Multi-step Coupled Graph Convolution with Temporal-Attention for Traffic Flow Prediction

XIAOHUI HUANG1, YUMING YE1, XIAOFEI YANG2, AND LIYAN XIONG1.

1School of Information Engineering Department, East China Jiaotong University, Nanchang 330013, China
2Faculty of Science and Technology, Faculty of Science and Technology University of Macau, E11, Macau, 999078, China
Corresponding author: Yuming Ye (e-mail: yming.yes@foxmail.com)
This work was supported in part by the National Natural Science Foundation of China under Grant 62062033 and Grant 62067002, and the Natural Science Foundation of Jiangxi Province under Grant No.20212BAB202008.

ABSTRACT
Forecasting traffic flow is significant for intelligent transportation systems (ITS), such as urban road planning, traffic control, traffic planning, and many more. A flow prediction model aims at forecasting the traffic flow of future time slices at certain regions by learning the historical traffic flow data and environmental information. However, due to the complicated traffic network topology and the dynamicity of traffic patterns in the real world, it is difficult to capture the multi-level spatial dependencies (e.g. global and local impacts to the traffic) and temporal dependencies (e.g. long-term and short-term impacts to the traffic). In this paper, we propose a Multi-step Coupled Graph Convolution Neural network (MCGCN) with temporal attention to capture the spatial and temporal dependencies of different levels in a traffic network, simultaneously, to predict traffic flow. First, a Multi-step Coupled Graph Convolution module (MCGC) is designed to learn the representation of a traffic network by coupling learning the relationship matrices, to capture the different levels’ information of a traffic network. Then, the traffic network information extracted by MCGC is fed into a Multi-step Coupled Graph Gated Recurrent Unit (MCGRU) module to realize the fusion of traffic network information and temporal features. Finally, a Multi-step Coupled Graph Attention mechanism (MCGCAtt) is used to extract the temporal information of historical time steps to predict the future traffic flow. The experiments are conducted on the NYCTaxi and NYCBike datasets, and the evaluation results demonstrate that our proposed model performs better than the eight compared methods.

INDEX TERMS Graph convolutional network; Multi-step attention; Traffic flow prediction

I. INTRODUCTION
As a vital part of intelligent transportation systems, traffic flow forecasting plays an important role in road planning, flow control, vehicle scheduling, etc. It provides a scientific decision-making basis for making reasonable dredging measures by perceiving road traffic congestion in advance, so as to offer better travel service for passengers. However, accurately predicting traffic flow is a tough work since it is often affected by many factors, such as commute structure between regions, weather, traffic accident, and other external factors. The patterns of traffic flow are usually complex and dynamic, which makes the prediction become a challenging problem. The main challenges are as follows:

- Spatial dependence: The traffic flow of a location is often affected by the topology of the traffic road network, the locations of intersections, the front and back traffic condition of the location.
- Dynamic temporal dependence: For most regions in a road network, the traffic flow at a certain moment usually changes dynamically over time, but often shows a certain periodicity and trend.
- External environmental factors: In a transportation system, different environmental factors have different degrees of influence on the traffic flow. For example, heavy rain may result in a traffic jam in the whole city, but blockage of a certain road will often just influence the local traffic flow near the road.

