Applied Intelligence (2022) 52:4300–4316
https://doi.org/10.1007/s10489-021-02648-0

Adaptive spatial-temporal graph attention networks for traffic flow forecasting

Xiangyuan Kong[1] · **Jian Zhang**[1] · **Xiang Wei**[1] · **Weiwei Xing**[1] · **Wei Lu**[1]

Accepted: 25 June 2021 / Published online: 20 July 2021

Abstract

Traffic flow forecasting, which requires modelling involuted spatial and temporal dependence and uncertainty regarding road networks and traffic conditions, is a challenge for intelligent transportation systems (ITS). Recent studies have mainly focused on modelling spatial-temporal dependence through a fixed weighted graph based on prior knowledge. However, collecting up-to-date and accurate road information is costly. Moreover, is a single fixed graph enough to describe the correlation between sensors? The fixed weighted graph cannot directly relate to prediction tasks, which may result in considerable biases. To tackle this issue, in this paper, we propose a novel deep learning model framework: an adaptive spatial-temporal graph attention network (ASTGAT). Our ASTGAT simultaneously learns the dynamic graph structure and spatial-temporal dependency for traffic flow forecasting. Specifically, our framework consists of two joint training parts: a Network Generator model that generates a discrete graph with the Gumbel-Softmax technique and a Spatial-Temporal model that utilizes the generated network to predict traffic speed. Our Network Generator can adaptively infer the hidden correlations from data. Moreover, we propose a graph talking-heads attention layer (GTHA) for capturing spatial dependencies and design a gate temporal convolution (GTCN) layer for handling long temporal sequences. We evaluated our ASTGAT on two public datasets: METR-LA is collected in Los Angeles and PEMS-BAY is collected in California. Experimental results indicate that our ASTGAT outperforms the state-of-the-art (SOTA) baselines. Finally, to further describe our model, we visualize the forecasting results and the generated graph.

Keywords Traffic flow forecasting · Spatial-temporal graph neural networks · Talking-heads attention · Intelligent transportation systems

✉ Wei Lu
luwei@bjtu.edu.cn

Xiangyuan Kong
xiangyuankong@bjtu.edu.cn

Jian Zhang
jianzh@bjtu.edu.cn

Xiang Wei
xiangwei@bjtu.edu.cn

Weiwei Xing
wwxing@bjtu.edu.cn

[1] School of Software Engineering, Beijing Jiaotong University, Beijing, China

1 Introduction

Traffic forecasting has gained more attention with the rapid development of intelligent transportation systems (ITSs) [1]. The objective of traffic flow prediction is to provide future traffic flow conditions in road networks based on historical observations. Previous work divided traffic flow forecasting into 3 categories based on the length of the forecasting time window: short-term forecasting (from 5 to 30 minutes), medium-term forecasting (from 30 to 60 minutes), and long-term forecasting (over 60 minutes) [2].

Numerous deep learning models have been previously proposed for traffic speed prediction [1, 3–6]. However, previous approaches with Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) can

only process grid structures (e.g., images and videos), and ignore the non-Euclidean correlations determined by complex road networks. To tackle this problem, recent studies on spatial-temporal graph modelling formulate traffic forecasting as a graph modelling problem. Graph Neural Networks (GNNs) [7, 8] are utilized to capture such spatial correlations in traffic networks, and time series models capture the temporal correlations in time sequences.

Spatial-temporal graphs are a popular method for dealing with complex spatial and temporal correlations. Most spatial-temporal (ST) graph models integrated with RNNs [9–11] or CNNs [12, 13] to embed the prior knowledge of the road network and capture the correlations between pairwise nodes. Although introducing the graphical structure into the model achieves a better performance, these previous methods still have the following challenges.

First, the previous methods only construct the fixed weighted directed graph according to the node similarity. However, the fixed weighted adjacency matrix is difficult to generate, e.g., some free editable maps (for example, OpenStreetMap [14]) has difficulty keeping spatial topology information up-to-date. On the other hand, researchers have explored various ways of using the given graph structure as input for GNNs, but without considering whether this fixed weighted graph is suitable for downstream tasks. For example, in addition to the nearby nodes, distant nodes may also share similar temporal patterns. Previous work [15] consideed the correlation between distant roads that share a similar temporal pattern. Therefore, [15] construct the adjacency matrix based on time series similarity. The literature [16] has a similar idea, Dynamic Time Warping (DTW) algorithm is used to measure the similarity of time series. ST-MGCN [17] suggests that non-Euclidean pairwise dependencies among distant regions are also important for forecasting. Hence, the ST-MGCN defines the node proximity by measuring the similarity of PoI information. A fixed weighted directed graph cannot represent the relationship between these nodes. Therefore, a potential dynamic reconstruction graph structure based on a given fixed weighted graph adjacency matrix (obtained from observed time series data) is an essential task. Our proposed ASTGAT uses historical node data to infer the connection structure between nodes and learn the dynamics of the interaction between nodes. Second, recent studies generally use the existing GNNs approach directly or an attention mechanism to capture spatial dependencies. Hence, further improving existing models to achieve better performance is also important. Third, in long-term prediction, there is much redundant information not related to future traffic conditions, which makes prediction the far future very challenging.

To overcome the above challenges, we propose a novel deep learning framework to predict future traffic conditions, namely an Adaptive Spatial-Temporal Graph Attention Networks (ASTGAT). Our ASTGAT aims to predict traffic conditions on a road network graph over time steps ahead without requiring pre-specified graph. Inspired by recent studies on graph reconstruction [18–20], our ASTGAT can jointly learn the graph structure and spatial-temporal correlations simultaneously. We design our model as two joint training parts: Network Generator Model that generates a network adjacency matrix with the Gumbel-SoftMax technique and a Spatial-Temporal Model that uses the generated network to predict traffic speed in future steps. For the network generator, we use the Gumbel-softmax trick [21] to rebuild the adjacency matrix of the graph. Moreover, inspired by talking-heads attention [22], we propose a graph talking-heads attention (GTHA) layer, which improves the performance of graph attention network [8], to capture spatial dependencies. We also design a gating temporal convolution (GTCN) layer to filter the redundant information in the historical sequence and are capable of handling long sequences. We aggregate the GTHA and GTCN layers as spatial-temporal block to consider spatial-temporal dependencies at the same time. The main contributions of our work are summarized as follows.

- We propose a novel deep learning model framework ASTGAT that consists of two jointly trained parts: a network generator and Spatial-Temporal model. Our ASTGAT can build a hidden graph structure from traffic data, to augment the initial graph structure (even if the initial graph structure is missing) and capture spatial-temporal correlations simultaneously.
- Instead of just using prior knowledge (constructing a fixed weighted directed graph according to the node proximity), we propose a Network Generator model, which can adaptively reconstruct the graph adjacency matrix from historical traffic data. Experimental results show that the performance of this generated graph structure outperforms manually constructed graph structures.
- We propose a talking-heads graph attention layer to better capture spatial dependencies. Moreover, we design a gating temporal convolution layer to handle long historical sequences. Combining these two components, our ASTGAT achieves better performance in long-term prediction.
- Our ASTGAT is evaluated on two public real-world traffic datasets and achieves state-of-the-art results. The source codes of ASTGAT will be publicly available from GitHub.

2 Related work

2.1 Traffic flow forecasting

Modern communication systems and networks (Internet of Things (IoT) and cellular networks) generate a massive amount of traffic data [23]. Deep learning has been effectively used to facilitate analysis and knowledge discovery in big data systems to identify hidden and complex patterns. For example, TSCRNN [24] proposed a method for traffic classification, [25] proposed a reliable VANET routing decision scheme, [26] studied for learning a responsive traffic control policy for short-term traffic demand changes, [27] proposed a parallel computing approach of traffic network flow control, [28] researched traffic sign recognition algorithm, and [29] studied for predicting the urban traffic volume.

Previous studies often treat traffic as flow data, because of its close properties to fluid [30]. Traffic flow forecasting has always been regarded as an important part of ITSs. The prediction results help prevent unexpected events, such as traffic jams or other abnormal conditions on the road. Traffic flow forecasting is a classic time series forecasting task, that is, forecasting the next most likely p step traffic conditions given h step historical traffic flow observation data.

In the 1970s, the Autoregressive Integrated Moving Average model (ARIMA) was used to predict short-term highway traffic flow [31]. There are many studies based on variants of ARIMA, such as VARMA [32], the KARIMA method [33] and STARIMA [32, 34]. Moreover, [35, 36] both use a Kalman Filter based approach to predict traffic conditions. Nikovski et al. [37] is a Linear Regression (LR) based approach and [38] is a k-nearest neighbor (KNN) based approach. Most of the early traffic prediction methods use shallow models, and the results are still not satisfactory. The latest advanced approaches in deep learning show better performance compared to shallow models, such as reference [1, 39].

The key to achieving a better prediction performance is to model complicated spatial and temporal dependencies. Graph Spatial-Temporal Neural Networks assemble time series models and graph neural network model (e.g., graph convolutional neural networks (GCNs) [7] and graph attention networks (GATs) [8]) to consider spatial and temporal dependencies simultaneously. Many researchers have proposed new approaches for effective spatial-temporal correlation modelling. They view sensor node networks as a fixed weighted graph according to the node similarity. For example, Diffusion Convolutional Recurrent Neural Network (DCRNN) [10] proposes an encoder-decoder architecture and further improvement GRU [40] by replacing the FC layer with a diffusion convolution layer. GGRU [41] uses an RNN-based approach and attention mechanisms in graph convolution. In addition, STGCN [12] is a CNN-based approach that combines GCN layers with 1D convolutional layers. ASTGCN [42] applies an attention mechanism to the spatial and temporal dimensions respectively. Furthermore, Graph Wavenet [13] adjusts an adaptive dependency matrix through the learned node embedding, and improves on STGCN [12] with a stacked dilated 1D convolution operation. MRA-BGCN [43] uses a bicomponent graph convolution to model the correlations of both nodes and edges. GMAN [44] is a graph multi-attention network that use spatial and temporal attention mechanisms and a gate mechanism to fuse complex spatial-temporal correlations. ST-GRAT [45] uses node attention operate to handle the spatial dependence among traffic sensors and temporal attention is used to consider the temporal dependence.

