Expert Systems With Applications 227 (2023) 120259

Contents lists available at ScienceDirect

Expert Systems With Applications

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

DAGCRN: Graph convolutional recurrent network for traffic forecasting with dynamic adjacency matrix

Zheng Shi [a], Yingjun Zhang [a,*], Jingping Wang [b], Jiahu Qin [c], Xiaoqian Liu [a], Hui Yin [d], Hua Huang [d]

[a] School of Computer and Information Technology, Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing, 100044, China
[b] Information Technology Center, Beijing Jiaotong University, Beijing, 100044, China
[c] Department of Automation, University of Science and Technology of China, Hefei, 230027, China
[d] Key Laboratory of Beijing for Railway Engineering, Beijing Jiaotong University, Beijing, 100044, China

ARTICLE INFO

Keywords:
Traffic forecasting
Graph convolutional network
Dynamic adjacency matrix
Gated recurrent unit

ABSTRACT

Accurate and real-time traffic forecasting is of great significance for urban traffic planning, traffic control, and traffic management. However, the time-varying dynamic spatial relations and the complicated spatial–temporal dependencies are still open problems to be considered in traffic forecasting. To address these issues, we propose a graph convolutional recurrent network for traffic forecasting with a dynamic adjacency matrix within an encoder–decoder framework, named DAGCRN. The DAGCRN consists of a spatial relation extraction module (SREM), an adjacency matrix update module (AMUM), a dynamic graph convolutional recurrent module (DGCRM), and a global temporal attention module (GTAM). Specifically, SREM and AMUM are proposed to capture nodes' mutual relations at each time step and to model the evolution of the dynamic adjacency matrix, respectively. DGCRM captures the spatial–temporal dependencies of traffic data based on dynamic graph convolution and gated recurrent unit. GTAM is designed to extract the long-range temporal dependencies between future time steps and historical time steps. Extensive experiments on two real-world traffic speed datasets demonstrate that the proposed DAGCRN outperforms a number of representative baselines consistently.

1. Introduction

In the past decades, many countries have been devoted to establishing the intelligent transportation system (ITS) as a pivotal aspect of a smart city owing to the rapid advancement of urbanization and sensor technology (Guo et al., 2020; Yin et al., 2021b). As an important component of ITS, traffic forecasting seeks to predict future traffic conditions (*e.g.* traffic flow or traffic speed) (Wang et al., 2022) based on the historical observations (*e.g.* recorded via sensors or derived from surveillance videos) (Jiang & Luo, 2022). Up to now, traffic forecasting has gained a great deal of research interest as a cornerstone of such systems due to its widespread applications in daily urban transportation (Zheng et al., 2020). It serves to provide references for transportation schedules and traffic management by properly forecasting future traffic states, which helps to mitigate traffic congestion and enhance travel efficiency (Yin et al., 2021a). Furthermore, precise traffic forecasting assists citizens in route planning, which has a substantial impact on their quality of life.

As a typical spatial–temporal modeling problem, traffic forecasting has been widely investigated in terms of complicated spatial–temporal dependencies among nodes (Wang et al., 2022). Early studies focus on traffic forecasting at a single point or of a single lane (Guo & Yuan, 2020) based on some traditional models including Auto-Regression (VAR) (Lu et al., 2016), Auto-Regressive Integrated Moving Average (ARIMA) (Kumar & Vanajakshi, 2015), and Support Vector Regression (SVR) (Wu et al., 2004), etc. Note that these models mainly consider the temporal dependency while ignoring the spatial–temporal correlations among nodes, which may lead to poor forecasting performances.

Owing to the powerful data mining capabilities of artificial intelligence (AI) technologies (Elsheikh, 2022; Elsheikh, Shehabeldeen, et al., 2021; Khoshaim et al., 2021; Moustafa & Elsheikh, 2023), several advanced AI approaches have been successfully utilized to overcome the limitations of traditional traffic forecasting models (Jiang et al., 2021). Among these models, Convolutional Neural Networks (CNNs) and Graph Neural Networks (GNNs) are frequently employed because

* Corresponding author.
E-mail addresses: 20120404@bjtu.edu.cn (Z. Shi), zhangyj@bjtu.edu.cn (Y. Zhang), wangjp@bjtu.edu.cn (J. Wang), jhqin@ustc.edu.cn (J. Qin), 20112016@bjtu.edu.cn (X. Liu), hyin@bjtu.edu.cn (H. Yin), hhua@bjtu.edu.cn (H. Huang).

https://doi.org/10.1016/j.eswa.2023.120259
Received 1 October 2022; Received in revised form 12 April 2023; Accepted 23 April 2023
Available online 29 April 2023

of their excellent performance in feature extraction (Atwood & Towsley, 2016; Kipf & Welling, 2017). More specifically, CNN-based methods first convert the city map into regular grids and then apply convolution operation on neighboring grids to capture spatial dependency (Liu et al., 2020; Yao et al., 2019; Zhang et al., 2017). However, these methods ignore the natural topology structure of the traffic network and fail to tackle non-euclidean traffic data (Jiang & Luo, 2022). To sidestep such limitations, GNN-based methods have been further investigated. They model the traffic signals as graph-structured data, which is more suitable to represent the real traffic network. They utilize GNNs to extract spatial features of traffic data in the graph domain. In addition, temporal features of traffic data can be captured by recurrent neural network (RNN) (Bai et al., 2020; Li et al., 2018) and its variants (*e.g.* long short-term memory (LSTM) (Hochreiter & Schmidhuber, 1997) and gated recurrent unit (GRU) Cho et al., 2014) or temporal convolutional network (TCN) (Wu et al., 2019; Yu et al., 2018). As stated above, these deep learning methods have greatly promoted the development of traffic forecasting and achieved state-of-the-art performances. Despite these achievements, obtaining accurate traffic prediction results is still challenging due to the following two aspects.

Firstly, the dynamic spatial relations among nodes have not been fully investigated to date. The spatial dependencies have time-varying dynamics and capturing such dynamic variations is non-trivial (Liu et al., 2020). A majority of the existing GNN-based methods adopt a predefined distance-based adjacency matrix directly when modeling the spatial relations (Li et al., 2018; Yu et al., 2018). Unfortunately, such a predefined distance-based adjacency matrix may not reflect the genuine mutual relations between nodes properly (Bai et al., 2020). A center node in the distance-based graph primarily considers its nearby neighbors while ignoring some distant nodes, even if they share similar traffic patterns (Fang et al., 2019). In addition, the fixed adjacency matrix restricts the variations of dynamic spatial relations in the traffic network. As described in Yin et al. (2021a), the traffic status at the intersection exhibits strong correlations with the two adjacent roads under normal circumstances. When one of the roads experiences congestion due to traffic accidents, its relationship with the junction degrades. In such a situation, the other road's relationship with the junction rises. In fact, different locations have distinct impacts on a given node, and even the same location has varying effects over time (Guo et al., 2019).

Secondly, how to capture the complicated spatial–temporal dependencies of traffic network simultaneously is still an open problem. As proved in Guo et al. (2019), the traffic status on a road is subject to periodicity and trends of itself in general. For example, the change of traffic flow shows strong periodic patterns over days or even several weeks (Guo et al., 2019). In addition, the change of traffic status is dominated by the complex topological structure of traffic network (Jiang & Luo, 2022). Strictly speaking, these characteristics are with typical spatial–temporal dependencies to be fully considered when establishing a traffic forecasting model.

Concerning the challenges mentioned above, we propose a novel traffic forecasting model, named Dynamic Adjacency matrix Graph Convolutional Recurrent Network (DAGCRN) in this paper. Specifically, we first design a spatial relation extraction module (SREM) to model spatial relations among nodes at each time step. Then we propose an adjacency matrix update module (AMUM) to construct the dynamic adjacency matrix based on a gating mechanism and sparsely connected layers. We also present a dynamic graph convolutional recurrent module (DGCRM) that integrates the static and dynamic adjacency matrices together to capture the complicated spatial–temporal dependencies of traffic networks. Finally, we design a global temporal attention module (GTAM) to establish long-range temporal dependency. The main contributions of this work are summarized as follows.

- We propose a novel traffic forecasting model DAGCRN based on an encoder–decoder architecture, which captures the dynamic spatial relations and complicated spatial–temporal dependencies of traffic networks simultaneously.
- We propose a SREM together with an AMUM to model the dynamic spatial relations among nodes utilizing spatial attention, gating mechanism, and sparsely connected layers.
- We propose a DGCRM to capture the complicated spatial–temporal dependencies simultaneously based on dynamic graph convolution and GRU.
- Extensive experiments on two public real-world traffic datasets proves that our proposed DAGCRN not only reduces the prediction error significantly but also outperforms eight representative traffic forecasting baselines.

The remainder of this paper is organized as follows: The related work is briefly reviewed in Section 2. The traffic forecasting problem is formulated in Section 3. The details of the proposed model are introduced in Section 4. After that, we evaluate our model on two public datasets and then analyze the experimental results in Section 5. Finally, conclusions are supplied in Section 6.

2. Related works

2.1. Graph neural networks

Graphs are a kind of data structure that models a set of objects and their mutual relations (Zhou et al., 2020). GNN was first proposed to extend existing neural networks for processing irregular graph data in the non-Euclidean domain Scarselli et al. (2009). With the rapidly growing body of recent research interests in graph neural networks, GNNs have achieved excellent performance in several tasks such as node classification (Kipf & Welling, 2017) and link prediction (Zhang & Chen, 2018). According to the ways of information propagation and aggregation, GNNs can be divided into two mainstreams, *i.e.* , graph convolutional networks (GCNs) and graph attention networks (GATs).