Currently, the mainstream approaches [1]–[3] for solving the problem of traffic flow prediction are deep learning-based methods in which CNN and RNN modules are utilized to
capture the spatial and temporal dependencies, respectively. For example, Zhang et al. [4] divided a city into grids according to longitude and latitude, and extracted the spatial dependence of traffic patterns with CNN. Zhao et al. [5] regarded historical traffic flow data as a priori knowledge and utilized LSTM to extract the temporal dependences of traffic flow data. However, since the complex topological structure of the traffic network and dynamic traffic patterns with the changing of the external environment, it is difficult to extract the multi-level spatial dependencies (e.g., global and local impacts to the traffic) and temporal dependencies (e.g., long-term and short-term impacts to the traffic) with traditional methods. Owing to the graph neural network can better tackle non-Euclidean distance data such as traffic flow data, more and more researchers apply graph convolution to traffic flow prediction. Li et al. [6] modeled the traffic flow as a diffusion procedure on a directed graph and introduced Diffusion Convolutional Recurrent Neural Network (DCRNN). Yu et al. [7] proposed a new deep learning model, Spatio-Temporal Graph Convolutional Networks (STGCN), to solve the time-series data prediction issue in the traffic field. The traffic flow of the traffic network at a certain time slice may be affected by many different factors, for example, the congestion at the intersection may affect the flow of nearby sections, while the rainstorm weather may affect the traffic flow of the whole city. For most popular GCNs-based methods, since the adjacency matrix of graph convolution is fixed, it is tough to capture complex dynamic spatial dependencies. Moreover, the traffic flow in different historical time steps has different effects on the flow in the future (for example, the traffic in the afternoon peak may be related to traffic in the morning peak at 8 or 9 a.m.), and most existing methods do not make full use of this information in the multi-step prediction.

To resolve the problems mentioned above, we introduce a multi-step coupled graph convolution neural network with temporal attention mechanism. First, we designed a multi-step coupled mapping mechanism to dynamically learn global and local traffic information by multiple graph convolutional layers to capture the different degrees of influences caused by various factors. Then, the graph representation produced by the multi-step coupled graph convolution is fed into a recurrent neural network to realize the fusion of traffic network information and temporal features. Finally, we combine the multi-step attention mechanism with the encoder-decoder structure to dynamically forecast the future traffic flow. In general, the contributions of our work are three folds:

- We devise a novel multi-step coupled graph convolutional module to learn global and local traffic features by multiple graph convolutional layers.
- We use multi-step coupled graph convolution instead of linear operation in GRU to redesign a new MCGRU module to realize the fusion of spatial characteristics and time dependence of traffic prediction.
- We propose a multi-step temporal attention mechanism that can extract useful information from historical traffic flow, to more accurately predict traffic flow in the future.

The outline of the remaining sections is as follows: Section II summarizes the different types of methods of traffic flow prediction. The definitions of traffic flow and graph convolutional neural network are given in Section III. Section IV elaborates the framework structure and detailed workflow of the MCGCN model. The experimental results to verify the effectiveness of MCGCN are placed in Section V. Finally, we summarize this work in Section VI and prospects for future works.

II. RELATED WORK

In this section, we briefly review the previous methods above traffic flow prediction from the following three aspects: statistical analysis-based methods, recurrent neural network-based methods and graph neural network-based methods.

A. STATISTICAL ANALYSIS-BASED PREDICTION METHODS

The prediction models based on statistical analysis usually use prior knowledge and hypothetical reasoning to model the correlations between traffic flow and environmental factors and solve the traffic prediction problems by establishing a set of mathematical mapping functions. Alghamdi et al. [8] presented a short-term prediction model (ARIMA) for Non-Euclidean distance data. Comparing with other parametric models such as ARCH [9], the ARIMA model can achieve lower error. However, this method ignores the spatial dependence in the process of flow prediction. Four kinds of ARIMA-GARCH [10] methods are built by integrating ARIMA [8] and GARCH [11] algorithms to forecast the short-term passenger flow. Comparing with ARIMA model, ARIMA-GARCH is able to obtain better prediction results. Lin et al. [12] proposed another ARIMA-GARCH-based model to obtain the fluctuation features of the traffic flow to get better prediction results. The experimental results demonstrate that the NRMSE and MAPE of ARIMA-GARCH-based model are only 3.13% and 8.76%, respectively, which is better than the two compared methods. However, ARIMA-GARCH-based model is only suitable for short-term traffic flow forecasting, and it is still not competent for capturing long-term traffic flow characteristics. Xiao et al. [13] put forward the dynamics grey model of traffic flow mechanics (TFDGN), and predicted traffic flow by establishing traffic flow differential equation and parameter analysis. The experiment is carried out in the traffic data of two cities. Compared with other 6 grey prediction models such as Verhulst and GM, the error of this method is lower. However, the matrix multiplication in this model brings the problem of large amount of computational cost.