2.2 Graph neural network

Various attempts have been made in the previous studies to extend neural networks to handle graphs with arbitrary structures. Graph Neural Networks (GNNs) was first mentioned in [46], and [47] can handle more general graphs directly, including cyclic, directed, and undirected graphs [48]. GNNs play a key role in dealing with structural dependencies. The first spectral convolution operation was proposed by SpectralCNN [49]. Chebyshev [50] defined the Chebyshev polynomial of the diagonal matrix of the eigenvector as a filter avoid calculating the eigenvectors of the Laplacian. ChebNet [7] restricts the filters to calculate in an 1-hop neighbourhood around each node to simplify the computational complexity. Graph Attention Networks (GAT) [8] allow different weights to be assigned to different neighbouring nodes.

2.3 Attention mechanism

Neural Attention was introduced by [51] extracting information from variable length representations. Vaswani et al. [52] proposed the Transformer model, which has a multi-head attention mechanism. A multi-hop attention mechanism is proposed in [53] as a possible choice for integrating information from multi-head attention in Transformer. Shazeer et al. [22] proposed a talking-heads mechanism by including linear projections across the heads before and after the softmax operation in the original multi-head attention mechanism. In this paper, we extend the graph attention network with a talking-heads attention mechanism for traffic flow forecasting.

3 Methodology

This section proposes the framework of ASTGAT for traffic flow forecasting. We first introduce some notations, definitions and preliminaries. Then, we introduce our proposed model and describe how it works for the traffic flow forecasting task.

3.1 Preliminaries

3.1.1 Traffic networks

The traffic network can be represented as a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$. In the formula, $\mathcal{V}$ is the set of all nodes (representing sensors in the traffic road networks), the scale is $|\mathcal{V}| = N$. E is the set of all edges (indicates the strength of connectivity between nodes). The graph $\mathcal{G}$ is presented by adjacency matrix $\mathcal{W} \in \mathbb{R}^{N \times N}$, where $\mathcal{W}_{i,j}$ represents the distance function (define the graph based on the geographic road network distance) or similarity function (define the graph by the similarity of node features, e.g., PoI information and traffic time series) between $Node_i$ and $Node_j$. Both datasets are aggregated into 5-minute windows, resulting in 288 time steps per day. For time step t, it has a feature matrix $X \in \mathbb{R}^{N \times F}$. In other words, each node have F quantification value at time step t in the traffic network $\mathcal{G}$.

3.1.2 Problem definition

Traffic flow forecasting is a classic time series forecasting task. The historical observation traffic data H is denoted by $H = \{X_{t-h+1}, X_{t-h+2}, ..., X_t\}$ (represent h historical time steps, each step t has a feature vector X_t). The flow observation data of a given H is used to predict the next p most likely time steps. Assume that we have h historical time steps, and time step t has a feature matrix X_t. According to the above notation, the problem can be defined as follows: given the historical data H, predict the next p time step traffic flow $P = \{X_{t+1}, X_{t+2}, ..., X_{t+p}\}$. In previous research, the problem is classified into three categories (based on the size of p): short-term forecasting ($5minutes \leq p \leq 30minutes$), medium-term forecasting ($30minutes \leq p \leq 60minutes$), long-term forecasting $60minutes \leq p$.

3.2 Overview of model architecture

Figure 1 illustrates the architecture of our proposed ASTGAT, which consists of two co-trained parts: a Network Generator model that generates a network adjacency matrix with the Gumbel-Softmax trick and a Spatial-Temporal model that uses the generated network to predict traffic speed. Instead of just using prior knowledge (i.e., constructing a fixed weighted directed graph according to the node proximity), our proposed Network Generator model can adaptively reconstruct the graph adjacency matrix from the historical traffic data. The Spatial-Temporal Model is construct by GTCN and GTHA layers. We use the GTCN layer to capture the temporal dependency and the GTHA layer to handle the spatial intercorrelations.

Using a threshold Gaussian kernel [54] is a classic method of constructing a graph adjacency matrix based on road network distances. Following the previous works on DCRNN [10] and Graph Wavenet [13], we construct the initial weighted adjacency matrix of the traffic sensor according to the road network distance. In (1), $\mathcal{W}_{ij}$ represents the connection strength between traffic sensors v_i and v_j, and $\text{dist}(v_i, v_j)$ indicates the distance on the road network between traffic sensors v_i to v_j. σ is the

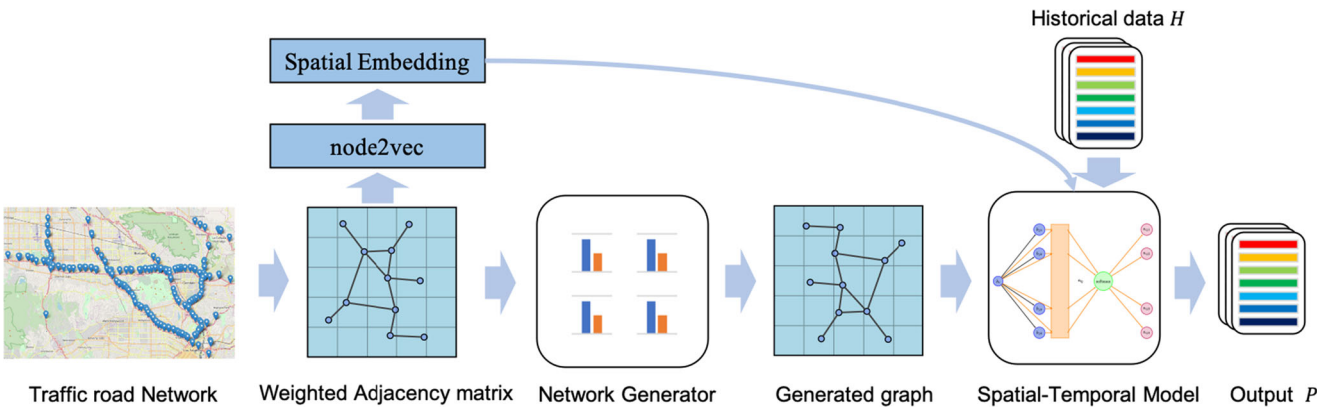


Fig. 1 Basic structure of ASTGAT. Our framework contains two main parts: a Network Generator and a Spatial-Temporal Model. We construct a fixed weighted adjacency matrix based on the road network. The Network Generator generates new weighted adjacency matrix with the information from the give adjacency matrix. The Spatial-Temporal Model uses the new graph and historical traffic data to predict future traffic conditions

standard deviation of the road network distances and k is the threshold.

$$\mathcal{W}_{ij} = \begin{cases} \exp\left(-\frac{\text{dist}(v_i, v_j)^2}{\sigma^2}\right) & , \text{dist}(v_i, v_j) > k \\ 0 & , \text{otherwise} \end{cases} \tag{1}$$

Second, we leverage the node2vec approach [55] to learn the initial spatial embedding of nodes to provide static representations SE. Then, we input $\mathcal{W}$ into the Network Generator model to generate the new adjacency matrix $\mathcal{A}$. Spatial-Temporal model will use the generated adjacency matrix $\mathcal{A}$, initial spatial embedding SE and the historical traffic data $H = \{X_{t-h+1}, X_{t-h+2}, ..., X_t\}$ as input to predict the next p time step graph signals $P = \{X_{t+1}, X_{t+2}, ..., X_{t+p}\}$. Instead of generating X_p recursively through p steps, our ASTGAT predicts $P = \{X_{t+1}, X_{t+2}, ..., X_{t+p}\}$ as a whole.

3.3 Network generator

The difficulty in reconstructing a network from the historical data is the discreteness of the graph. The back-propagation technique cannot be applied directly. To tackle this issue, we use Gumbel-softmax trick [21] to rebuild the adjacency matrix of the network following [20]'s work. Network generator uses an $N \times N$ parameterized matrix to determine the $N \times N$ cells in the adjacency matrix $\mathcal{A}$, where N is the number of nodes the in initial graph. We can generate an adjacency matrix $\mathcal{A}$ using (2):

$$\mathcal{A}_{ij} = \frac{e^{((a_{ij_1} + \xi_{ij})/\tau}}{e^{((a_{ij_1} + \xi_{ij})/\tau} + e^{((a_{ij_2} + \xi'_{ij})/\tau}}, \tag{2}$$

where a_{ij_1} and a_{ij_2} are trainable parameters, a_{ij_1} indicates the probability that $\mathcal{A}_{ij}$ takes a value of 1, ξ_{ij}, and ξ'_{ij} are independent and identically distributed random numbers following a Gumbel distribution [56]. τ is the temperature parameter, which adjusts the sharpness of the output. When $\tau \to 0$, $\mathcal{A}_{ij}$ will take a value of 1 with probability a_{ij_1} or a value of 0 with probability a_{ij_2}. Thanks to the features of the Gumbel-SoftMax trick, the gradient information can be back-propagated through the whole computation graph. Furthermore, we want a weighted adjacency matrix, so we multiply $\mathcal{A}_{ij}$ by a trainable parameter b_{ij}, as shown in (3).

$$\mathcal{A}_{wij} = \mathcal{A}_{ij} \times b_{ij}, \tag{3}$$

To ensure a stable learning process, we extend the generation process by repeating it K times and using the average of K times matrix $\mathcal{A}_w^{(K)}$. Moreover, we also consider fusing the information of the given weight adjacency matrix. The formula is shown in (4)

$$\mathcal{A}_w = \frac{\lambda}{K} \sum_{k=1}^{K} \mathcal{A}^{(k)} + (1 - \lambda)\mathcal{W}, \tag{4}$$

where $\lambda \in [0, 1]$ is the hyperparameter, $\lambda = 0$ means do not use the given weighted adjacency matrix, and $\lambda = 1$ means use the initial adjacency matrix without augmentation.

