAI*, pp. 914–921, (2020).

[22] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, 'Sequence to sequence learning with neural networks', *Advances in neural information processing systems*, **27**, (2014).

[23] David Alexander Tedjopurnomo, Zhifeng Bao, Baihua Zheng, Farhana Choudhury, and Alex Kai Qin, 'A survey on modern deep neural network for traffic prediction: Trends, methods and challenges', *IEEE Transactions on Knowledge and Data Engineering*, (2020).

[24] JWC Van Lint and CPIJ Van Hinsbergen, 'Short-term traffic and travel time prediction models', *Artificial Intelligence Applications to Critical Transportation Issues*, **22**(1), 22–41, (2012).

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, 'Attention is all you need', in *Advances in Neural Information Processing Systems*, pp. 5998–6008, (2017).

[26] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio, 'Graph attention networks', *stat*, **1050**, 20, (2017).

[27] Xiaoyang Wang, Yao Ma, Yiqi Wang, Wei Jin, Xin Wang, Jiliang Tang, Caiyan Jia, and Jian Yu, 'Traffic flow prediction via spatial temporal graph neural network', in *WWW*, pp. 1082–1092, (2020).

[28] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon, 'Cbam: Convolutional block attention module', in *ECCV*, pp. 3–19, (2018).

[29] Chun-Hsin Wu, Jan-Ming Ho, and Der-Tsai Lee, 'Travel-time prediction with support vector regression', *IEEE Trans. Intell. Transp. Syst.*, 276–281, (2004).

[30] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu, 'A comprehensive survey on graph neural networks', *IEEE Trans. Neural Networks Learn. Syst.*, **32**(1), 4–24, (2021).

[31] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang, 'Connecting the dots: Multivariate time series forecasting with graph neural networks', in *KDD*, pp. 753–763, (2020).

[32] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang, 'Graph wavenet for deep spatial-temporal graph modeling', in *IJCAI*, pp. 1907–1913, (2019).

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

[34] Mingxing Xu, Wenrui Dai, Chunmiao Liu, Xing Gao, Weiyao Lin, Guo-Jun Qi, and Hongkai Xiong, 'Spatial-temporal transformer networks for traffic flow forecasting', *CoRR*, **abs/2001.02908**, (2020).

[35] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li, 'Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction', in *AAAI*, pp. 5668–5675, (2019).

[36] Bing Yu, Haoteng Yin, and Zhanxing Zhu, 'Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting', in *IJCAI*, pp. 3634–3640, (2018).

[37] Hongyuan Yu, Ting Li, Weichen Yu, Jianguo Li, Yan Huang, Liang Wang, and Alex X. Liu, 'Regularized graph structure learning with semantic knowledge for multi-variates time-series forecasting', in *IJCAI*, pp. 2362–2368, (2022).

[38] Junbo Zhang, Yu Zheng, and Dekang Qi, 'Deep spatio-temporal residual networks for citywide crowd flows prediction', in *AAAI*, pp. 1655–1661. AAAI Press, (2017).

[39] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi, 'GMAN: A graph multi-attention network for traffic prediction', in *AAAI*, pp. 1234–1241, (2020).