Bai et al. [14] proposed the location congestion tensor predictor model (PrePCT) for traffic congestion location prediction. Firstly, the congestion matrix is constructed by using the relative position information of road nodes, and then the long-short term memory network is used to predict the loca-
tion congestion in near time steps. Yang [15] et al. proposed a new Adaptive Graph Convolution Network (ASTGCN) for urban crowd flow prediction, and extracted the correlation of stacking traffic flow by designing different spatio-temporal convolution components. The complexity and dynamics of traffic system make the prediction of traffic speed difficult. Zhang [16] et al. proposed the Evolutionary Time Graph Convolution Network (ETGCN). ETGCN uses the attention mechanism with similarity to fuse multiple graph adjacency matrices, and combines the gating recurrent unit with GCN to capture the temporal and spatial correlation of traffic speed. Ye et al. [17] introduced a short-term traffic flow prediction model based on XGBoost method to forecast the traffic speed. Wu et al. [18] proposed a gradient boosting decision tree (SSGBDT) to forecast bus passenger flow. Experiments on two real public transport data sets in Guangzhou show that SSGBDT can achieve high and stable prediction accuracy, and can better deal with the multi-collinearity problem of multi-source data.

However, the traditional statistical analysis method often needs a complex modeling process. For example, some classical statistical methods such as VAR [19] and Kalman filtering [20] have a broad and solid mathematical foundation. However, the stability hypothesis of these models are destroyed by the high nonlinearity and dynamics of traffic data, causing these methods to perform poorly in practical applications [21]. Moreover, due to the limitation of model capacity, it is difficult to fuse the diverse modes and capture complex nonlinear patterns hidden in data for traditional linear models.

B. RNN-BASED PREDICTION METHODS

Recently, recurrent neural network-based methods have been widely used by the academic and industrial worlds for handling various time series data. According to a learned nonlinear function, recurrent neural network can better extract the complicated state of a traffic system and achieve the purpose of traffic flow prediction. Law et al. [22] constructed a deep network architecture to identify highly relevant features to predict monthly passenger flow in Macau. However, the complex spatial dependencies of road connections and the dynamic temporal modes of traffic states make it challenging. To solve these problems, Ma et al. [23] devise a novel capsule network (CapsNet) to extract the spatial features of traffic networks and design a nested LSTM (NLSTM) module to extract the temporal dependencies in traffic sequence data. Dynamic Transition Convolutional Neural Network (DTCNN) is proposed [24] for the purpose of precise traffic demand prediction. Experiments conducted on NYC taxi and bike-sharing data validate the effectiveness of the proposed method. A successful predicting model should consider the impact of the far and near segments on the traffic flow at the same time. Based on this theory, Lee et al. [25] extends the traffic road network structure to the deep neural network to accommodate citywide spatial-temporal dependencies. Ma et al. [26] used the real-time parking data provided by SFPark.org and the vehicle speed data in downtown San Francisco to study the relationship between vehicle speed and parking space occupancy ratio. Cui et al. [27] investigates the performance of deep Convolutional Neural Network (CNN) for recognizing highway traffic congestion state in surveillance camera images. The experimental results show that the robustness and accuracy of the proposed method can achieve satisfactory performance. Due to the complex spatio-temporal correlation of actual roads and the limitations of intersection detection equipment, there are still many challenges in spatio-temporal traffic flow prediction.