3.3.1 Graph regularization

To use the Network Generator to generate more suitable graphs for traffic forecasting, we added some constraints to the generated graphs. A widely adopted assumption for graph signals is that values change smoothly across adjacent nodes [54]. In urban transportation, the traffic conditions in adjacent areas are similar most of the time (e.g., the speed of each node will be affected by the speed of all the adjacent nodes). Therefore, we introduce a graph regularization loss for smoothness following the previous work [18]. For generated weighted adjacency matrix $\mathcal{A}_w$, the smoothness constraints of the nodes spatial-temporal embedding $X_{emb} = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^{N \times F_{emb}}$ are as follows.

$$\begin{aligned} \Omega(\mathcal{A}_w, X) &= \frac{1}{2N^2} \sum_{i,j} \mathcal{A}_{wij} ||x_i - x_j||^2 \\ &= \frac{1}{2N^2} tr(X_{emb}^T L X_{emb}) \end{aligned} \tag{5}$$

where N is the number of nodes, $tr(\cdot)$ denotes the trace of a matrix, $L = D - \mathcal{A}_w$, and $D = \sum_j A_{wij}$ is the degree matrix. Minimizing the smoothness constraints $\Omega(\mathcal{A}_w, X)$ will force adjacent nodes to have similar embeddings. However, only minimizing the smoothness constraints will lead to $\mathcal{A}_w = 0$. Following [18], we also apply two additional constraints (connectivity and sparsity) to the generated graph.

$$f(\mathcal{A}) = \frac{-\beta}{N} \mathbf{1}^T log(\mathcal{A}\mathbf{1}) + \frac{\gamma}{N^2} ||\mathcal{A}||_F^2 \tag{6}$$

where $||\dddot{|}|_F$ denotes the Frobenius norm. The previous term is the connectivity loss (minimizing the connectivity constraints prevents isolated nodes from existing in the graph), and the second term is obviously a loss of sparsity (minimizing the sparsity constraints can force the graph to be sparse).

In our work, we borrow this general prior knowledge and apply it as regularization to the graph generated by Network Generator (4). In general, the graph regularization loss constrains the adaptively generated graph from three aspects: smoothness, connectivity and sparsity. The formula is as follows:

$$\mathcal{L}_{\mathcal{G}} = \alpha \Omega(\mathcal{A}, X) + f(\mathcal{A}) \tag{7}$$

where α, β and γ are all non-negative hyperparameters.

3.4 Spatial-temporal model

The Spatial-Temporal Model aims to capture spatial and temporal dependencies at the same time. The Spatial-Temporal Model consists of m spatial-temporal layers and Fully Convolutional Networks (namely EndConv) to output the prediction results. Each layer contains p gated temporal convolution (GTCN) layers and one graph talking-heads attention layer (GTHA). In each spatial-temporal layer, we first use the proposed GTCN layer to capture the complex and dynamic temporal correlation. Then we use the proposed GTHA layer to propagate temporal-state among generated graph to capture the spatial correlation. The spatial-temporal layers combine GTHA and GTCN, which can extract the most useful spatial features and capture the most essential temporal features coherently. Moreover, for the graph talking-heads attention layer, an initial spatial embedding get through node2vec approach and the generated adjacency matrix from the Network Generator is required. Through m spatial-temporal layers, we can obtain the spatial-temporal embedding $X_{emb} \in \mathbb{R}^{N \times F_{emb}}$ of each node, where F_{emb} is the dimension of the embedding, and N is the number of nodes. Finally, the spatial-temporal embedding X_{emb} is used to predict traffic conditions. As aforementioned in Preliminaries, the input data is graph-structured timeseries. In the training phase, the input including traffic speed and time (same with the setting in paper DCRNN [10]). In order to fuse features from both spatial and temporal domains, the Spatial-Temporal Model is constructed to jointly process graph-structured timeseries. In each spatial-temporal layer, we first use the proposed GTCN layer to capture the complex and dynamic temporal correlation. Then we leverage to use the proposed GTHA layer to propagate temporal-state among generated graph to capture the spatial correlation. The spatial-temporal layers combine GTHA and GTCN, which can extract the most useful spatial features and capture the most essential temporal features coherently.

3.4.1 Graph talking-heads attention layer

Graph Attention Network (GAT) [8] is a novel attention mechanism based approach that operates on graph-structured data. We propose a Graph Talking-heads Attention (GTHA) layer based on a GAT to capture the highly dynamic impact over nodes. We introduce linear projections across all the attention-heads before and after the softmax operation. The implementation of the GTHA mechanism is as follows. The input is the node feature matrix (i.e., the initial spatial embedding of the node and traffic state is concatenated), $\mathbf{h} = \{\boldsymbol{h}_1, \boldsymbol{h}_2, ..., , \boldsymbol{h}_N\}, \boldsymbol{h}_i \in \mathbb{R}^F$ (where N nodes in the graph and each node has F features). The formula of the GTHA is shown in (8) to (14).

$$e_{ij}^{(l)} = LeakyReLU(\boldsymbol{\alpha}_i^{\boldsymbol{T}(l)}(W^{(l)}\boldsymbol{h_i} \parallel W^{(l)}\boldsymbol{h_j})) \tag{8}$$

In (8), $\parallel$ denotes concate operation, (l) means the lth head in L multi-heads, $\boldsymbol{\alpha}_i^{(l)} \in \mathbb{R}^{2F'}$ and $W^{(l)} \in \mathbb{R}^{F' \times F}$ are both learnable weight parameters, we use LeakyReLU as activation function. Then we insert a learned linear projection across the attention-heads dimension of attention coefficients $e_{ij}^{(l)}$ to get $e_{ij}^{\prime(l)}$ as shown in (12)–(13).

$$e'_{ij} = W_e \left|\left|_{l=1}^{L} e_{ij}^{(l)}\right. \right. , \tag{9}$$

where $W_e \in \mathbb{R}^{L \times L}$ is learnable parameters, and

$$e'_{ij} = ||_{l=1}^{L} e_{ij}^{\prime(l)}, e_{ij}^{\prime(l)} \in \mathbb{R}^{N \times N} \tag{10}$$

Then we normalize $e_{ij}^{\prime(l)}$ across all choices of j using the softmax function, $\mathcal{N}_{(i)}$ denotes node i's neighborhood, the formula is shown follow:

$$a_{ij}^{(l)} = softmax_j(e_{ij}^{\prime(l)}) = \frac{exp(e_{ij}^{\prime(l)})}{\sum_{k \in \mathcal{N}_{(i)}} exp(e_{ik}^{\prime(l)})} \tag{11}$$

After that, we insert the second linear projection following the softmax operate.

$$a'_{ij} = W_a \left|\left|_{l=1}^{L} a_{ij}^{(l)}\right. \right. , \tag{12}$$

where $W_a \in \mathbb{R}^{L \times L}$ is learnable parameters, and

$$a'_{ij} = ||_{l=1}^{L} a_{ij}^{\prime(l)}, a_{ij}^{\prime(l)} \in \mathbb{R}^{N \times N} \tag{13}$$

Next, the final output features of each node is calculate by the linear combination of new normalized attention coefficients a'_{ij}.

$$h_i^{\prime(l)} = \sigma\left(\sum_{j \in \mathcal{N}_{(i)}} a_{ij}^{\prime(l)} W^{(l)} \boldsymbol{h_j}\right) \tag{14}$$

where σ denotes nonlinearity function (e.g., ELU), and h'_i is the hidden representation of node i.

Then, the L attention heads have independent parameters. The outputs of each heads have two ways to merge: concatenation (15) and average (16). In this work, we applied average for the last layer and concatenation for others.

$$h'_i = \left|\left|_{l=1}^{L} \sigma\left(\sum_{j \in \mathcal{N}_{(i)}} a_{ij}^{(l)} W^{(l)} \boldsymbol{h}_j\right)\right. \right. \tag{15}$$

$$h'_i = \sigma\left(\frac{1}{L}\sum_{l=1}^{L}\sum_{j \in \mathcal{N}_{(i)}} a_{ij}^{(l)} W^{(l)} \boldsymbol{h}_j\right) \tag{16}$$

3.4.2 Gated temporal convolution layer

The Gated Temporal Convolution (GTCN) layer is designed to capture the complex and dynamic temporal correlations, following our previous work STGAT [57]. The structure is shown in Fig. 2. Each layer of the GTCN contains two convolution operations with the same hyperparameter settings. To ensure an appropriate temporal window to better utilize the long-term dependence of traffic signals, we set the dilation factor in the ith layer to $dilated(i) = 2^{(i-1)}$ (e.g., for the 1st layer, the dilation factor is $2^0 = 1$, and for the 2nd layer the dilation factor is $2^1 = 1$). The input of the GTCN layer is $\mathcal{X} \in \mathbb{R}^{N \times F \times C}$, where C is the number of kernels, the formula is shown in (17) to (18).

$$gated = \sigma(\mathcal{X} \times M + b) \tag{17}$$

$$\begin{aligned} GTCN(\mathcal{X}) = \; & gated \otimes (\mathcal{X} \times V + c) \\ & + (1 - gated) \otimes (\mathcal{X} \times U + d) \end{aligned} \tag{18}$$

where M, V, U, b, c, d are learnable parameters in convolution operation, $\otimes$ is the element-wise product, σ is the sigmoid function.

3.4.3 Fully convolutional networks

The last layer of the Spatial-Temporal model is a Fully Convolutional Network, namely EndConv. We follow [58] replaces fully connected layers with convolutions and apply a residual architecture. The formula is given in (19):

$$X_{emb}^{(l+1)} = \mathcal{F}(X_{emb}^{(l)}, \theta^{(l)}) + X_{emb}^{(l)} \tag{19}$$

where $\mathcal{F}$ is the residual function (in our work, it is "ReLU + Convolution"), and $\theta^{(l)}$ represents all the learnable parameters in the lth layer. This EndConv layer is applied to integrate information from the node embedding to output the prediction results (Table 1).