GCNs aim to generalize standard convolution operations to the graph domain. Bruna et al. (2014) proposed a spectral network, in which the convolution operation is defined in the Fourier domain via computing the eigenvalue decomposition of the graph Laplacian matrix. But this network has high computational complexity. Defferrard et al. (2016) designed ChebNet to reduce the complexity by applying truncated expansion in terms of Chebyshev polynomials. After their pioneering work, Kipf and Welling (2017) proposed an extended GCN to further reduce computation, which limited the order of layer-wise convolution operation and approximated the largest eigenvalue of Laplacian matrix. GCN finally generalized the graph convolution operation with a renormalization trick to avoid numerical instabilities and exploding/vanishing gradients. Atwood and Towsley (2016) proposed a diffusion convolutional neural network (DCNN) to utilize transition matrices to model the diffusion process across each node. Hamilton et al. (2017) designed GraphSAGE to generate node embeddings by sampling and aggregating features from a fixed-size set of neighbors of node.

Meanwhile, a number of attention mechanisms have been successfully utilized in many natural language processing (NLP) tasks like machine translation (Vaswani et al., 2017). Veličković et al. (2018) proposed GAT, which takes an attention mechanism into consideration when determining the weights of neighbors. Besides GAT, Zhang et al. (2018) proposed GaAN to introduce an attention mechanism for updating the nodes' hidden states. The difference lies in that GaAN uses a gated self-attention approach to gather information from different heads rather than the average pooling operation in GAT.

All in all, GNNs utilize different aggregators to gather information from each node's neighbors and design specific mechanisms to update nodes' hidden states (Zhou et al., 2020). Up to now, GNNs have been applied to many traffic forecasting tasks to exploit the complicated spatial relations of real-world traffic networks due to the powerful abilities of handling graph data.

2.2. Deep learning-based traffic forecasting methods

With the rise of artificial intelligence, machine learning methods are widely used in many fields by learning data to improve model performance for various tasks such as prediction (Elsheikh, 2023; Khoshaim et al., 2022) and investigation (Elsheikh et al., 2023). Among them, deep learning has recently demonstrated its remarkable feature extraction and aggregation capabilities in the modeling of different engineering systems (Elsheikh, Muthuramalingam, et al., 2021). The result is that deep learning-based traffic forecasting methods have been thoroughly investigated from different perspectives (Jiang & Luo, 2022; Yin et al., 2021a). In terms of the complicated spatial–temporal dependencies of traffic data, deep learning-based traffic forecasting approaches primarily focus on two aspects: spatial modeling and temporal modeling.

Deep learning-based traffic forecasting methods for spatial dependency can be broadly divided into two categories: CNN-based methods and GNN-based methods. Generally, CNN-based forecasting methods first divide a city into a grid map based on the longitude and latitude where a grid represents a region, and then they apply CNNs to extract the spatial correlation between different regions for traffic prediction (Liu et al., 2020; Yao et al., 2019; Zhang et al., 2017). Notice that CNN-based approaches are limited to model Euclidean data due to their low spatial resolution. Concerning the limitations of CNN-based forecasting methods, GNN-based methods were proposed to model the non-Euclidean traffic topological network as a graph and adopt GNNs to capture the spatial dependency of traffic data (Li et al., 2018; Yu et al., 2018; Zhang et al., 2018). To be more specific, most GNN-based methods perform graph convolution operations directly on a predefined distance-based graph (Li et al., 2018; Yu et al., 2018) while ignoring the dynamic spatial relations among nodes. Some methods design different self-adaptive mechanisms to uncover latent graph structures from training data without any prior knowledge (Bai et al., 2020, 2019; Diao et al., 2019; Wu et al., 2020, 2019). Another branch of GNN-based methods utilize the predefined graph as a mask to adjust the dynamic graphs generated by some spatial attention mechanisms (Guo et al., 2019; Li, Wang, Zhang, & Wu, 2021; Li et al., 2022). Additionally, GATs are also adopted to filter away adjacent but irrelevant nodes and attend to distant but related nodes (Park et al., 2020; Yin et al., 2021b; Zhang et al., 2018), contributing to significant enhancements in spatial dependency modeling.

In terms of handling temporal dependency, earlier studies apply traditional statistic-based time series methods, like VAR (Lu et al., 2016), ARIMA (Kumar & Vanajakshi, 2015), and SVR (Wu et al., 2004), to predict future traffic data. Compared with these traditional time series methods, deep learning-based methods show more superiority in tackling complicated temporal dependency due to their powerful representation abilities. Deep learning-based traffic forecasting methods for temporal dependency can be broadly divided into three categories: CNN-based methods, RNN-based methods, and attention-based methods. The representative CNN-based methods employ TCN along the temporal dimension for temporal dependency modeling (Guo et al., 2019; Wang et al., 2022; Wu et al., 2019; Zhao et al., 2022). However, such an implicit temporal modeling approach makes each time step invisible, boosting efficiency at the expense of flexibility. Regarding RNN-based methods, LSTM (Lu et al., 2020; Shi et al., 2021; Yin et al., 2021b) and GRU (Bai et al., 2020; Li et al., 2018; Zhang et al., 2021) are mainly used to capture long-range temporal dependency. Despite the fact that RNN-based methods are able to design elaborate mechanisms to update hidden node state incorporated with GNN modules, they still fail to adequately describe the dynamics of traffic data and suffer from error accumulation in the long-term prediction process (Wang et al., 2023). Beyond that, attention-based methods adopt the self-attention mechanism of Transformer (Vaswani et al., 2017) to establish the temporal relationships between each time step directly (Cai et al., 2020; Park et al., 2020; Reza et al., 2022; Wang et al., 2020; Zheng et al., 2020).

To summarize, the aforementioned methods have highlighted the significance of capturing both spatial and temporal features for traffic forecasting. However, most existing models overlook the dynamics of traffic data and are not capable of effectively capturing the complex spatial–temporal dependencies (Shao et al., 2022). Despite the impressive mechanisms proposed by some works to model dynamic spatial relations (Guo et al., 2019; Park et al., 2020; Zhao et al., 2022), they only take data from a single time step into account when computing spatial relations, failing to utilize contextual information. In order to address these issues, we present a novel traffic forecasting model DAGCRN in this paper.

3. Problem formulation

Traffic forecasting is a classical spatial–temporal prediction problem that aims to predict the future traffic features, *i.e.* traffic flow or speed, by leveraging previously observed traffic data. Generally, a traffic network with N nodes can be represented as a directed graph $G = (V, E, A)$, where $V = \{v_1, \dots, v_N\}$ is a set of nodes and each node represents a traffic sensor deployed by the roadside. E is a set of edges which stands for the spatial connectivity of these nodes. $A \in \mathbb{R}^{N\times N}$ is a weighted adjacency matrix representing the spatial relationship strength between nodes. At time step t, we define the traffic data of ith node as $x_t^i \in \mathbb{R}^D$, where D is the feature dimension. The collection of all nodes' features is regarded as a graph signal, which is defined as $X_t = [x_t^1, \dots, x_t^N]^T \in \mathbb{R}^{N\times D}$.

With the aforementioned notations, the traffic forecasting problem can be formally as mentioned below. Given the predefined graph $G = (V, E, A)$ and the historical P time steps observed graph signals $X_{1:P} = [X_1, \dots, X_P]^T \in \mathbb{R}^{P\times N\times D}$, we aim to learn a map function $F_\Theta(\cdot)$, which takes G and $X_{1:P}$ as inputs and forecasts the graph signals for the next Q time steps $\hat{X}_{(P+1):(P+Q)} = [\hat{X}_{P+1}, \dots, \hat{X}_{P+Q}]^T \in \mathbb{R}^{Q\times N\times D}$. The whole problem can be represented as follows:

$$[X_{1:P}, G] \xrightarrow{F_\Theta(\cdot)} \hat{X}_{(P+1):(P+Q)} \quad (1)$$

where Θ are all the parameters to be learned in the model.

4. The proposed model

We propose a novel deep model for traffic forecasting based on an encoder–decoder architecture, named DAGCRN. Figs. 1 and 2 show the framework of DAGCRN and a single DAGCRN cell, respectively.

In encoding stage, the encoder randomly generates DA_0 and H_0 as the initial dynamic adjacency matrix and the initial hidden node state. The encoder takes the historical observed traffic data $X_{1:P} = [X_1, \dots, X_P]^T$ as input and encodes the input data step by step to capture the spatial–temporal dependencies based on its recurrent structure. The outputs of the encoder contain the hidden states $[H_1, \dots, H_P]^T$ of each input time step and the generated dynamic adjacency matrix DA_P.

In decoding stage, the decoder takes DA_P and H_P as the initial dynamic adjacency matrix and hidden states, respectively. Additionally, the decoder takes an all-zero vector as the "GO" symbol to start the decoding process, which is a common practice in encoder–decoder models. The decoder generates the output data to perform multi-step forecasting in an auto-regressive manner, which means the output of the current time step $\hat{X}_t$ will be used as the input of the next time step $t+1$.

DAGCRN consists of four primary modules: SREM, AMUM, DGCRM, and GTAM. SREM is employed to model the spatial relations among nodes at each time step based on a static adjacency matrix. AMUM aims to fully leverage the dynamic contextual information of consecutive time steps and generate a dynamic adjacency matrix based on a gating mechanism and sparsely connected layers. DGCRM not only integrates

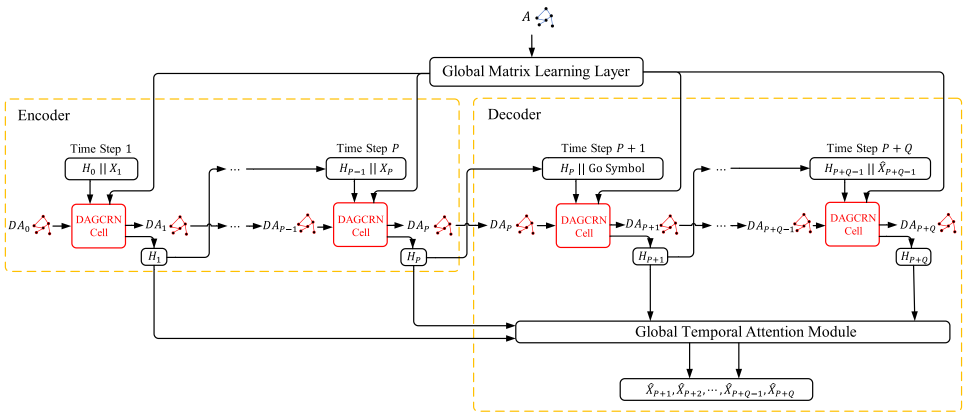


Fig. 1. The overall framework of DAGCRN.