In order to predict the vessel trajectory with high accuracy, Liu et al. [30] proposed an AIS data-driven trajectory prediction framework based on long-term and short-term memory network (LSTM). The vessel traffic conflict state modeling generated by dynamic AIS data and social force concept is embedded into LSTM. Experiments conducted on a large number of real vessel trajectory data sets show that the robustness and accuracy of the proposed method can achieve satisfactory performance. Due to the complex spatio-temporal correlation of actual roads and the limitations of intersection detection equipment, there are still many challenges in spatio-temporal traffic flow prediction. In order to capture the temporal and spatial correlation between roads for traffic flow prediction, Zhang et al. [31] proposed an adversarial learning traffic prediction model named TrafficGAN, where the generated network of the GAN is used to predict traffic flow. In order to solve the problem of multi-scaled (coarse-grained and fine-grained) traffic flow prediction, Wang et al. [32] proposed a multi-task spatio-temporal network model MTstnets, which uses cross-scale spatio-temporal feature learning and fusion technique to process fine-grained and coarse-grained traffic data. Considering the multi-channel and irregularity of urban traffic flow, a more efficient deep spatio-temporal learning model is needed. Based on this, Du et al. [33] proposed a deep irregular convolution residual LSTM network to test different types of traffic flow data. The results show that the proposed method is significantly better than other urban traffic flow prediction methods based on deep learning.

Most recurrent neural network-based methods regard a city as regular grids, and then employ convolution to extract the features of the traffic network. These methods cannot extract the information of the real physical structure of a traffic network, nor can they mine complex traffic patterns from non-Euclidean distance data.

C. GNN-BASED PREDICTION METHODS

Recently, as an emerging learning technique, Graph Neural Network (GNN) has received extensive attention for dealing with graph data. [21], [34] Compared with traditional neural networks, graph neural networks can process data in non-Euclidean space, such as knowledge graphs [35], community relations [36], etc. Since the traffic network in a city is natural graph structure, graph neural network is logically chosen for traffic flow prediction by some researchers [37], [38]. For examples, Luo et al. [37] proposed a data-driven flow forecasting method to predict the long-term demand of stations,
which uses graph convolution neural network to dynamically capture the correlation between stations, and Guo et al. [38] proposed an optimized graph volume recurrent neural network for traffic prediction. Temporal graph convolutional network and spatiotemporal self-attention network (GACNet) are proposed [39] to capture traffic status features and spatiotemporal feature from input sequence. Experimental results gained on two traffic speed datasets show that GACNet achieves the performance over the compared algorithms.

Chu [40] proposed a deep learning approach to estimate the waiting time at transportation sites. Guo et al. [41] proposed an optimized recursive graph convolution neural network, which represented road spatial information as a graph. Additionally, this method continuously optimized the graph structure in a data-driven manner during the training process, thereby dynamically updating the spatial relationship between road segments. By combining the current CNN-based and GCN-based traffic flow prediction models, Qiu et al. [42] employed a Topological Graph Convolutional Network (ToGCN) combined with a encoder-decoder module to predict future traffic flow. The experiment is conducted on two real taxi data sets, which prove the validity of the method. Wang et al. [43] proposed a hierarchical traffic flow prediction model based on spatiotemporal graph convolution network (ST-GCN). By combining the Adjacent-Similarity graph, the proposed model can effectively predict the intersections traffic flow. The experiment based on the actual traffic data of Qingdao shows that the ST-GCN-based model is better than the advanced baseline model. Peng et al. [44] proposed a long-term traffic flow prediction method based on dynamic graph, which uses dynamic traffic flow probability graph to model the traffic network. However, the existing graph neural network methods usually learn the graph representation of a static traffic network structure, which cannot dynamically capture the dependencies of different levels, such as DCRNN [6], STGCN [45] in which a static adjacency matrix is used in the whole learning process. And some recent graph convolution methods extract spatial dependence and temporal dependence separately, so that they could not achieve better prediction results. For example, Wu et al. [46] first captured the time dependence through two parallel temporal convolutional networks layers, and then used a GCN model to extract the spatial dependence. Similarly, Yu et al. [45] successively used Temporal Gate-Convolution and Spatial Gate-Convolution structure to extract temporal and spatial features, respectively. These methods simply stack time and spatial dependences, rather than combine to learn the features, so it is difficult to obtain the fusing features of two dependencies.