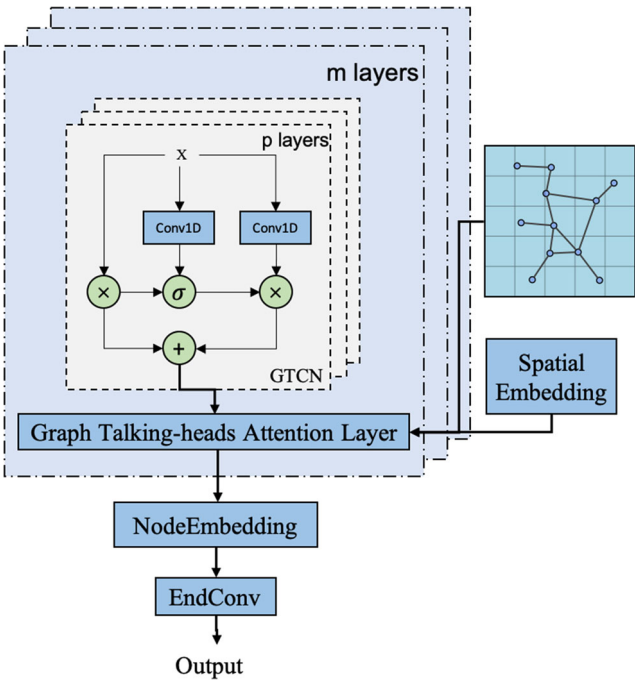


Fig. 2 The framework of Spatial-Temporal Model. It consists of m spatial-temporal layers and one Fully Convolutional Networks (EndConv) for output. The spatial-temporal layers consists of p gated temporal convolution layers (GTCN) and one graph talking-heads attention layer (GTHA)

4 Experiments

In this section, we compare our ASTGAT with SOTA methods on two public traffic datasets and design other experiments to help understanding the insight of our ASTGAT.

4.1 Datasets

We applied the same experiment procedures follow the work in DCRNN [10], verifying our ASTGAT on two publicly available traffic datasets (the details of the datasets are shown in Table 2) : (1) **METR-LA** includes 207 Los Angeles County highway traffic sensor speed observations, from March 1st, 2012 to June 30th, 2012 [59]. (2) **PEMS-BAY** is collected by the California Transportation Agencies Performance Measurement System (PeMS). We selected 325 data sensors in the Bay Area and collected 6 months of data from January 1st, 2017 to May 31st, 2017, following the work of DCRNN [10]. We visualize the sensor distributions in Fig. 3. For both datasets, the sensor readings are aggregated into a 5-minute time window. We also apply Z-Score normalization to the speed data in both datasets.

4.2 Data preprocessing

The data preprocessing procedure is designed as in our previous work on STGAT [57]. We construct an initial adjacency matrix according to road network distance with a threshold Gaussian kernel [54], and the formula is shown in (20).

$$\mathcal{W}_{ij} = \begin{cases} \exp\left(-\frac{\text{dist}(v_i, v_j)^2}{\sigma^2}\right) & , \text{dist}(v_i, v_j) > k \\ 0 & , \text{otherwise} \end{cases} \tag{20}$$

The threshold k in (20) assigned to 0.1 follow the previous works DCRNN [10] and GMAN [44]. Moreover, we divide datasets into three parts: 70% as training set, 20% as testing test, and 10% as validation set.

Table 1 Performance comparison of ASTGAT and baseline models

Data	Model	15 min			30 min			60 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PEMS-BAY	ARIMA	1.62	3.30	3.50%	2.33	4.76	5.40%	3.38	6.50	8.30%
	FC-LSTM	2.05	4.19	4.80%	2.20	4.55	5.20%	2.37	4.96	5.70%
	WaveNet	1.39	3.01	2.91%	1.83	4.21	4.16%	2.35	5.43	5.87%
	STGCN	1.36	2.96	2.90%	1.81	4.27	4.17%	2.49	5.69	5.79%
	DCRNN	1.38	2.95	2.90%	1.74	3.97	3.90%	2.07	4.74	4.90%
	GGRU	–	–	–	–	–	–	–	–	–
	Graph WaveNet	1.30	2.74	2.73%	1.63	3.70	3.67%	1.95	4.52	4.63%
	MRA-BGCN	1.29	2.72	2.9%	1.61	3.67	3.8%	1.91	4.46	4.6%
	FC-GAGA	1.36	2.86	2.87%	1.68	3.80	3.80%	1.97	4.52	4.67%
	GMAN	1.34	2.82	2.81%	1.62	3.72	3.62%	1.86	4.32	4.31%
	STGRAT	**1.29**	**2.71**	**2.67**%	1.61	3.69	3.63%	1.95	4.54	4.64%
	ASTGAT w/o Generator	1.32	2.77	2.76%	1.63	3.68	3.68%	1.87	4.32	4.43%
	ASTGAT w/o GTHA	1.31	2.79	2.73%	1.62	3.70	3.59%	1.86	4.33	4.39%
	ASTGAT w/o GTCN	1.34	2.90	2.82%	1.66	3.93	3.85%	1.93	4.52	4.61%
	ASTGAT	1.30	2.78	2.74%	**1.60**	**3.67**	**3.62**%	**1.85**	**4.31**	**4.30**%
METR-LA	ARIMA	3.99	8.21	9.60%	5.15	10.45	12.70%	6.90	13.23	17.40%
	FC-LSTM	3.44	6.30	9.60%	3.77	7.23	10.90%	4.37	8.69	13.20%
	WaveNet	2.99	5.89	8.04%	3.59	7.28	10.25%	4.45	8.93	13.62%
	STGCN	2.88	5.74	7.62%	3.47	7.24	9.57%	4.59	9.40	12.70%
	DCRNN	2.77	5.38	7.30%	3.15	6.45	8.80%	3.60	7.60	10.50%
	GGRU	2.71	5.24	6.99%	3.12	6.36	8.56%	3.64	7.65	10.62%
	ST-MetaNet	2.68	5.15	6.96%	3.09	6.28	8.43%	3.60	7.52	10.54%
	Graph WaveNet	2.69	5.15	6.90%	3.07	6.22	8.37%	3.53	7.37	10.01%
	MRA-BGCN	2.67	5.12	6.8%	3.06	6.17	8.3%	3.49	7.30	10.00%
	FC-GAGAT	2.75	5.34	7.25%	3.10	6.30	8.57%	3.51	7.31	10.14%
	GMAN	2.77	5.48	7.25%	3.07	6.34	8.35%	3.40	7.21	9.72
	STGRAT	**2.60**	**5.07**	**6.61**%	**3.01**	6.21	**8.15**%	3.49	7.42	10.01%
	ASTGAT w/o Generator	2.72	5.30	7.17%	3.08	6.32	8.58%	3.46	7.28	9.98%
	ASTGAT w/o GTHA	2.69	5.24	7.02%	3.05	6.19	8.21%	3.42	7.14	9.71%
	ASTGAT w/o GTCN	2.72	5.26	7.15%	3.09	6.36	8.53%	3.49	7.30	10.08%
	ASTGAT	2.69	5.17	6.86%	3.03	**6.13**	**8.15**%	**3.39**	**7.05**	**9.55**%

Bold entries indicate the best performance in experiments

4.3 Baselines

To evaluate the performance of our proposed ASTGAT, We chose the following methods as the baselines. All baselines experimental settings are based on previous literature.

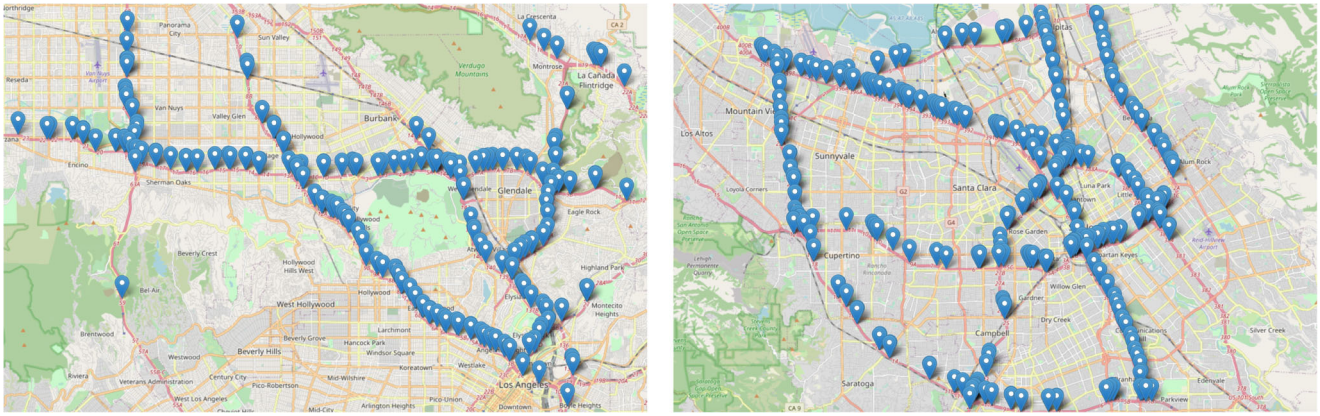
ARIMA [32], Autoregressive Integrated Moving Average model with Kalman filter. The orders are (3, 0, 1), and the model is implemented using the *statsmodel* python package.

FC-LSTM [60], The encoder-decoder framework uses Long Short-Term Memory (LSTM). Both the encoder and the decoder contain two recurrent layers with 256 LSTM units.

DCRNN [10], Diffusion Convolutional Recurrent Neural Network, models the traffic conditions as a diffusion process on a directed graph. Both encoder and decoder contain two recurrent layers with 64 units.

Table 2 Detailed dataset information of METR-LA and PEMS-BAY

Dataset	#Sensor(Nodes)	#Edges	#Time Steps
METR-LA	207	1515	34272
PEMS-BAY	325	2369	52116



(a) METR-LA (b) PEMS-BAY

Fig. 3 Sensor distribution of the **METR-LA** and **PEMS-BAY** dataset

GGRU [41], Gated Recurrent Unit Network applies a gated mechanism to control each attention head's influences. The hidden dimension is 64 and the number of attention heads is 4.