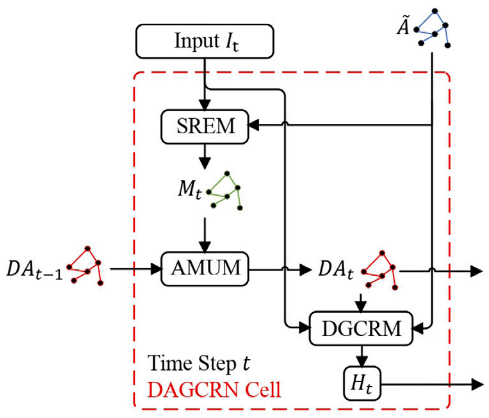


Fig. 2. The framework of a single DAGCRN cell.

both the dynamic and static adjacency matrices, but also handles information flow over spatially dependent nodes. GTAM is designed to capture temporal dependencies from a global perspective by directly modeling long-range temporal relationships between future time steps and historical time steps. The detailed implementation mechanisms of each module will be described in the following subsections.

4.1. Spatial relation extraction module

Before introducing the implementation of SREM, we first present the generation process of the static adjacency matrix, which is the input of DAGCRN cell. Since traffic conditions on the traffic network are complex and spatially dependent, an intuitive idea is to construct a static adjacency matrix based on pairwise road network distance between sensor nodes using a thresholded Gaussian kernel (Li et al., 2018), as illustrated below.

$$A_{v_i,v_j} = \begin{cases} \exp\left(-\frac{d_{v_i,v_j}}{\sigma^2}\right), & v_i \neq v_j,\ \ d_{v_i,v_j} \le \kappa \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where d_{v_i,v_j} is the road network distance from node v_i to v_j, σ is the standard deviation of distance, and κ is the threshold for sparsity which is assigned to 0.1. However, such a distance-based adjacency matrix may not reflect the genuine static spatial dependency of the traffic network. On the one hand, a center node can only attend to the information from its neighboring nodes while ignoring the information from remote nodes with similar traffic patterns (Fang et al., 2019). On the other hand, it is far from adequate and precise to solely take the distance information to represent inter-node spatial relations. In fact, the population density, the vehicle density, the road conditions and some factors may affect traffic conditions (Wang et al., 2020). Furthermore, the fixed adjacency matrix restricts the variations of dynamic spatial relations in the traffic network.

Inspired by Guo et al. (2020), we define the static adjacency matrix by a global matrix learning layer to discover the latent static spatial relations adaptively as follows,

$$\tilde{A} = A + I_N + A_{par} \tag{3}$$

where $\tilde{A}$ is the distance-based adjacency matrix supplemented by an identity matrix I_N for self-loop and a learnable parameterized matrix A_{par}. After being modified by all training samples with A_{par}, $\tilde{A}$ can adequately match the static spatial relations of the traffic network and address the aforementioned issues in a relatively simple way.

At each time step, we first concatenate the current traffic data input X_t and the hidden node state $H_{t-1} \in \mathbb{R}^{N\times d_{model}}$ from the previous time step:

$$Z_t = X_t \ \ \| \ \ H_{t-1} \tag{4}$$

where $Z_t \in \mathbb{R}^{N\times(D+d_{model})}$, $\|$ represents the concatenation operation, D is the feature dimension of input X_t, and d_{model} is the feature dimension of hidden node state H_{t-1}. Z_t is regarded as dynamic input, which will be fed into a global diffusion GCN to extract static traffic features:

$$(\tilde{D}_O)_{i,i} = \sum_j \tilde{A}_{i,j} \tag{5}$$

$$(\tilde{D}_I)_{i,i} = \sum_j \tilde{A}^T_{i,j} \tag{6}$$

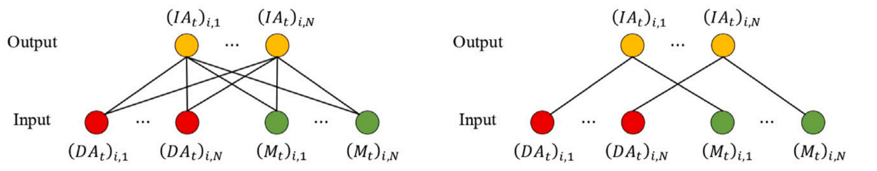
$$F_t = \sigma(\tilde{D}_O^{-1}\tilde{A}Z_tW_O + \tilde{D}_I^{-1}\tilde{A}^TZ_tW_I) \tag{7}$$

where $\tilde{D}_O$ and $\tilde{D}_I$ denote the two degree matrices of outflow and inflow, $\tilde{D}_O^{-1}\tilde{A}$ and $\tilde{D}_I^{-1}\tilde{A}^T$ denotes the transition matrices of the diffusion process and its reverse one, respectively. W_I and W_O are two parameter matrices of outflow and inflow. $\sigma(\cdot)$ is an activation function. Here we adopt the bidirectional diffusion to capture the influence from both the upstream and downstream traffic (Li et al., 2018). The output $F_t \in \mathbb{R}^{N\times d_{model}}$ denotes the nodes features of traffic network, which contains the static spatial information at time step t.

To further model the nodes' mutual spatial relations, we adopt the attention mechanism (Vaswani et al., 2017) to derive the current adjacency matrix at time step t as follows:

$$M_t = \frac{(F^tW^1)(F^tW^2)^T}{\sqrt{d_{model}}} \tag{8}$$

where $W^1, W^2 \in \mathbb{R}^{d_{model}\times d_{model}}$ are two linear transformation matrices and $(\cdot)^T$ denotes the tensor transpose operation. In this paper, we adopt inner product to quantify the strength of spatial relations between nodes. In fact, multi-head attention mechanism is with a top priority



(a) Illustration of a fully connected layer (b) Illustration of a sparsely connected layer

Fig. 3. An example of two methods of matrix feature transformation. $(\cdot)_{i,j}$ denotes the spatial relation between node i and node j of corresponding adjacency matrix.

since it can lead to the creation of richer representations, which in turn allows for increased performance on machine learning tasks (Vaswani et al., 2017).

We can obtain h adjacency matrices M_t^i $(i = 1, \dots, h)$:

$$M_t^i = \frac{(F^t W_i^1)(F^t W_i^2)^T}{\sqrt{d_{model}/h}} \tag{9}$$

where $W_i^1, W_i^2 \in \mathbb{R}^{d_{model} \times d_{model}/h}$ are two linear transformation matrices of the ith head and $M_t^i \in \mathbb{R}^{N \times N}$ is the output of the ith head. To aggregate spatial information from different sub-spaces adaptively, we apply average pooling to obtain the final current adjacency matrix $M_t \in \mathbb{R}^{N \times N}$ as follows:

$$(M_t)_{i,j} = \frac{1}{h} \sum_{k=1}^{h} (M_t^k)_{i,j} \tag{10}$$

4.2. Adjacency matrix update module

The core idea of DAGCRN is the update of the dynamic adjacency matrix based on the gating mechanism of GRU, which models the dynamic evolution process of spatial relations. At time step t, the inputs to the adjacency matrix update module include the dynamic adjacency matrix DA_{t-1} of the previous time step and the current adjacency matrix A_t from SREM. We first concatenate the two adjacency matrices $\left[DA_t \parallel M_t\right] \in \mathbb{R}^{N \times 2N}$, where $2N$ denotes the matrix feature dimension. The most common way to perform matrix transformation and derive the output adjacency matrix $IA_t \in \mathbb{R}^{N \times N}$ is through a fully connected layer (Guo et al., 2020) whose parameter size is $\mathbb{R}^{2N \times N}$. The calculation process of fully connected layer is illustrated in Fig. 3(a). However, such a fully connected layer has two limitations: (1) The value of $(IA_t)_{i,j}$ is the linear combination of $(DA_t)_{i,\cdot}$ and $(M_t)_{i,\cdot}$, that is, $(IA_t)_{i,j}$ not only counts on $(DA_t)_{i,j}$ and $(M_t)_{i,j}$ but also attends to other inter-node relations, which will bring noise especially in the circumstance that no spatial correlations exist. (2) A fully connected layer requires a total of $2N^2$ parameters, which is computationally intensive. However, it is difficult to optimize the entire model when N is quite large.

To address the above two limitations, we propose a sparsely connected layer for matrix transformation, depicted in Fig. 3(b). The value of the intermediate adjacency matrix $(IA_t)_{i,j}$ is a linear combination of $(DA_t)_{i,j}$ and $(M_t)_{i,j}$, avoiding the noise of other inter-node relations. Also notice that the sparsely connected layer only requires $2N$ parameters due to its sparsity. The process of the sparsely connected layer can be defined as follows:

$$IA_t = W_{sc} \left[DA_t \parallel M_t\right] \tag{11}$$

where W_{sc} denotes the learnable parameters of the sparsely connected layer and $IA_t \in \mathbb{R}^{N \times N}$ is the output of sparse connected layer.