III. PRELIMINARIES

In this part, we will give some symbols and definitions of traffic flow forecast.

A. REPRESENTATION OF TRAFFIC NETWORK

1) Traffic network graph

Given the transportation network of a city as shown in Fig. 1, we can convert it into a graph as shown in Fig. 2. First, we take the stations (such as No. 1-6 with green circle) in the traffic network as the nodes in the graph, and the flow produced by each station over time as a feature vector of the node. Then we calculate the similarity between nodes, and use a threshold to filter some edges of the low similarity. For example, station 5 is connected to station 2 by road in Fig. 1, but if the similarity of traffic between the two stations is less than a certain threshold, it will be considered that there is no edge connection between the two nodes in Fig. 2. Thus, we can transform a traffic network (e.g Fig. 1) into a graph (e.g Fig. 2). By learning the similarity among stations, we can encode dependencies among the stations into a relationship matrix, and then we fuse the information of the similarity as the spatial dependence in the process of flow prediction to improve the accuracy of flow prediction.

Different from the traditional methods that a station or area is regarded as a grip on a map, our proposed method treats a station or area as a node of the graph. By learning the graph, our proposed method can obtain an effective representation of the stations.

2) Relationship matrix construction among stations

Given a graph structure $G = (V, E)$, where $V$ is a set of the nodes and $E$ is a set of the edges. In the transportation system, the traffic stations can be regarded as the nodes $V$ of the graph $G$, and the similarity between the stations can be
regarded as the edges of the graph. At time step $t$, the feature matrix of the nodes in a graph $G$ is $X_t \in R^{N \times d}$ where $d$ is the dimension of the input features (in our experiment, it includes two features: outflow and inflow, i.e. $d = 2$) and $N$ is the number of nodes in the graph. Given graph signal $X$ of $\tau$ time steps, our purpose is to use the data-driven method to obtain a relationship matrix $A^0 \in R^{N \times N}$ to represent the similarities of traffic flow among two stations. The function $F_1$ can be expressed as:

$$[X_{t-\tau+1:t}] \xrightarrow{F_1} A^0,$$

where $t-\tau+1$ represents the first step time in the generation of the relationship matrix and $\tau$ is the number of time steps which is used to learn the relationship matrix.

3) Station traffic flow prediction

At time step $t$, given the graph structure $G$ and the graph signal $X$ of historical $P$ time steps, our goal is to learn an implicit function $F_2$ to predict the flow of the future $Q$ time steps. This process can be defined as:

$$[X_{t-P+1:t}, G] \xrightarrow{F_2} X_{t+1:t+Q},$$

where $X_{t+1:t+Q} \in R^{Q \times N \times d}$ and $X_{t-P+1:t} \in R^{P \times N \times d}$.

B. GRAPH CONVOLUTIONAL NETWORK

Given a graph structure $G = (V, E)$, we define $\hat{A}$ as a normalized relationship matrix:

$$\hat{A} = D^{-1}A,$$

where $A$ is the relationship matrix and $D$ is the diagonal matrix of node degree, $D_{ii} = \sum_j A_{ij}$. In this work, we use diffusion graph convolution [6] on undirected graph to learn the representation of the graph. The graph convolution can be expressed as:

$$X \ast_G g(\theta) = \sum_{i=0}^{k} (\hat{A})^i X \theta^i,$$

where $g(\theta)$ is the parameters $\theta$ of the filter, $\ast_G$ represents the graph convolution operation and $X$ is the input signal.