STGCN [12], Spatial-Temporal graph convolution network proposes a completely convolutional framework to forecasting future traffic conditions. The hidden dimension in ST-Block are 64, 16, and 64.

Graph WaveNet [13], Graph WaveNet designs an adaptive dependency matrix to capture the hidden spatial correlations in the data. They use stacked dilated 1D convolution like WaveNet to capture long-term traffic information. The hidden dimension is 32.

ST-MetaNet [11], ST-MetaNet proposes a deep-meta-learning based sequence-to-sequence model. It uses grid search over (16, 32, 64, 128) to choose the hidden dimension.

GMAN [44], GMAN designs an encoder-decoder architecture with attention mechanism to model the impact of the spatio-temporal correlations on traffic conditions. They use 8 attention heads and the hidden dimension is 64.

MRA-BGCN [43], MRA-BGCN builds the node-wise graph and edge-wise graph and uses bicomponent graph convolution handle interactions of both nodes and edges. The hidden dimension is 64.

STGRAT [45], ST-GRAT proposes an attention-based framework for traffic forecasting. The framework mainly includes spatial attention, temporal attention, and spatial sentinel vectors. It uses 4 attention heads, and the hidden dimension is 128.

FC-GAGA [61], FC-GAGA proposes a hard graph gate mechanism for traffic forecasting. The embedding dimension is set to 64 and the hidden layer for fully-connected layers is set to 128.

More specific implementation details can be found in the original paper and GitHub project.

4.4 Experimental details

We use grid search strategy and the validation set to find the suitable parameters and hyperparameters. All tests take 12 observed data points which each points aggregated 5 minutes readings. Then, forecasting traffic conditions in the next 3, 6, and 12 observed data points (15, 30, 60 and minutes) respectively.

4.4.1 Metrics

In the experiments, the MAE, RMSE, and MAPE are used to evaluate the results following [13]. Missing values are excluded in our experiments. In (21) to (23) we define the metrics, $\mathbf{y} = y_1, \ldots, y_n$ represents the ground truth, $\hat{\mathbf{y}} = \hat{y}_1, \ldots, \hat{y}_n$ represents the our model's prediction results, and Δ denotes the indices of the observed samples.

$$MAE(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{|\Delta|} \sum_{i \in \Delta} |y_i - \hat{y}_i| \tag{21}$$

$$RMSE(\mathbf{y}, \hat{\mathbf{y}}) = \sqrt{\frac{1}{|\Delta|} \sum_{i \in \Delta} (y_i - \hat{y}_i)^2} \tag{22}$$

$$MAPE(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{|\Delta|} \sum_{i \in \Delta} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \tag{23}$$

4.4.2 Hyperparameters

In the Network Generator, we set the generated process repeat $K = 6$ times and set $\lambda = 0.5$ to fuse the information from the initial and generated graphs. The dimension of initial spatial embedding $X_{init} = \mathbb{R}^{N \times F_{init}}$ obtained from node2vec is $F_{init} = 64$. The Spatial-Temporal Model consists of $m = 3$ spatial-temporal layers and one Fully Convolutional layer. Each spatial-temporal layer consist of $p = 3$ GTCN layer and one GTHA layer. The number of channels in each GTCN layer is $D_{GTCN} = 128$, the number of channels in each GTHA layer is $D_{GTHA} = 48$, and the number of channels in end convolution layers are $D_{conv} = 512$. We design GTHA with $L = 6$ attention heads. The merging method for GTHA in the last layer is averaging (16), and the other layers are concatenated (15).

We train our model using the Adam optimizer [62] to minimize the total graph regularization loss (7) and mean absolute error (MAE, (21)), and the formula is as follows:

$$loss = MAE(\mathbf{y}, \hat{\mathbf{y}}) + \mu(epoch)\mathcal{L}_{\mathcal{G}} \tag{24}$$

where $\mu(epoch)$ is a loss weighting function that increase with epoch, starting from 0, along a Gaussian curve during the first 100 training epochs. We set the initial learning rate to $lr = 0.0005$ and batch size to 36 for training on the **PEMS-BAY** dataset. We also set the initial learning rate to $lr = 0.005$ and batch size to 64 for training on the **METR-LA** dataset. For both datasets, we multiply the learning rate by 0.97 after each epoch. In addition, L2 normalization with a weight decay of $1e{-}9$ is applied for better generalizability.

4.5 Results

We experiment with our ASTGAT and baseline models for 15 minutes, 30 minutes, and 60 minutes ahead forecasting selected from the 12 forecasting horizons on both datasets. The overall performance of our ASTGAT and all baseline methods are illustrated in Table 1. We observe the following phenomena.

- Our ASTGAT achieves competitive performance on the three evaluation metrics in both datasets. The results show that our ASTGAT framework can be applied to various forecasting schemes.
- Our ASTGAT outperforms both the temporal methods (including ARIMA, FC-LSTM, and WaveNet) and spatial-temporal methods (including STGCN, Graph WaveNet, DCRNN, GGRU, MRA-BGCN and GMAN). The explanation is that our Spatial-Temporal Model and Network Generator argument the existing spatial and temporal dependencies from the historical traffic data.
- With respect to the latest MRA-BGCN, GMAN and STGRAT models, we can observe that ASTGAT outperforms in medium-term (30 minutes) and long-term (60 minutes) prediction. The explanation is that as a vehicle continuous moving, the traffic conditions are more affected by the distance nodes in a long time interval. Our ASTGAT uses network generator arguments the spatial information from the initial graph from the historical data to help us capture spatial dependencies more efficiently. Moreover, our GTCN layer with a gated mechanism can filter the redundant information in the historical sequence and is capable of handling long sequences, which means that effective long-term traffic prediction allows transportation agencies to have more time to take actions to optimize traffic.
- Compared with the baseline GMAN [44]. Experimental results indicate that our ASTGAT has better performance. Same as our model, GMAN also use Graph Attention Networks for Traffic Flow Forecasting. GMAN compute the attention scores among all vertices. Our proposed model is different from GMAN. We do not directly calculate the attention scores from all vertices. We generate a new weighted adjacency matrix with the information derived from the given historical traffic data and adjacency matrix. And we calculate the attention scores based on generated new weighted adjacency matrix. This better performance shows that the our proposed ASTGAT is more effective than GMAN.
- Since Los Angeles has complicated traffic conditions, all the methods perform better in the PEMS-BAY dataset than in the METR-LA dataset, and each method achieves a similar performance in the PEMS-BAY dataset, including ARIMA model. Our ASTGAT model is superior to all baseline methods in medium and long term forecasting. Moreover, our ASTGAT greatly reduces the prediction error of the data set METR-LA, which means that our ASTGAT can better handle more complex situations than the baseline method.

We also performed an ablation experiment to compare the contributions of GAT and GTHA layers in the model. The results are shown in Table 3. Our model with the proposed GTHA layer outperforms the original GAT model. The experimental results suggest that our ASTGAT with GTHA can capture the highly dynamic impact over nodes.

4.5.1 Ablation study

To investigate the effect of each component in our model, we evaluated three variants ASTGAT w/o GTHA (removing the talking-heads mechanism), ASTGAT w/o Generator (removing the Network Generator model), and ASTGAT

Table 3 Performance comparison of GTHA and GAT in model

Data	Model	15 min			30 min			60 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PEMS-BAY	ASTGAT with GAT	1.33	2.83	2.79%	1.69	3.81	3.68%	1.89	4.43	4.60%
	ASTGAT with GTHA	**1.30**	**2.78**	**2.74%**	**1.60**	**3.67**	**3.62%**	**1.85**	**4.31**	**4.30%**
METR-LA	ASTGAT with GAT	2.71	5.24	7.02%	3.09	6.28	8.41%	3.48	7.39	9.97%
	ASTGAT with GTHA	**2.69**	**5.17**	**6.86%**	**3.03**	**6.13**	**8.15%**	**3.39**	**7.05**	**9.55%**

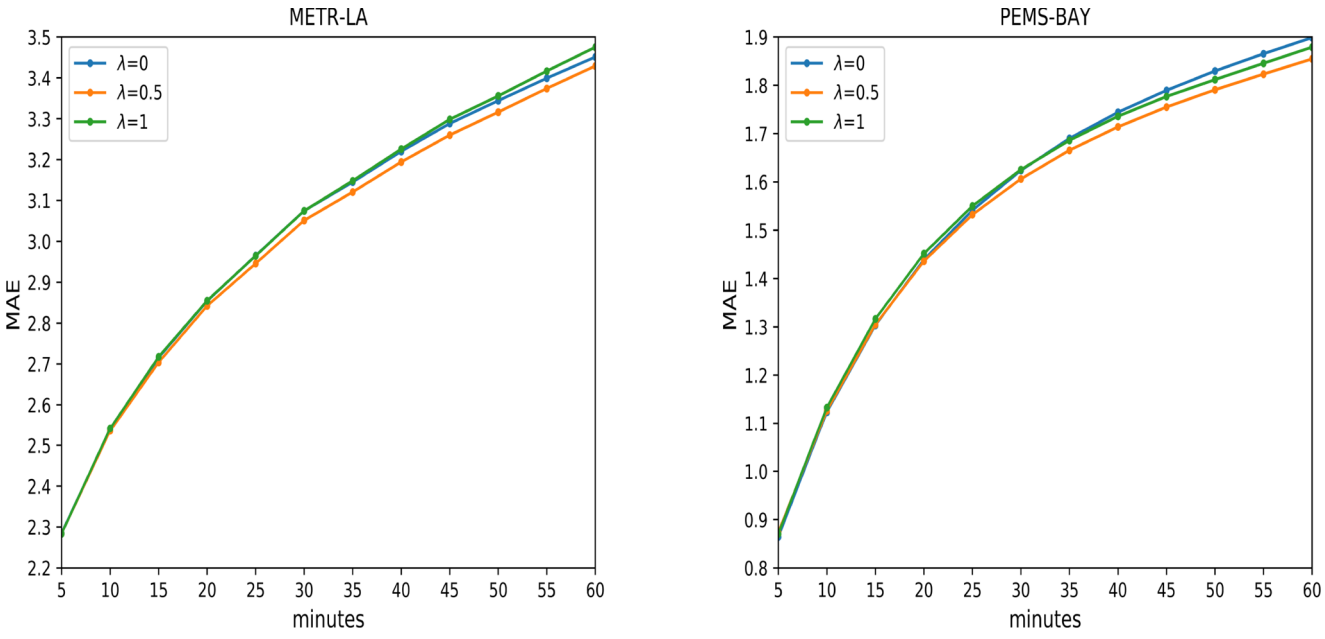
Bold entries indicate the best performance in experiments

w/o GTCN (removing the Gated Temporal Convolution Layer). In Table 1, compared with the full version of ASTGAT, both simplified versions of ASTGAT will cause performance degradation. This finding proves that several important components of the ASTGAT are effective.