The gating mechanism applied in GRU has shown its powerful ability to control information flow along the temporal dimension (Cho et al., 2014; Wang et al., 2022). We use the current and historical spatial information to generate the dynamic spatial adjacency matrix DA_t via a modified GRU, where the original fully connected layers are

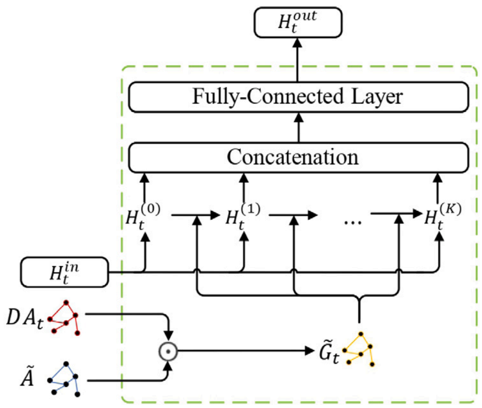


Fig. 4. The process of dynamic graph convolution.

substituted with the sparsely connected layers. The calculation process is denoted by:

$$z_t^A = \sigma(W_{sc}^z \left[DA_t \parallel M_t\right]) \tag{12}$$

$$r_t^A = \sigma(W_{sc}^r \left[DA_t \parallel M_t\right]) \tag{13}$$

$$\tilde{H}_t^A = \tanh(W_{sc}^h \left[r_t^A \odot DA_{t-1} \parallel M_t\right]) \tag{14}$$

$$DA_t = (1 - z_t^A) \odot DA_{t-1} + z_t^A \odot \tilde{H}_t^A \tag{15}$$

where $\odot$ represents Hadamard product, W_{sc}^z, W_{sc}^r, W_{sc}^h are parameters for the corresponding sparsely connected layers, z_t^A, r_t^A are update gate and reset gate, respectively, $\tilde{H}_t^A$ is the GRU cell state, and $DA_t \in \mathbb{R}^{N \times N}$ is the derived dynamic adjacency matrix, which contains contextual information and will be used in the establishment of dynamic graph convolutional recurrent module. All the intermediate variables in AMUM are identified by the superscript $(\cdot)^A$.

To summarize, we combine GRU's recurrent structure with the generation of the dynamic adjacency matrix by adding the previous time step's hidden node state to SREM's input and updating the dynamic adjacency matrix in AMUM. By this way, the dynamic spatial adjacency matrix becomes more effective and meaningful due to the graph convolution and spatial attention in SREM, which fuses a center node's information with its neighbors' information and captures mutual spatial relations between nodes. Furthermore, we can fully explore the dynamic spatial relations by employing AMUM to obtain a dynamic representation of the traffic network, which serves as a complement to the static adjacency matrix.

4.3. Dynamic graph convolutional recurrent module

We first introduce dynamic graph convolution, which aims to transform and fuse information between nodes and their neighbors to express dynamic spatial dependencies in the traffic network. The static adjacency matrix $\tilde{A}$ and the dynamic adjacency matrix DA^t reflect spatial interactions among nodes from different perspectives. In this module, we incorporate both of them to exploit the static and dynamic spatial properties of traffic data. We present the implementation process of dynamic graph convolution as illustrated in Fig. 4. Specifically, we first utilize DA_t to adjust $\tilde{A}$ dynamically with an element-wise product operation as follows:

$$\tilde{G}_t = DA_t \odot \tilde{A} \tag{16}$$

$$(\tilde{E}_{tO})_{i,i} = \sum_j (\tilde{G}_t)_{i,j} \tag{17}$$

$$(\tilde{E}_{tI})_{i,i} = \sum_j (\tilde{G}_t^T)_{i,j} \tag{18}$$

where $\tilde{G}_t$ denotes the fused dynamic adjacency matrix, $\tilde{E}_{tO}$ and $\tilde{E}_{tI}$ denotes the two degree matrices of outflow and inflow.

Then we perform graph convolution in a two-stage manner: propagation stage and mix stage. The propagation stage aims to fuse a node's information with its deeper neighbors' in a recursive manner. We define the largest depth of propagation as K. The k^{th} $(k = 1, \ldots, K)$ hop propagation process can be described as follows:

$$\tilde{H}_t^{(k)} = \tilde{E}_{tO}^{-1}\tilde{G}_t H_t^{(k-1)} W_O^{(k)} + \tilde{E}_{tI}^{-1}\tilde{G}_t^T H_t^{(k-1)} W_I^{(k)} \tag{19}$$

$$H_t^{(k)} = \mathrm{ReLU}\left(\left(\alpha H_t^{in} + (1-\alpha)\tilde{H}_t^{(k)}\right) W_P^{(k)}\right) \tag{20}$$

$$H_t^{(0)} = H_t^{in} \tag{21}$$

where $\tilde{E}_{tO}^{-1}\tilde{G}_t$ and $\tilde{E}_{tI}^{-1}\tilde{G}_t^T$ denotes the transition matrices of the diffusion process and its reverse process, $W_O^{(k)}$ and $W_I^{(k)}$ are two parameter matrices of outflow and inflow, $W_P^{(k)}$ is the parameter matrix of kth hop propagation, α is a hyper-parameter to retain a proportion of node original information so that the propagated state can both preserve locality and explore a deeper neighborhood (Wu et al., 2020), and H_t^{in} represents the input node state of dynamic graph convolution.

The mix stage is introduced to aggregate results produced at different hops and filter out irrelevant information as well. We first concatenate the multi-hop node states along the feature dimension and then apply a fully connected layer to select useful information. The process is defined as follows:

$$H_t^{out} = \left(\|_{k=0}^{K} H_t^{(k)}\right) W^O \tag{22}$$

where H_t^{out} represents the output node state of dynamic graph convolution and W^O is a learnable feature transformation matrix. In fact, W^O functions as a feature selector. Under the extreme circumstance that no spatial dependencies exist, it is still able to retain the original node state by setting the corresponding weights to 0. The whole process of the aforementioned dynamic graph convolution can be shortened as:

$$H_t^{out} = \Theta_{\star G}(H_t^{in}) \tag{23}$$

where $\star G$ and Θ denote the dynamic graph convolution and its learnable parameters.

To capture spatial and temporal dependencies simultaneously, we replace the matrix multiplications in GRU with the dynamic graph convolution following previous study (Bai et al., 2020; Li, Feng, et al., 2021; Li et al., 2018) to develop DGCRM as follows:

$$z_t^F = \sigma(\Theta_{\star G}^z \left[H_{t-1} \;\|\; X_t\right]) \tag{24}$$

$$r_t^F = \sigma(\Theta_{\star G}^r \left[H_{t-1} \;\|\; X_t\right]) \tag{25}$$

$$\tilde{H}_t^F = \tanh(\Theta_{\star G}^h \left[r_t^F \odot H_{t-1} \;\|\; X_t\right]) \tag{26}$$

$$H_t = (1 - z_t^F) \odot H_{t-1} + z_t^F \odot \tilde{H}_t^F \tag{27}$$

where $\Theta^z, \Theta^r, \Theta^h$ are parameters for the corresponding dynamic graph convolution layers, z_t^F and r_t^F respectively represent the update gate and the reset gate, $\tilde{H}_t^F$ is the candidate GRU cell state, and $H_t \in \mathbb{R}^{N \times d_{model}}$ is the output hidden node state at time step t. All the intermediate variables in DGCRM are identified by the superscript $(\cdot)^F$. We also add a residual shortcut path to speed up the convergence rate.

4.4. Global temporal attention module

To alleviate the error propagation effect in long-term forecasting and further explore global temporal relationships, we utilize a global temporal attention to directly model the relations between future time steps and historical time steps (Guo et al., 2021).

At each decoding time step j, we first calculate the attention coefficients between decoder's output node state $H_j^{De} \in \mathbb{R}^{N \times d_{model}}$ and the encoded node states $H^{En} = \left[H_1, \ldots, H_P\right]^T \in \mathbb{R}^{P \times N \times d_{model}}$ from encoder as follows:

$$E_j = \sigma\left((H_j^{De}U_1)U_2((H^{En})^T U_3)^T + b_e\right) V_e \tag{28}$$

where $U_1 \in \mathbb{R}^{d_{model} \times 1}$, $U_2 \in \mathbb{R}^{N \times d_{model}}$, $U_3 \in \mathbb{R}^{N \times 1}$, $b_e \in \mathbb{R}^{1 \times P}$, and $V_e \in \mathbb{R}^{P \times P}$ are all parameters to be learned. The global temporal attention mechanism can select relevant historical time steps adaptively.

Then we use a softmax function to guarantee that the sum of attention weights equals to 1,

$$Z_j^i = \frac{\exp(E_j^i)}{\sum_{p=1}^{P} \exp(E_j^p)} \tag{29}$$

where $Z_j = \left[Z_j^1, \ldots, Z_j^P\right]^T \in \mathbb{R}^{P \times 1}$ is a normalized vector, which describes the relative importance of different historical time steps to decoding time step j. Based on the normalized attention weights and the encoded node states, we get the context vector $C_j \in \mathbb{R}^{N \times d_{model}}$ as the weighted sum of H^{En}:

$$C_j = \sum_{i=1}^{P} Z_j^i H_i^{En} \tag{30}$$

Finally, we concatenate the decoder's output node state H_j^{De} and the context vector C^j together and feed it into a fully-connected layer to generate the final prediction results:

$$\hat{X}_j = \left(H_j^{D_e} \;\|\; C_j\right) W^R \tag{31}$$

where $W^R \in \mathbb{R}^{2d_{model} \times D}$ denotes the learnable parameters for the final feature transformation.

L1 loss is used as the loss function to train the whole model in an end-to-end manner:

$$\mathcal{L} = \frac{1}{Q}\sum_{i=1}^{Q} \left|\hat{X}_{P+i} - X_{P+i}\right| \tag{32}$$

where P is the length of historical time steps and Q is the length of future time steps. $\hat{X}_{P+i}$ is the prediction result of all nodes' traffic data at time step $P+i$ and X_{P+i} is the ground truth of all nodes' traffic data at time step $P+i$.

5. Experiments and result analysis

5.1. Data

We conduct experiments on two real-world traffic datasets:

- METR-LA (Li et al., 2018): This dataset contains traffic speed information collected from the highway of Los Angeles with 207 sensors. The time period of this dataset ranges from Mar 1st 2012 to Jun 30th 2012. The samples are aggregated to 5 min. The unit of speed is mile/h.
- PEMS-BAY (Li et al., 2018): This dataset contains traffic speed information collected from the Bay Area by California Transportation Agencies with 325 sensors. The time period of this dataset ranges from Jan 1st 2017 to May 31th 2017. The samples are aggregated to 5 min. The unit of speed is mile/h.