In addition, many complex external factors affect traffic flow, and the proposed network generator can capture extra useful information from the observation traffic data to enhance the initial graph structure. We believe that historical traffic data imply more spatial information that cannot be represented in the initial weighted adjacency matrix. Future traffic conditions will be affected not only by neighbouring areas, but also by distant areas. Therefore, through the efforts of Network Generator model, the ASTGAT can adaptively augment the initial graph structure to use implicit information from the historical traffic data. Moreover, comparing ASTGAT with ASTGAT w/o Generator, the variant model led to more performance degradation. This finding means that a better graph structure can help us predict future traffic conditions much better.

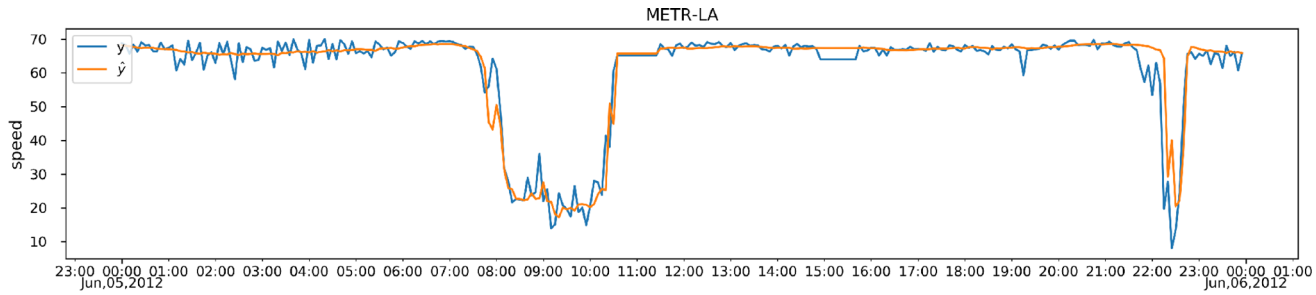
4.5.2 Effect of hyperparameter λ in network generator

To further verify the effectiveness of the hyperparameter λ in Network Generator, we evaluate ASTGAT with the following variants using different values in both datasets, e.g., $\lambda = 0$, $\lambda = 0.5$ and $\lambda = 1$. The experimental results are illustrated in Fig. 4. Figure 4a shows the experiment on the **METR-LA** dataset and Fig. 4b shows the experiment on the **PEMS-BAY** dataset. We observe that when $\lambda = 0.5$ the performance is the best. The possible explanation is that when $\lambda = 0.5$, we augment the initial graph with the generated graph structure which learned from the historical data. Moreover, $\lambda = 0$ means ignoring the generated graph structure and $\lambda = 1$ means ignoring the initial graph structure. The results of these two situations show that we

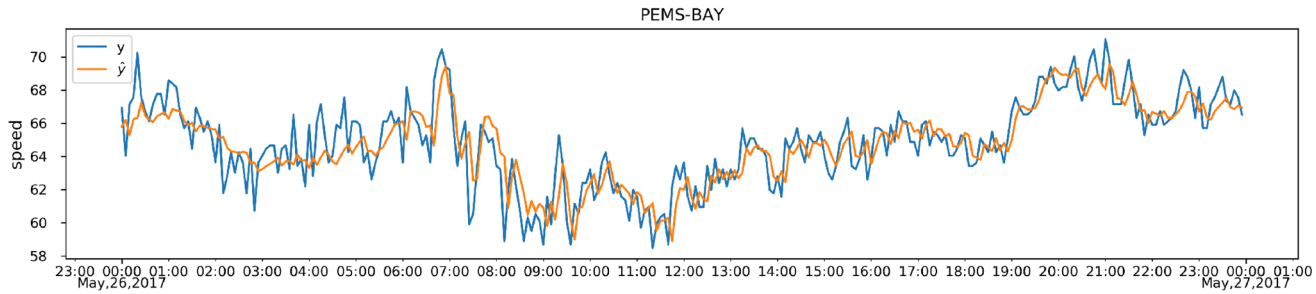


(a) METR-LA (b) PEMS-BAY

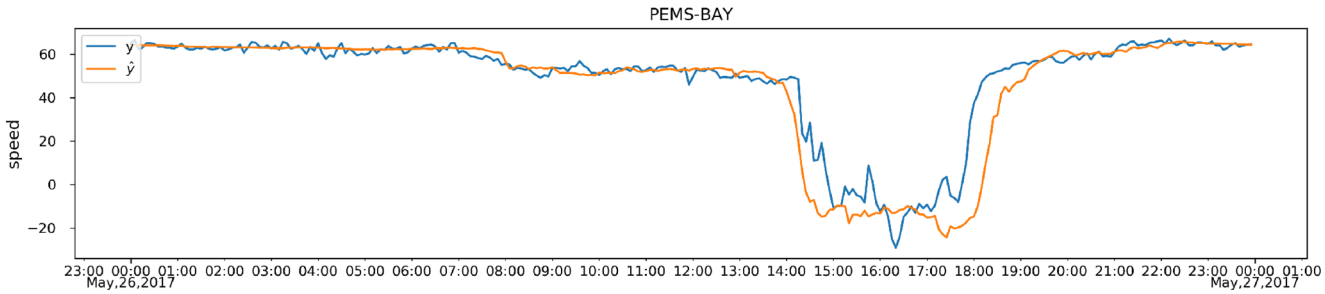
Fig. 4 Performance comparison of ASTGAT with different value of λ



(a) METR-LA 15 minutes ahead prediction.

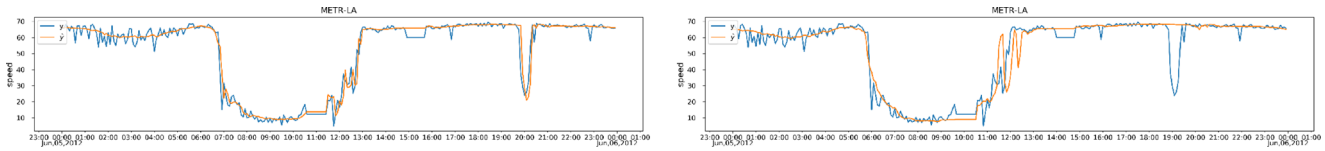


(b) PEMS-BAY 15 minutes ahead prediction.



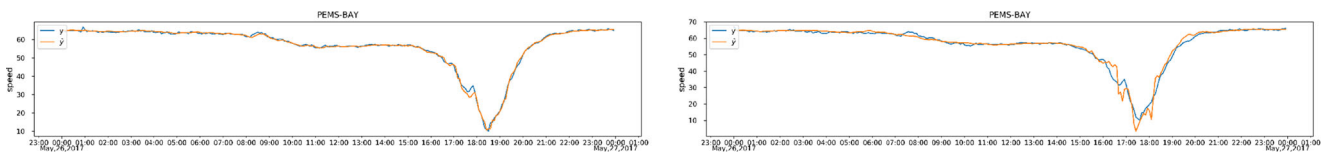
(c) PEMS-BAY 1 hour ahead prediction.

Fig. 5 Traffic time series forecasting visualization



(a) METR-LA 15 minutes ahead. (b) METR-LA 1 hour ahead.

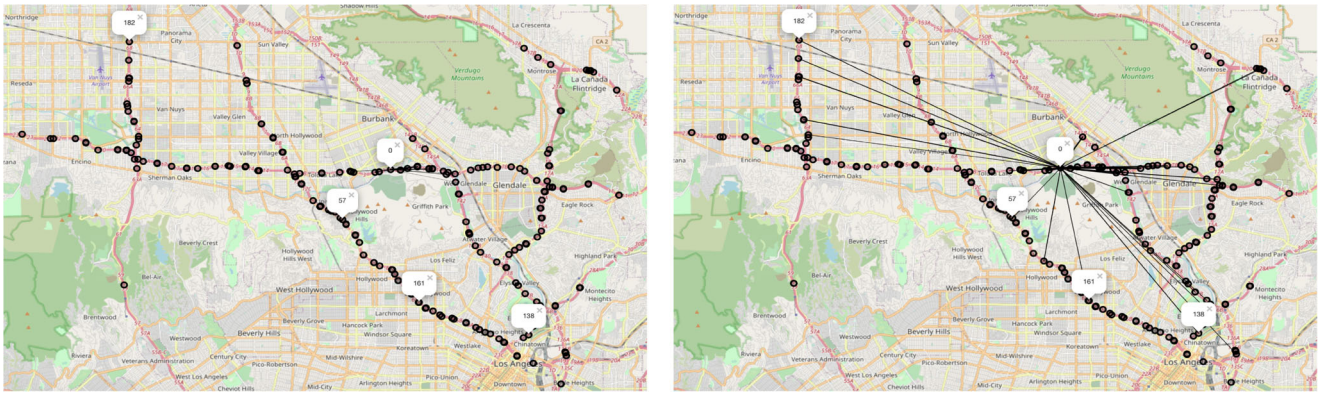
Fig. 6 Traffic time series forecasting visualization on dataset METR-LA for 15 minutes and 1 hour ahead prediction with the same node



(a) PEMS-BAY 15 minutes ahead. (b) PEMS-BAY 1 hour ahead.

Fig. 7 Traffic time series forecasting visualization on dataset PEMS-BAY for 15 minutes and 1 hour ahead prediction with the same node

Springer



(a) Fixed weighted adjacency matrix. (b) Generated weighted adjacency matrix.

Fig. 8 The visualization of fixed weighted adjacency matrix and generated weighted adjacency matrix. We select node 0 and its adjacent nodes as the study object

need to join the efforts of both the generated graph from the historical data and the initial graph construct based on prior knowledge.

4.6 Model interpretation

For a deeper understanding of the model, we visualize the

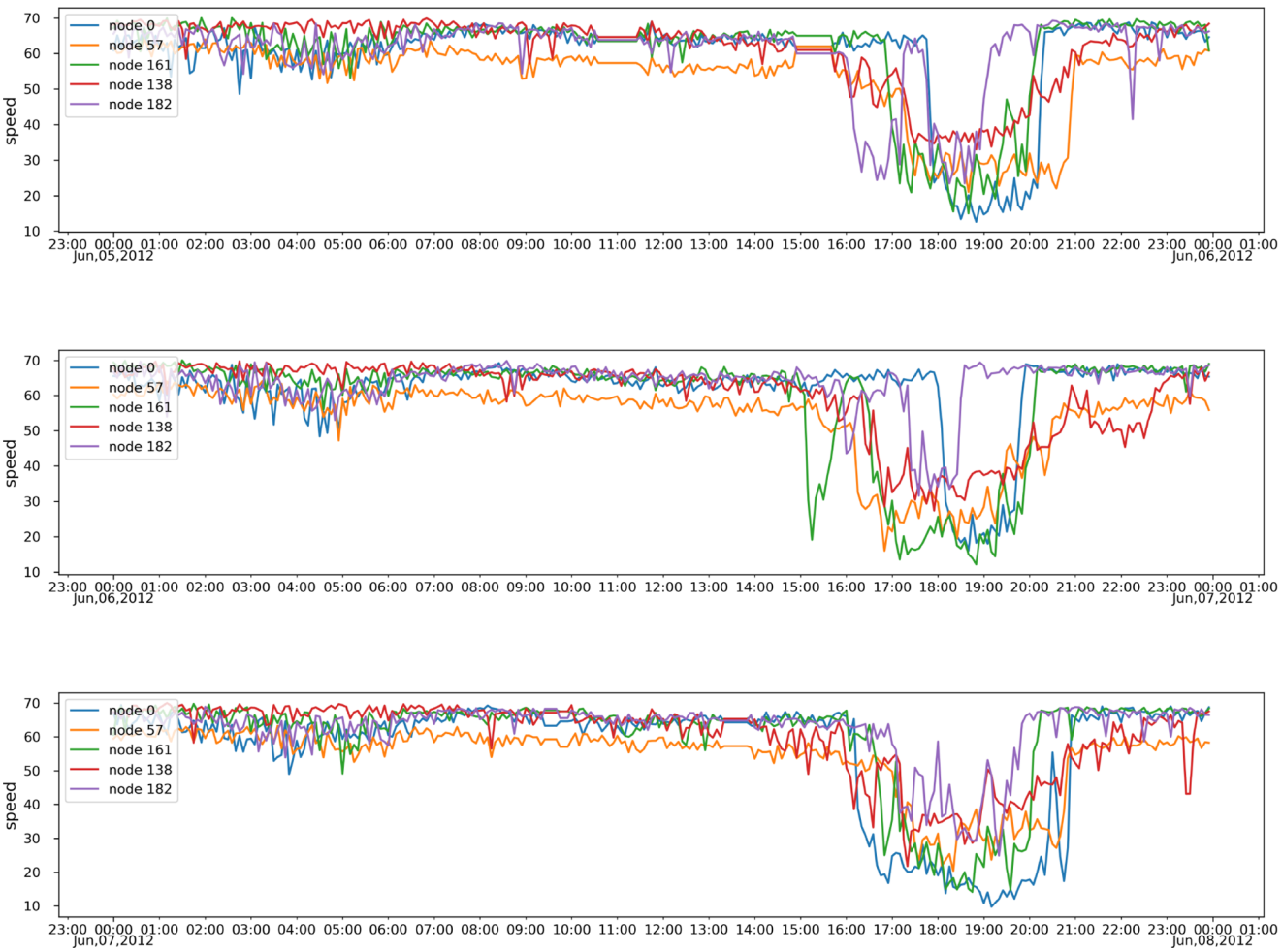


Fig. 9 The speed curves of nodes on the METR-LA dataset

forecasting results and the generated graph. Figure 5 shows the visualization results. Figure 6 shows the visualization of 15-minutes- and 1-hour-ahead forecasting with the same node on the METR-LA dataset. Figure 7 shows the visualization of 15-minutes- and 1-hour-ahead forecasting with the same node on the PEMS-BAY dataset. We make the following observations: (1) As shown in Fig. 5a and b, our ASTGAT generates a smooth prediction result close to the mean when slight oscillation exist. This is the reason that our model is weak in short-term prediction compared with the baseline model: STGRAT. (2) As shown in Fig. 5b and c, our ASTGAT predicts the start and the end of the peak hours. Especially in the Fig. 5c, ASTGAT with 1-hour-ahead forecasting predicted a peak in advance. The explanation is that our GTCN can handle long-range sequence with the gated mechanism and our GTHA is able to utilize the speed changes in neighbourhood sensors for more accurate forecasting. This is why our model outperforms the baseline models in long-term prediction. (3) The 15-minutes-ahead forecast is more sensitive for predicting abrupt changes in the traffic speed than the 1-hour-ahead forecast. For example, in Fig. 6a and b at approximately 20:00, there is a short-term peak, and only 15-minutes-ahead prediction forecasts this change. This is why short-term prediction has better performance than long-term prediction.

To further understand the Network Generator, we also visualize the fixed weighted adjacency matrix in Fig. 8a and generate the adjacency matrix in Fig. 8b. As shown in Fig. 8a, node 0 only connects to a few neighbouring nodes. Our Network Generator creates some new edges as shown in Fig. 8a, which means that our model finds a correlation between these nodes. To further study the correlation, we plot the line charts of speed during the day in Fig. 9. We found an interesting phenomenon: the curves of all nodes in the charts have the same trend. They all have a peak starting approximately 16:00 and ending approximately 21:00. These nodes show similar traffic patterns in the same period. In other words, our model captures that these distant nodes share a similar temporal patterns. This finding explains how our model achieves a better performance on long-term prediction. Our model can extract the similar temporal patterns between distant roads from the historical traffic data themselves. Combined with the given fixed weighted adjacency matrix, we can consider the information in both the temporal and spatial dimensions simultaneously. This is similar to the work [15] that constructs the road graph based on spatial information and learns it by comparing the similarity between time series for each road. However, our ASTGAT does not need to calculate the similarity and can completely capture this correlation from the historical traffic data.

5 Conclusion

We proposed a novel deep learning model framework for traffic forecasting, namely, an Adaptive Spatial-Temporal Graph Attention Networks (ASTGAT), to forecast future traffic conditions. Specifically, we propose a Network Generator that reconstructs a graph structure from historical traffic data that augments the initial graph structure (even when the initial graph structure is missing) and captures spatial-temporal correlations simultaneously. Moreover, we propose a talking-heads graph attention layer to better capture spatial dependencies. We further design a gating temporal convolution layer to handle long historical sequences. Experiments on two publicly available real-world traffic datasets show that the ASTGAT outperforms baseline models and can be applied to various forecasting scenarios. In future work, we plan to apply the ASTGAT model to update the weighted adjacency matrix based on the output of each layer in the model to further improve the adaptability of the generated graph structure. Moreover, we want to achieve a better performance in scenarios where a large number of sensors cannot work properly.

Acknowledgements The authors would like to thank the National Natural Science Foundation of China (61876017, 61876018, 61906014) for their support in this research.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. Lv Y, Duan Y, Kang W, Li Z, Wang FY (2014) Traffic flow prediction with big data: a deep learning approach. IEEE Trans Intell Transp Syst 16(2):865
2. Wang Y, Zhang D, Liu Y, Dai B, Lee LH (2019) Enhancing transportation systems via deep learning: A survey. Transp Res C Emerg Technol 99:144. https://doi.org/10.1016/j.trc.2018.12.004, http://www.sciencedirect.com/science/article/pii/S0968090X18304108
3. Yu H, Wu Z, Wang S, Wang Y, Ma X (2017) Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. Sensors 17(7):1501
4. Tan H, Wu Y, Shen B, Jin PJ, Ran B (2016) Short-term traffic prediction based on dynamic tensor completion. IEEE Trans Intell Transp Syst 17(8):2123
5. Ma X, Dai Z, He Z, Ma J, Wang Y, Wang Y (2017) Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors 17(4):818
6. Zhang J, Zheng Y, Qi D, Li R, Yi X, Li T (2018) Predicting citywide crowd flows using deep spatio-temporal residual networks. Artif Intell 259:147