Following previous work (Li et al., 2018), we use the first 70% of data as train set, the last 20% of data as test set, and the remaining 10% of data as validation set in chronological order. We use the mean and standard deviation of the train set to apply Z-Score normalization to the whole dataset. For both datasets, we regard each sensor as a node in the graph. The pre-defined adjacency matrix A is constructed based on pair-wise road network distance between sensors with thresholded Gaussian kernel as described in Eq. (2).

5.2. Parameters settings

In our experiments, We repeat the same experiments 5 times and record the average value of metrics. The inputs of the model include time of day and the normalized traffic speed, while the output of the model is the predicted traffic speed. The length of the input sequence $P = 12$ and the size of the predicting window $Q = 12$, that is, we use the past 1 hour historical data to predict the next 1 hour traffic speed. We set the $d_{model} = 64$ and $d_{model} = 80$ for METR-LA and PEMS-BAY, respectively. The number of multi-head is set to 4. The largest propagation depth K in dynamic graph convolution is set to 2. The batch size is set to 64 for both datasets. The hyper-parameter α in Eq. (20) for preserving original information is set to 0.05. Adam optimizer is utilized for training with learning rate 0.001. Early stopping is employed to avoid overfitting. Furthermore, we utilize the scheduled sampling strategy to bridge the gap between the training stage and the inference stage (Bengio et al., 2015).

5.3. Baselines

We compare the proposed DAGCRN with several traditional statistic-based methods and GNN-based methods for traffic forecasting tasks.

- VAR (Lu et al., 2016): Vector Auto-Regression uses an auto-regressive component to model multiple time series.
- ARIMA (Kumar & Vanajakshi, 2015): Autoregressive Integrated Moving Average is a classical method for time series prediction, which integrates auto-regression with moving average model.
- FC-LSTM (Sutskever et al., 2014): Fully-Connected LSTM network is a variant of RNN, which learns the spatial dependency of sequence data with fully connected hidden units.
- DCRNN (Li et al., 2018): Diffusion Convolutional Recurrent Neural Network integrates diffusion graph convolution with GRU to predict traffic data.
- ASTGCN (Guo et al., 2019): Attention Based Spatial–Temporal Graph Convolutional Networks designs spatial and temporal attention mechanisms to model dynamics of traffic data.
- STSGCN (Song et al., 2020): Spatial–Temporal Synchronous Graph Convolutional Network proposes a novel multi-module mechanism to capture localized spatial–temporal heterogeneity simultaneously.
- AGCRN (Bai et al., 2020): Adaptive Graph Convolutional Recurrent Network utilizes a node adaptive parameter learning module to enhance GCN and combines it with GRU to capture spatial–temporal dependencies of traffic data.
- MTGNN (Wu et al., 2020): It is a deep learning model for multivariate time series forecasting, which employs adaptive graph, GNN, and dilated inception layers to capture spatial–temporal dependencies.

The performances of all methods are evaluated by three commonly used metrics, including (1) mean absolute error (MAE), which reflects the actual situation of prediction accuracy, (2) root mean squared error (RMSE), which is more sensitive to abnormal values, and (3) mean absolute percentage error (MAPE), which eliminates the influence of data units. These metrics are defined as follows:

$$\mathrm{MAE}(\hat{X}, X) = \frac{1}{|\Omega|} \sum_{i \in \Omega} \left| \hat{X}_i - X_i \right| \tag{33}$$

$$\mathrm{RMSE}(\hat{X}, X) = \sqrt{\frac{1}{|\Omega|} \sum_{i \in \Omega} \left(\hat{X}_i - X_i \right)^2} \tag{34}$$

$$\mathrm{MAPE}(\hat{X}, X) = \frac{1}{|\Omega|} \sum_{i \in \Omega} \left| \frac{\hat{X}_i - X_i}{X_i} \right| \tag{35}$$

where $\hat{X}$ denotes the prediction results after inverse normalization, X is the ground truth, and Ω represents the set of temporal indices.

5.4. Overall prediction performance results

Table 1 shows the multi-step forecasting results on the two datasets generated by our proposed DAGCRN and other baselines. Horizon refers to the time interval between the current time step and the predicted time step. Specifically, the results are composed of the performance comparisons for 15 min (horizon 3), 30 min (horizon 6), and 1 hour (horizon 12) ahead forecasting. By comparison with traditional statistical methods (VAR and ARIMA), deep-learning based method (FC-LSTM), and other GNN-based methods (DCRNN, ASTGCN, STSGCN, AGCRN, and MTGNN), the evaluation metrics (MAE, RMSE, and MAPE) of DAGCRN reduce at least (1) {2.28%, 1.93%, 2.92%} for horizon 3, {1.31%, 1.13%, 1.59%} for horizon 6, and {1.43%, 0.83%, 1.22%} for horizon 12 on the METR-LA dataset, (2) {3.03%, 2.87%, 3.25%} for horizon 3 and {2.42%, 1.60%, 1.36%} for horizon 6 on the PEMS-BAY dataset. The prediction performance results show that DAGCRN can generally outperform other baselines on the two datasets for most forecasting tasks except for certain metrics in the long-range horizon (*e.g.* 1 hour ahead) on the PEMS-BAY dataset. The remarkable improvements indicate the effectiveness of modeling dynamic spatial relations. The employment of SREM and AMUM can dynamically adjust the spatial adjacency matrix based on historical information, which is crucial for performance improvement. We also find that the difficulty of traffic prediction task increases with the increase of horizon. DAGCRN only gets marginal improvements for horizon 12 on the PEMS-BAY dataset due to error accumulation in the long-term prediction, but it still achieves on-par performance by comparison with other baselines.

The limitations of other baselines can be summarized as below. Traditional statistical time series prediction methods perform worse than deep learning methods because they fail to model the complex spatial–temporal dependencies of non-linear traffic data. FC-LSTM performances better than traditional statistical methods due to its ability of capturing complicated temporal dependency. But it ignores the mutual influence between nodes, which restraints the overall prediction performances. Graph-based methods further enhance the forecasting performances significantly due to the consideration of traffic network topology. However, DCRNN only counts on the given distance-based adjacency matrix, which restricts its forecasting performance heavily. ASTGCN employs spatial attention to capture spatial dependencies dynamically, but it depends on the whole input sequence which fails to precisely capture the dynamics between adjacent time steps. STSGCN designs a spatial–temporal graph convolutional module to capture the localized spatial–temporal dependencies synchronously, but breaks down when employed on the METR-LA dataset due to the missing values and the restricted representation ability of the fixed graph structure. AGCRN benefits a lot from the self-adaptive adjacency matrix generated from node embedding. However, once the training stage is complete, the self-adaptive adjacency matrix remains static in the inference stage, making it ineffective in capturing dynamic spatial dependency. MTGNN shows outstanding forecasting performances owing to its well-designed graph learning layer and dilated inception layer, but it fails to model the spatial dynamics between consecutive time steps. To sum up, most baselines neglect the dynamic properties of the traffic network, which degrades their representation abilities. The results demonstrate that the proposed DAGCRN can effectively model the dynamic spatial relations of traffic data and capture the complicated spatial–temporal dependencies.

Table 1
Multi-step forecasting performance comparison on the two traffic speed datasets (The best performances are in bold type).

Dataset	Methods	Horizon 3			Horizon 6			Horizon 12		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
METR-LA	VAR	4.42	7.89	10.20%	5.41	9.13	12.70%	6.52	10.11	15.80%
	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%
	DCRNN	2.77	5.38	7.30%	3.15	6.45	8.80%	3.60	7.60	10.50%
	ASTGCN	4.86	9.27	9.21%	5.43	10.61	10.13%	6.51	12.52	11.64%
	STSGCN	3.31	7.62	8.06%	4.13	9.77	10.29%	5.06	11.66	12.91%
	AGCRN	2.87	5.58	7.70%	3.23	6.58	9.00%	3.62	7.51	10.38%
	MTGNN	2.69	5.18	6.86%	3.05	6.17	8.19%	3.49	7.23	9.87%
	DAGCRN(ours)	**2.63**	**5.08**	**6.67%**	**3.01**	**6.10**	**8.06%**	**3.44**	**7.17**	**9.75%**
PEMS-BAY	VAR	1.74	3.16	3.60%	2.32	4.25	5.00%	2.93	5.44	6.50%
	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%
	DCRNN	1.38	2.95	2.90%	1.74	3.97	3.90%	2.07	4.74	4.90%
	ASTGCN	1.52	3.13	3.22%	2.01	4.27	4.48%	2.61	5.42	6.00%
	STSGCN	1.44	3.01	3.04%	1.83	4.18	4.17%	2.26	5.21	5.40%
	AGCRN	1.37	2.87	2.94%	1.69	3.85	3.87%	1.96	4.54	4.64%
	MTGNN	1.32	2.79	2.77%	1.65	3.74	3.69%	1.94	**4.49**	**4.53%**
	DAGCRN(ours)	**1.28**	**2.71**	**2.68%**	**1.61**	**3.68**	**3.64%**	**1.92**	4.51	4.56%

Table 2
Ablation study: effects of different components on the PEMS-BAY dataset.

Dataset	Methods	Horizon 3			Horizon 6			Horizon 12		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
PEMS-BAY	DAGCRN w/o A_{par}	1.39	2.95	2.97%	1.79	4.13	4.08%	2.16	4.88	4.90%
	DAGCRN w/o DA_t	1.40	2.91	2.93%	1.73	3.90	3.87%	2.12	4.86	4.93%
	DAGCRN w/o AMUM	1.37	2.88	2.90%	1.71	3.85	3.81%	2.07	4.82	4.79%
	DAGCRN w/o Mix	1.41	2.89	2.96%	1.74	3.84	3.86%	2.07	4.69	4.78%
	DAGCRN w/o GC	1.41	3.02	2.95%	1.83	4.18	4.12%	2.28	5.30	5.54%
	DAGCRN w/o GTAM	1.35	2.86	2.84%	1.70	3.91	3.82%	2.10	4.92	4.81%
	DAGCRN w/o directed	1.33	2.82	2.79%	1.68	3.80	3.76%	2.01	4.65	4.73%
	DAGCRN(ours)	**1.28**	**2.71**	**2.68%**	**1.61**	**3.68**	**3.64%**	**1.92**	**4.51**	**4.56%**

5.5. Ablation study

In this subsection, we conduct two groups of ablation studies on the PEMS-BAY dataset to analyze the effects of different components in DAGCRN and the effectiveness of the proposed sparse connections separately.