7. Kipf TN, Welling M (2017) Semi-Supervised classification with graph convolutional networks. In: International conference on learning representations (ICLR)
8. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph attention networks. In: International conference on learning representations. https://openreview.net/forum?id=rJXMpikCZ
9. Seo Y, Defferrard M, Vandergheynst P, Bresson X (2018) Structured sequence modeling with graph convolutional recurrent networks. In: Cheng L, Leung ACS, Ozawa S (eds) Neural information processing. Springer International Publishing, Cham, pp 362-373
10. Li Y, Yu R, Shahabi C, Liu Y (2018) Diffusion convolutional recurrent neural network: data-driven traffic forecasting. https://openreview.net/forum?id=SJiHXGWAZ
11. Pan Z, Liang Y, Wang W, Yu Y, Zheng Y, Zhang J (2019) Urban traffic prediction from spatio-temporal data using deep meta learning. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, pp 1720–1730
12. Yu B, Yin H, Zhu Z (2018) Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. pp 3634–3640. http://dl.acm.org/citation.cfm?id=3304222.3304273
13. Wu Z, Pan S, Long G, Jiang J, Zhang C (2019) Graph WaveNet for deep spatial-temporal graph modeling. In: The 28th international joint conference on artificial intelligence (IJCAI) (International Joint Conferences on Artificial Intelligence Organization
14. Haklay M, Weber P (2008) Openstreetmap: User-generated street maps. IEEE Pervasive Comput 7(4):12
15. Yu B, Li M, Zhang J, Zhu Z (2019) 3d graph convolutional networks with temporal graphs: A spatial information free framework for traffic forecasting. arXiv:1903.00919
16. Li M, Zhu Z (2020) Spatial-temporal fusion graph neural networks for traffic flow forecasting. arXiv:2012.09641
17. Geng X, Li Y, Wang L, Zhang L, Yang Q, Ye J, Liu Y (2019) Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol. 33. pp 3656–3663
18. Chen Y, Wu L, Zaki M (2020) Iterative deep graph learning for graph neural networks: better and robust node embeddings. Adv Neural Inf Process Syst 33
19. Franceschi L, Niepert M, Pontil M, He X (2019) Learning discrete structures for graph neural networks. In: Chaudhuri K, Salakhutdinov R (eds) Proceedings of the 36th international conference on machine learning, proceedings of machine learning research, (PMLR), vol 97, pp 1972–1982. http://proceedings.mlr.press/v97/franceschi19a.html
20. Zhang Z, Zhao Y, Liu J, Wang S, Tao R, Xin R, Zhang J (2019) A general deep learning framework for network reconstruction and dynamics learning. Appl Netw Sci 4(1):1
21. Jang E, Gu S, Poole B. (2016) Categorical reparameterization with gumbel-softmax. arXiv:1611.01144
22. Shazeer N, Lan Z, Cheng Y, Ding N, Hou L (2020) Talking-heads attention. arXiv:2003.02436
23. Abbasi M, Shahraki A, Taherkordi A (2021) Deep learning for network traffic monitoring and analysis (ntma): a survey. Comput Commun
24. Lin K, Xu X, Gao H (2021) TSCRNN: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of IIoT. Comput Netw 190:107974
25. Gao H, Liu C, Li Y, Yang X (2020) V2VR: reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability. IEEE Trans Intell Transp Syst
26. Nishi T, Otaki K, Hayakawa K, Yoshimura T (2018) Traffic signal control based on reinforcement learning with graph convolutional neural nets. In: 2018 21st International conference on intelligent transportation systems (ITSC). IEEE, pp 877–883
27. Zhang Y, Zhou Y, Lu H, Fujita H (2021) Spark cloud-based parallel computing for traffic network flow predictive control using non-analytical predictive model. IEEE Trans Intell Transp Syst
28. Bi Z, Yu L, Gao H, Zhou P, Yao H (2020) Improved VGG model-based efficient traffic sign recognition for safe driving in 5G scenarios. Int J Mach Learn Cybern :1–12
29. Kuang L, Hua C, Wu J, Yin Y, Gao H (2020) Traffic volume prediction based on multi-sources GPS trajectory data by temporal convolutional network. Mobile Netw Appl 25(4):1405
30. Nagy AM, Simon V (2018) Survey on traffic prediction in smart cities. Pervasive Mob Comput : S1574119217306,521–
31. Ahmed MS, Cook AR (1979) Analysis of freeway traffic time-series data by using Box-Jenkins techniques. 722
32. Kamarianakis Y, Prastacos P (2003) Forecasting traffic flow conditions in an urban network: Comparison of multivariate and univariate approaches. Transp Res Rec 1857(1):74
33. Van Der Voort M, Dougherty M, Watson S (1996) Combining Kohonen maps with ARIMA time series models to forecast traffic flow. Transp Res C Emerg Technol 4(5):307
34. Min W, Wynter L (2011) Real-time road traffic prediction with spatio-temporal correlations. Transp Res C Emerg Technol 19(4):606
35. Chien SIJ, Kuchipudi CM (2003) Dynamic travel time prediction with real-time and historic data. J Transp Eng 129(6):608
36. Anand A, Ramadurai G, Vanajakshi L (2014) Data fusion-based traffic density estimation and prediction. J Intell Transp Syst 18(4):367
37. Nikovski D, Nishiuma N, Goto Y, Kumazawa H (2005) Univariate short-term prediction of road travel times. In: Proceedings. 2005 IEEE Intelligent transportation systems, 2005. IEEE, pp 1074–1079
38. Li S, Shen Z, Wang FY (2012) A weighted pattern recognition algorithm for short-term traffic flow forecasting. In: Proceedings of 2012 9th IEEE international conference on networking, sensing and control. IEEE, pp 1–6
39. Ma X, Tao Z, Wang Y, Yu H, Wang Y (2015) Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp Res C Emerg Technol 54:187
40. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv:1412.3555
41. Zhang J, Shi X, Xie J, Ma H, King I, Yeung D (2018) GaAN: Gated attention networks for learning on large and spatiotemporal graphs. arXiv:1803.07294
42. Guo S, Lin Y, Feng N, Song C, Wan H (2019) Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, p 922. https://doi.org/10.1609/aaai.v33i01.3301922, https://wvvw.aaai.org/ojs/index.php/AAAI/article/view/3881
43. Chen W, Chen L, Xie Y, Cao W, Gao Y, Feng X (2020) Multi-range attentive bicomponent graph convolutional network for traffic forecasting. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 3529–3536
44. Zheng C, Fan X, Wang C, Qi J (2020) GMAN: a graph multi-attention network for traffic prediction. In: AAAI, pp 1234–1241
45. Park C, Lee C, Bahng H, Tae Y, Jin S, Kim K, Ko S, Choo J (2020) ST-GRAT: a novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed. In: Proceedings of the 29th ACM international conference on information & knowledge management, pp 1215–1224

46. Gori M, Monfardini G, Scarselli F (2005) A new model for learning in graph domains. In: IEEE International joint conference on neural networks
47. Franco S, Marco G, Chung TAh, Markus H, Gabriele M (2009) The graph neural network model. IEEE Trans Neural Netw 20(1):61
48. Wu Z, Pan S, Chen F, Long G, Zhang C, Philip SY (2020) A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst
49. Bruna J, Zaremba W, Szlam A, LeCun Y (2013) Spectral networks and locally connected networks on graphs
50. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. In: Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R (eds) Advances in neural information processing systems, vol 29. Curran associates Inc., pp 3844–3852
51. Bahdanau D, Cho KH, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: 3rd international conference on learning representations, ICLR 2015
52. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł., Polosukhin I (2017) Attention is all you need. In: Advances in neural information processing systems, pp 5998–6008
53. Iida S, Kimura R, Cui H, Hung PH, Utsuro T, Nagata M (2019) Attention over heads: A multi-hop attention for neural machine translation. In: Proceedings of the 57th annual meeting of the association for computational linguistics: student research workshop, pp 217–222
54. Shuman DI, Narang SK, Frossard P, Ortega A, Vandergheynst P (2013) The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE Signal Process Mag 30(3):83
55. Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp 855–864
56. Nadarajah S, Kotz S (2004) The beta Gumbel distribution. Math Probl Eng 2004(4):323
57. Kong X, Xing W, Wei X, Bao P, Zhang J, Lu W (2020) STGAT: spatial-temporal graph attention networks for traffic flow forecasting. IEEE Access 8:134363
58. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3431–3440
59. Jagadish HV, Gehrke J, Labrinidis A, Papakonstantinou Y, Patel JM, Ramakrishnan R, Shahabi C (2014) Big data and its technical challenges. Commun Acm 57(7):86
60. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. Adv Neural Inf Process Syst :3104–3112
61. Oreshkin BN, Amini A, Coyle L, Coates MJ (2021) FC-GAGA: Fully connected gated graph architecture for spatio-temporal traffic forecasting. In: AAAI
62. Kingma DP, Ba J (2015) Adam: A Method for Stochastic Optimization. In: ICLR (Poster)

Xiangyuan Kong received the B.E. degree in software engineering from Beijing Jiaotong University, Beijing, China, in 2017. He is currently pursuing the Ph.D. degree with the School of Software Engineering, Beijing Jiaotong University. His research interests include machine learning and traffic flow forecasting in intelligent transportation systems.

Jian Zhang received the B.S degree in Information Science from China University of Mining and Technology, Xuzhou, China in 2012, the M.E degree in software engineering from Beijing Jiao Tong University, Beijing, China in 2015, and the Ph.D. degree from the Centre de Recherceh en Informatique, Signal et Automatique de Lille, Ecole Centrale de Lille, France in 2018. In 2019, he jointed the faculty of the School of Software Engineering, Beijing Jiao Tong University, where he is currently a lecturer. His main research interests include machine learning, deep learning and intelligent transportation systems.

Xiang Wei received his ph.D degree in Software Engineering from Beijing Jiaotong University in 2019. From 2019 to now, he is currently a lecturer with the School of Software Engineering, Beijing Jiaotong University. He has hold one search grants from National Science Foundation of China (NSFC). His research interest focuses on intelligent information processing, especially for semi-supervised deep learning and GANs.

Weiwei Xing received the B.S. degree in computer science and technology and the Ph.D degree in signal and information processing from Beijing Jiaotong University, Beijing, China, in 2001 and 2006, respectively. She is currently a Professor with the School of Software Engineering, Beijing Jiaotong University. Her research interests include intelligent information processing and machine learning.

Wei Lu received the B.S. degree in computer science from Fushun Petroleum Institute, Fushun, China, in 1985 and the M.S. degree in computer science and the Ph.D. degree in information and communication engineering from Sichuan University, Chengdu, China, in 1998 and 2006, respectively. He is currently a Professor with the School of Software Engineering, Beijing Jiaotong University, Beijing, China. His current research interests include computer networks and information systems, and multimedia information processing.