5.5.1. Effects of different components

We first conduct an ablation study to validate the impacts of different key components in DAGCRN. We name DAGCRN without different components as follows:

- DAGCRN w/o A_{par}: DAGCRN without learnable parameterized matrix A_{par} in Eq. (3).
- DAGCRN w/o DA_t: DAGCRN without dynamic adjacency matrix. That is, both SREM and AMUM are removed, and graph convolution is employed based on the static adjacency matrix A purely.
- DAGCRN w/o AMUM: DAGCRN without adjacency matrix update module. Dynamic graph convolution is performed based on the fusion of M_t and A.
- DAGCRN w/o Mix: DAGCRN without the mix stage in dynamic graph convolution. The result of the last hop is taken as the output of the dynamic graph convolution.
- DAGCRN w/o GC: DAGCRN without the graph convolution operation. We replace all the graph convolution layers with linear transformations.
- DAGCRN w/o GTAM: DAGCRN without GTAM.
- DAGCRN w/o directed: DAGCRN with undirected graph.

Table 2 shows the experimental results of different DAGCRN variants. From the results, it can be observed that: (1) A_{par} significantly improves the performance as it can discover the latent spatial relations between nodes besides the road distance information in a self-adaptive manner. (2) DAGCRN's forecasting performance declines rapidly when the dynamic adjacency matrix DA_t is removed, which implies that the consideration of time-varying spatial adjacency relations is indispensable. (3) The effect of the elaborate AMUM is also evident as it can model the evolution process of dynamic spatial relations with the shift of time. (4) The proposed mix stage in dynamic graph convolution is of great significance as it can aggregate and select useful information from each propagation hop. (5) The ablation study on graph convolution indicates that though these nodes are isolated geographically but interdependent with each other. The employment of graph convolution enables information flow among interdependent nodes, thus improving the model's performance. (6) In terms of short-term forecasting, GTAM has a minor impact, but it has a significant impact on long-term forecasting. (7) The ablation study on directed graph indicates that considering the direction between paired nodes can improve the prediction performance. The undirected graph ignores that the influence between nodes is different, leading to inadequate modeling of spatial dependency. The result indicates the effectiveness of GTAM in alleviating the error propagation effect in long-term forecasting and exploring global temporal relationships.

5.5.2. Effects of sparse connections

We conduct the second group of ablation study to verify the effectiveness of the proposed sparse connections, which are used in AMUM for matrix feature transformation. We compare the proposed DAGCRN with another variant, named DAGCRN-fully, which keeps the model structure unchanged except that the sparsely connected layers for matrix feature transformation are replaced by the fully connected layers. We present the overall forecasting performances of three metrics, computation time and the number of parameters in Table 3.

It can be observed that the employment of fully connected layers degrades the performance of model owing to noise from other unrelated

Table 3
Ablation study: effects of sparse connections on the PEMS-BAY dataset.

Dataset	Methods	Overall metrics			Computation time		# Parameters
		MAE	RMSE	MAPE	Training(s/epoch)	Inference(s)	
PEMS-BAY	DAGCRN	1.56	3.68	3.61%	225.13	20.83	409950
	DAGCRN-fully	1.65	3.82	3.78%	302.62	29.68	1675500

inter-node relations. Meanwhile, the fully connected layers require more computation costs and parameters. In contrast, the proposed sparse connections can effectively transform matrix dimension while avoiding the noise of irrelevant inter-node relations. In summary, the application of sparse connections improves the model's forecasting performance and reduces its complexity.

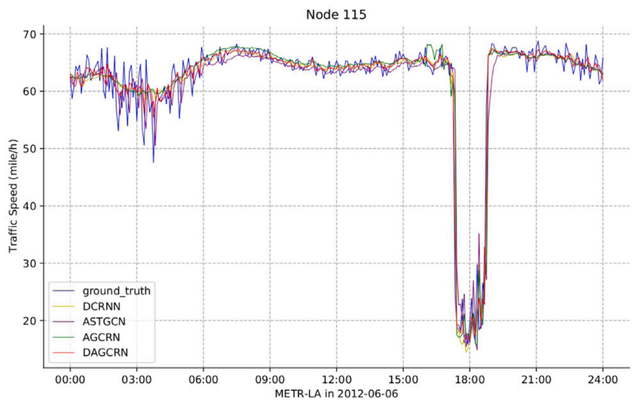
5.6. Case study

We randomly select one day (2012-06-06) and one sensor (Node 115) from METR-LA dataset and plot the time series of ground truth and prediction results as depicted in Fig. 5(a), where the x-axis and y-axis represent time and speed (mph), respectively. To make the chart clearer, we only plot the prediction results of DCRNN, ASTGCN, AGCRN, and the proposed DAGCRN instead of all the methods listed in Table 1. Through Fig. 5(a), we can observe that: (1) Compared with the daytime, violent fluctuations in traffic speed occurred more frequently at night. In comparison to the three baselines, the proposed DAGCRN could better capture the dynamic change in traffic speed. (2) This road was congested from 17:15 and the congestion was alleviated at approximately 18:50. Though all of the four models could learn the peak and valley trend, time lag could still be observed on the three baselines due to their insensitivity to the dynamic change in traffic speed.

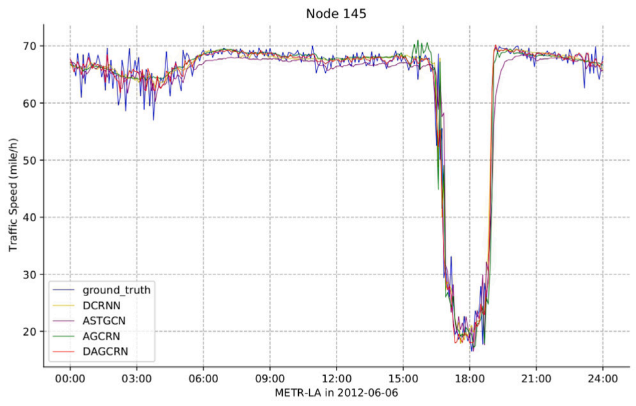
To further analyze how DAGCRN captures the spatial adjacency relations dynamically, we plot the dynamic spatial heatmap of Node 115 from 17:00 to 20:00 in Fig. 6, where the x-axis and y-axis represent node index and time, respectively. First, we find that DAGCRN gave less attention to other nodes and focused on the node itself more when the road was unimpeded. Then as time passed and congestion occurred, Node 36, 118, 145, and 184 gained more attention. We also plot the speed records of Node 145 in Fig. 5(b) due to its strong spatial attention. We find that the decline of traffic speed of Node 145 tended to precede that of Node 115 by about 45 min. When checking the geographical locations of the two nodes, we find that they are located on the same arterial road, and the distance between the two nodes is around 0.97 miles. Additionally, Node 145 is downstream of Node 115, which implies that congestion propagated from Node 145 to Node 115 with the shift of time. This result shows that DAGCRN could learn dynamically changing spatial dependencies effectively. Overall, the interpretable and superior experimental results of DAGCRN are due to the fact that DAGCRN uses SREM and AMUM to model the evolution of dynamic spatial relations and leverages DGCRM to capture the complicated spatial–temporal dependencies of traffic data.

5.7. Parameter study

In this subsection, we conduct a parameter study on core hyper-parameters of DAGCRN on the two datasets to analyze their impacts. The chosen hyper-parameters are listed as follows: (1) dimension of hidden node state d_{model}, ranging from 32 to 96, (2) number of heads h in spatial attention, ranging from 1 to 8, and (3) the largest propagation depth K in graph convolution, ranging from 0 to 3. We record the average of MAE on the validation set. In each experiment, we only change one hyper-parameter while fixing the other parameters. Figs. 7–9 show the experimental results of parameter study. As shown in Fig. 7, the representation ability of DAGCRN improves dramatically with the increase of the dimension of hidden node state. It is worthwhile noting that if we set the dimension of hidden node state too large, it will



(a) Prediction visualization of Node 115



(b) Prediction visualization of Node 145

Fig. 5. Time series of ground truth and prediction results on METR-LA dataset.

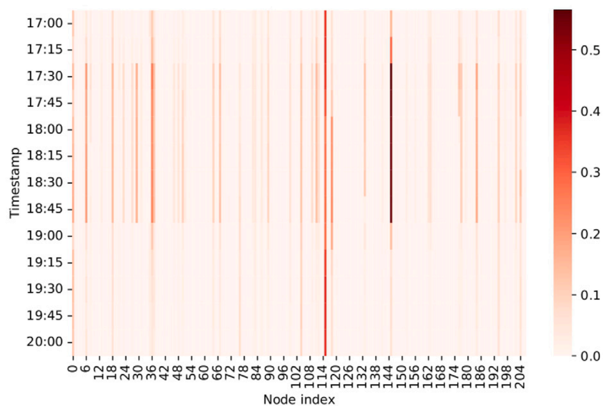


Fig. 6. Dynamic spatial heatmap of Node 115 in different time steps.

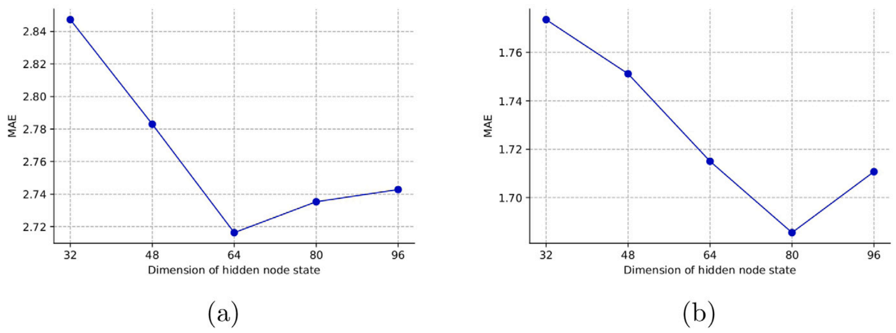


Fig. 7. Effects of dimension of hidden node state. (a) and (b) are respectively the results on METR-LA and PEMS-BAY.

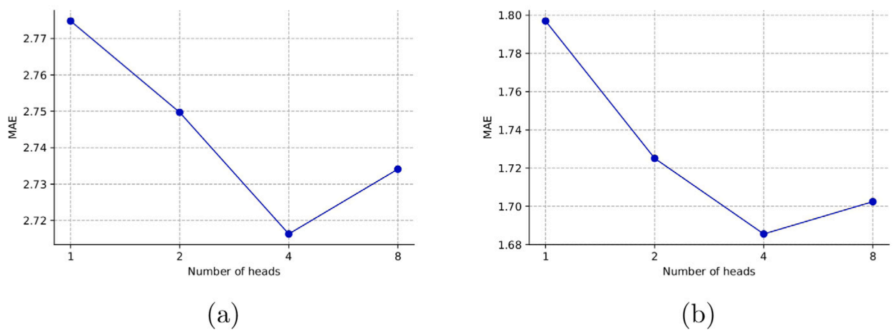


Fig. 8. Effects of number of heads. (a) and (b) are respectively the results on METR-LA and PEMS-BAY.

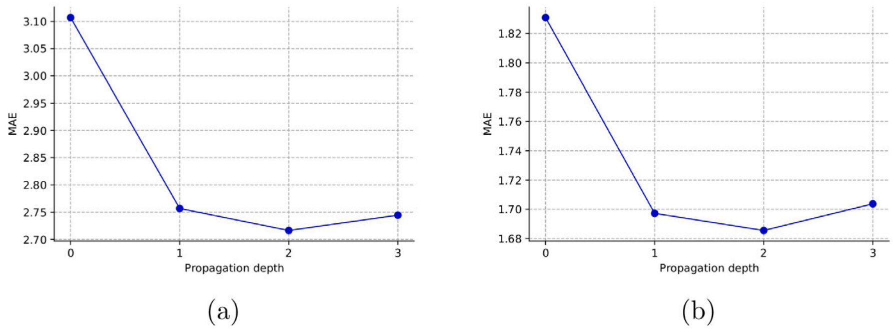


Fig. 9. Effects of propagation depth. (a) and (b) are respectively the results on METR-LA and PEMS-BAY.

result in over-fitting, deteriorating the performance of the model. There is an optimal value for the dimension of hidden node state at 64 for METR-LA and 80 for PEMS-BAY. We hold the opinion that there are more nodes and more samples in PEMS-BAY, so the model needs richer hidden representations to distinguish different nodes and store the more complicated temporal patterns. Fig. 8 shows that increasing the number of heads will improve the forecasting performance. This can be explained that the model can capture more diverse perspectives of spatial dependency with a larger number of heads. Likewise, if the number of heads is set too large, a single head will contain less information, and the spatial dependency will not be fully explored. There is an optimal value for the number of heads at 4 for both datasets. Fig. 9 shows the effects of propagation depth. The larger propagation depth indicates that the model can attend to more neighbors. We find that aggregating neighbors' information can remarkably enhance the model's performance compared with $depth = 0$. However, there exists an increase of MAE when the propagation depth changes from 2 to 3. This is because the larger propagation depth not only leads to more computation costs but also causes the over-smoothing problem. There is an optimal value for the propagation depth at 2 for both datasets.

To sum up, all the hyper-parameters have non-negligible impacts on the ultimate performance of DAGCRN and should be verified based on the specific dataset. We can find that the results are similar across the two datasets. This is because the two datasets have relatively similar traffic patterns, and also indicates that the model is with good generalization ability. We select the optimal values of all the hyper-parameters with reference to the model's performance on the validation set.

5.8. Complexity study

To evaluate the complexity of our model, we compare the computation time and the number of parameters of DAGCRN with several graph-based models, including DCRNN, ASTGCN, STSGCN, and AGCRN. All the experiments are conducted on the RTX 2080Ti GPU with 11 GB memory and the size of a mini-batch is uniformly set to 64. Under this experiment condition, we count the time cost of five models on the test set (see Table 4). We naturally get the following conclusions: (1) Compared with CNN-based temporal modeling methods (*e.g.* ASTGCN and STSGCN), RNN-based methods (*e.g.* DCRNN, AGCRN,

Table 4
Complexity study on the METR-LA dataset.

Methods	Computation time		# Parameters
	Training(s/epoch)	Inference(s)	
DCRNN	446.85	30.69	372353
ASTGCN	127.74	11.68	237731
STSGCN	140.29	12.56	1921886
AGCRN	216.12	23.61	751650
DAGCRN(ours)	187.06	15.85	236230

and DAGCRN) require more computation cost due to their intrinsic recurrent structures. However, the proposed DAGCRN still requires relatively less computation time owing to its efficient tensor operations. (2) Because STSGCN combines three graphs at adjacent time steps into a holistic graph as the adjacency matrix, it requires a large number of parameters to model the complex spatial–temporal dependencies. Since key components of DAGCRN are all shared across different time steps, it uses fewer parameters than other baselines.

Overall, our proposed DAGCRN uses reasonable computation costs and fewer parameters while obtains the best forecasting performance. The complexity study indicates the feasibility and practical application prospects of our model.

6. Conclusion

In this paper, we have presented a novel deep learning model DAGCRN for traffic forecasting, which successfully models the dynamics of spatial relations and handles the complicated spatial–temporal dependencies of traffic data. Experimental results on two real-world datasets for multi-step traffic speed forecasting prove the effectiveness and robustness of DAGCRN in both short-term and long-term forecasting tasks. What is more, the experimental results of three aspects, including ablation study, case study, and complexity study, also demonstrate various advantages of DAGCRN.

In the future study, more external factors will be considered and utilized to enhance the performance of the traffic forecasting model.

CRediT authorship contribution statement

Zheng Shi: Methodology, Writing and editing. **Yingjun Zhang:** Conceptualization, Methodology, Writing, Reviewing, Supervision. **Jingping Wang:** Methodology, Validation. **Jiahu Qin:** Methodology, Supervision. **Xiaoqian Liu:** Methodology, Validation. **Hui Yin:** Methodology, Validation. **Hua Huang:** Visualization, Investigation.

Declaration of competing interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service, and/or company that could be constructed as influencing the position presented in, or the review of the manuscript entitled.

Data availability

Data will be made available on request.

Acknowledgements

The authors are grateful to the editors and anonymous referees for their valuable comments and efforts. This work was supported in part by the National Key Research and Development Program of China (2022YFB2603302), the National Nature Science Foundation of China (51827813) and the R&D Program of Beijing Municipal Education Commission (KJZD20191000402).

References

Atwood, J., & Towsley, D. (2016). Diffusion-convolutional neural networks. In *Conference on neural information processing systems* (pp. 1993–2001).
Bai, L., Yao, L., Li, C., Wang, X., & Wang, C. (2020). Adaptive graph convolutional recurrent network for traffic forecasting. In *Conference on neural information processing systems* (pp. 17804–17815).
Bai, L., Yao, L., Wang, X., & Sheng, Q. (2019). STG2Seq: Spatial–temporal graph to sequence model for multi-step passenger demand forecasting. In *Proceedings of the international joint conference on artificial intelligence* (pp. 1981–1987).
Bengio, S., Vinyals, O., Jaitly, N., & Shazeer, N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. In *Conference on neural information processing systems* (pp. 1171–1179).
Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2014). Spectral networks and locally connected networks on graphs. In *International conference on learning representations* (pp. 1–14).
Cai, L., Janowicz, K., Mai, G., Yan, B., & Zhu, R. (2020). Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. *Transactions in GIS*, *24*, 736–755. http://dx.doi.org/10.1111/tgis.12644.
Cho, K., Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on empirical methods in natural language processing* (pp. 1724–1734).
Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Conference on neural information processing systems* (pp. 3842–3852).
Diao, Z., Wang, X., Zhang, D., Liu, Y., Xie, K., & He, S. (2019). Dynamic spatial–temporal graph convolutional neural networks for traffic forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 890–897).
Elsheikh, A. H. (2022). Bistable morphing composites for energy-harvesting applications. *Polymers*, *14*(9), 1893. http://dx.doi.org/10.3390/polym14091893.
Elsheikh, A. H. (2023). Applications of machine learning in friction stir welding: prediction of joint properties, real-time control and tool failure diagnosis. *Engineering Applications of Artificial Intelligence*, *121*, Article 105961. http://dx.doi.org/10.1016/j.engappai.2023.105961.
Elsheikh, A. H., El-Said, E. M., Abd Elaziz, M., Fujii, M., & El-Tahan, H. R. (2023). Water distillation tower: Experimental investigation, economic assessment, and performance prediction using optimized machine-learning model. *Journal of Cleaner Production*, *388*, Article 135896. http://dx.doi.org/10.1016/j.jclepro.2023.135896.
Elsheikh, A. H., Muthuramalingam, T., Shanmugan, S., Ibrahim, A. M. M., Ramesh, B., Khoshaim, A. B., Moustafa, E. B., Bedairi, B., Panchal, H., & Sathyamurthy, R. (2021). Fine-tuned artificial intelligence model using pigeon optimizer for prediction of residual stresses during turning of inconel 718. *Journal of Materials Research and Technology*, *15*, 3622–3634. http://dx.doi.org/10.1016/j.jmrt.2021.09.119.
Elsheikh, A. H., Shehabeldeen, T. A., Zhou, J., Showaib, E., & Abd Elaziz, M. (2021). Prediction of laser cutting parameters for polymethylmethacrylate sheets using random vector functional link network integrated with equilibrium optimizer. *Journal of Intelligent Manufacturing*, *32*, 1377–1388. http://dx.doi.org/10.1007/s10845-020-01617-7.
Fang, S., Zhang, Q., Meng, G., Xiang, S., & Pan, C. (2019). GSTNet: Global spatial–temporal network for traffic flow prediction. In *Proceedings of the international joint conference on artificial intelligence* (pp. 2286–2293).
Guo, K., Hu, Y., Qian, Z., Sun, Y., Gao, J., & Yin, B. (2020). Dynamic graph convolution network for traffic forecasting based on latent network of laplace matrix estimation. *IEEE Transactions on Intelligent Transportation Systems*, *23*(2), 1009–1018. http://dx.doi.org/10.1109/TITS.2020.3019497.
Guo, K., Hu, Y., Sun, Y., Qian, S., Gao, J., & Yin, B. (2021). Hierarchical graph convolution network for traffic forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 151–159).
Guo, S., Lin, Y., Feng, N., Song, C., & Wan, H. (2019). Attention based spatial–temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 922–929).
Guo, G., & Yuan, W. (2020). Short-term traffic speed forecasting based on graph attention temporal convolutional networks. *Neurocomputing*, *410*, 387–393. http://dx.doi.org/10.1016/j.neucom.2020.06.001.
Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive representation learning on large graphs. In *Conference on neural information processing systems* (pp. 1024–1034).
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*, 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.
Jiang, W., & Luo, J. (2022). Graph neural network for traffic forecasting: A survey. *Expert Systems with Applications*, Article 117921. http://dx.doi.org/10.1016/j.eswa.2022.117921.
Jiang, R., Yin, D., Wang, Z., Wang, Y., Deng, J., Liu, H., Cai, Z., Deng, J., Song, X., & Shibasaki, R. (2021). Dl-traff: Survey and benchmark of deep learning models for urban traffic prediction. In *Proceedings of the ACM international conference on information and knowledge management* (pp. 4515–4525).
Khoshaim, A. B., Elsheikh, A. H., Moustafa, E. B., Basha, M., & Mosleh, A. O. (2021). Prediction of residual stresses in turning of pure iron using artificial intelligence-based methods. *Journal of Materials Research and Technology*, *11*, 2181–2194. http://dx.doi.org/10.1016/j.jmrt.2021.02.042.

Khoshaim, A. B., Moustafa, E. B., Bafakeeh, O. T., & Elsheikh, A. H. (2022). An optimized multilayer perceptrons model using grey wolf optimizer to predict mechanical and microstructural properties of friction stir processed aluminum alloy reinforced by nanoparticles. *Coatings*, *11*(12), 1476. http://dx.doi.org/10.3390/coatings11121476.

Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International conference on learning representations* (pp. 1–14).

Kumar, S. V., & Vanajakshi, L. (2015). Short-term traffic flow prediction using seasonal arima model with limited input data. *European Transport Research Review*, *7*(3), 1–9. http://dx.doi.org/10.1007/s12544-015-0170-8.

Li, F., Feng, J., Yan, H., Jin, G., Jin, D., & Li, Y. (2021). Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution. *ACM Transactions on Knowledge Discovery from Data*, *17*(1), 1–21. http://dx.doi.org/10.1145/3532611.

Li, W., Wang, X., Zhang, Y., & Wu, Q. (2021). Traffic flow prediction over muti-sensor data correlation with graph convolution network. *Neurocomputing*, *427*, 50–63. http://dx.doi.org/10.1016/j.neucom.2020.11.032.

Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International conference on learning representations* (pp. 1–16).

Li, R., Zhang, F., Li, T., Zhang, N., & Zhang, T. (2022). DMGAN: Dynamic multi-hop graph attention network for traffic forecasting. *IEEE Transactions on Knowledge and Data Engineering*, 1–14. http://dx.doi.org/10.1109/TKDE.2022.3221316.

Liu, L., Zhen, J., Li, G., Zhan, G., He, Z., Du, B., & Lin, L. (2020). Dynamic spatial–temporal representation learning for traffic flow prediction. *IEEE Transactions on Intelligent Transportation Systems*, *22*, 7169–7183. http://dx.doi.org/10.1109/TITS.2020.3002718.

Lu, Z., Lv, W., Cao, Y., Xie, Z., Hao, P., & Du, B. (2020). Lstm variants meet graph neural networks for road speed prediction. *Neurocomputing*, *400*, 34–45. http://dx.doi.org/10.1016/j.neucom.2020.03.031.

Lu, Z., Zhou, C., Wu, J., Jiang, H., & Cui, S. (2016). Integrating granger causality and vector auto-regression for traffic prediction of large-scale wlans. *KSII Transactions on Internet and Information Systems*, *10*, 136–151.

Moustafa, E. B., & Elsheikh, A. H. (2023). Predicting characteristics of dissimilar laser welded polymeric joints using a multi-layer perceptrons model coupled with archimedes optimizer. *Polymers*, *15*(1), 233. http://dx.doi.org/10.3390/polym15010233.

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 ACM international conference on information and knowledge management* (pp. 1215–1224).

Reza, S., Ferreira, M., Machado, J., & Tavares, R. (2022). A multi-head attention-based transformer model for traffic flow forecasting with a comparative analysis to recurrent neural networks. *Expert Systems with Applications*, *202*, 1–12. http://dx.doi.org/10.1016/j.eswa.2022.117275.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, *20*, 61–80. http://dx.doi.org/10.1109/TNN.2008.2005605.

Shao, W., Jin, Z., Wang, S., Kang, Y., Xiao, X., Menouar, H., Zhang, Z., Zhang, J., & Salim, F. (2022). Long-term spatio-temporal forecasting via dynamic multiple-graph attention. In *Proceedings of the international joint conference on artificial intelligence* (pp. 2225–2232).

Shi, X., Qi, H., Shen, Y., Wu, G., & Yin, B. (2021). A spatial–temporal attention approach for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, *22*, 4909–4918. http://dx.doi.org/10.1109/TITS.2020.2983651.

Song, C., Lin, Y., Guo, S., & Wan, H. (2020). Spatial–temporal synchronous graph convolutional networks: A new framework for spatial–temporal network data forecasting. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 914–921).

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Conference on neural information processing systems* (pp. 3104–3112).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In *Conference on neural information processing systems* (pp. 5998–6008).

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In *International conference on learning representations*.

Wang, Y., Fang, S., Zhang, C., Xiang, S., & Pan, C. (2022). TVGCN: Time-variant graph convolutional network for traffic forecasting. *Neurocomputing*, *471*, 118–129. http://dx.doi.org/10.1016/j.neucom.2021)11.006.

Wang, X., Ma, Y., Wang, Y., Jin, W., Wang, X., Tang, J., Jia, C., & Yu, J. (2020). Traffic flow prediction via spatial temporal graph neural network. In *Proceedings of the web conference* (pp. 1082–1092).

Wang, Y., Ren, Q., & Li, J. (2023). Spatial–temporal multi-feature fusion network for long short-term traffic prediction. *Expert Systems with Applications*, Article 119959. http://dx.doi.org/10.1016/j.eswa.2023.119959.

Wu, C., Ho, J., & Lee, D. (2004). Travel time prediction with support vector regression. *IEEE Transactions on Intelligent Transportation Systems*, *vol. 5*, 276–281. http://dx.doi.org/10.1109/TITS.2004.837813.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X., & Zhang, C. (2020). Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 753–763).

Wu, Z., Pan, S., Long, G., Jiang, J., & Zhang, C. (2019). Graph wavenet for deep spatial–temporal graph modeling. In *Proceedings of the international joint conference on artificial intelligence* (pp. 1907–1913).

Yao, H., Tang, X., Wei, H., Zheng, G., & Li, Z. (2019). Revisiting spatial–temporal similarity: A deep learning framework for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 5668–5675).

Yin, X., Wu, G., Wei, J., Shen, Y., Qi, H., & Yin, B. (2021a). Deep learning on traffic prediction: Methods, analysis and future directions. *IEEE Transactions on Intelligent Transportation Systems*, *vol. 23*(6), 4927–4943. http://dx.doi.org/10.1109/TITS.2021.3054840.

Yin, X., Wu, G., Wei, J., Shen, Y., Qi, H., & Yin, B. (2021b). Multi-stage attention spatial–temporal graph networks for traffic prediction. *Neurocomputing*, *vol. 428*, 42–53. http://dx.doi.org/10.1016/j.neucom.2020.11.038.

Yu, B., Yin, H., & Zhu, Z. (2018). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the international joint conference on artificial intelligence* (pp. 3634–3640).

Zhang, M., & Chen, Y. (2018). Link prediction based on graph neural networks. In *Conference on neural information processing systems* (pp. 5171–5181).

Zhang, Z., Li, Y., Song, H., & Dong, H. (2021). Multiple dynamic graph based traffic speed prediction method. *Neurocomputing*, *vol. 461*, 109–117. http://dx.doi.org/10.1016/j.neucom.2021.07.052.

Zhang, J., Shi, X., Xie, J., Ma, H., King, I., & Yeung, D. (2018). GaAN: Gated attention networks for learning on large and spatiotemporal graphs. In *Proceedings of the conference on uncertainty in artificial intelligence* (pp. 339–349).

Zhang, J., Zheng, Y., & Qi, D. (2017). Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1655–1661).

Zhao, J., Liu, Z., Sun, Q., Li, Q., Jia, X., & Zhang, R. (2022). Attention-based dynamic spatial–temporal graph convolutional networks for traffic speed forecasting. *Expert Systems with Applications*, *vol. 204*, Article 117511. http://dx.doi.org/10.1016/j.eswa.2022.117511.

Zheng, C., Fan, X., Wang, C., & Qi, J. (2020). GMAN: A graph multi-attention network for traffic prediction. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1234–1241).

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, *1*, 57–81. http://dx.doi.org/10.1016/j.aiopen.2021.01.001.