IV. THE MULTI-STEP COUPLED GRAPH CONVOLUTION WITH TEMPORAL-ATTENTION METHOD

In this section, we will elaborate on our proposed method in detail. As shown in Fig. 3, the Multi-step Coupled Graph Convolution Neural network with temporal-attention model (MCGCN) has three modules: multi-step graph convolution module, graph convolution GRU module, and multi-step temporal attention module. Next, we will introduce each module of the MCGCN model in detail.

A. MULTI-STEP COUPLED GRAPH CONVOLUTION

1) Relationship matrix construction

The relationship matrix is very important in graph convolution since it determines how to aggregate the information of stations. In the following, we will introduce the method of generating relationship matrix by learning the historical flow features of stations. Given a series of graph signals $X_{t-\tau+1:t} \in R^{T \times N \times d}$, we firstly transform the 3-dimensional graph signal tensor into 2-dimensional graph signal tensor $X_{t-\tau+1:t} \in R^{(\tau \times d) \times N}$. In order to capture the similarities among different stations and filtrate out redundant information, we decompose the 2-dimensional matrix $X_{t-\tau+1:t}$ into two matrices:

$$X_{t-\tau+1:t} = X^t \sum (X^s)^T,$$

where $X^t$ and $X^s$ are two low-rank matrices, representing time and station information, respectively. In the experiment, we use the singular value decomposition (SVD) method to decompose matrix $X_{t-\tau+1:t}$. SVD decomposition can obtain two singular matrices, which represent the features of the stations and traffic information in every time step, respectively. We only use the 20 row eigenvectors corresponding the 20 biggest eigenvalues to represent the stations top 20 important features for reducing computational cost. By SVD decomposition of the 2-dimensional graph signal tensor (the shape is $(t \cdot d) \times N$), we can obtain the left singular matrix $X^t$ and the right singular matrix $X^s$, representing time information and stations information, respectively. SVD decomposition can retain information about repeated traffic patterns and filter out some redundant information.

Matrix $X^s \in R^{N \times \xi}$ ($\xi \ll N$) is a high-level compressing representation of each station, where $\xi$ is the dimension of a station feature after SVD decomposition, which represents the most important top $\xi$ features of the station. We use the Gaussian kernel function to calculate the similarity between the $i^{th}$ and $j^{th}$ stations as the weight of their edges:

$$A_{ij}^0 = exp(-\frac{||X^s_i - X^s_j||^2}{\varepsilon^2}),$$

where $\varepsilon$ represents the standard deviation of distances of any pairs of stations $||X^s_i - X^s_j||^2$, which can be defined as:

$$\varepsilon = \sqrt{\sum_{k=1}^{\xi} (||X^s_i - X^s_j||^2 - ||X^s_i - X^s_j||^2_{\xi})^2} / (\xi - 1).$$

2) Multi-step coupled graph convolution

In the traditional methods, a city is usually divided into some grids according to a certain rule, and then CNN is employed to extract the local traffic information. However, it is difficult to capture the associations between two stations with long distances due to the limited receptive field of CNN. To solve this problem, some researchers [37], [47] built a graph structure among the stations and applied graph convolution on the graph instead of traditional convolution on a traffic flow image. Since most of the existing GCN only use the static adjacency matrix in the convolution process to extract the spatial dependences of the traffic flow, it is difficult to mine the hidden information of different environmental factors on the flow (for example, the blockage of an intersection will affect the flow of the nearby road segments, while the
In order to solve this problem, Ye et al. [48] proposed a coupled graph convolutional (CGC) operation on different relationship matrices in different convolution layers. The equation of CGC can be defined as:

$$Z^{m+1} = Z^m \ast_G g^m = \sum_{i=0}^{K} (A^m)^i Z^m \theta^m_i, \quad (8)$$

where $Z^m$ represents the features of stations in $m^{th}$ layer on the graph convolution, $A^m$ is used to model the $m^{th}$ relationship between stations, and $K$ is the number of the layers of diffusion network. And then, a coupling implicit function is used to generate the relationship matrix of the $(m+1)^{th}$ layer as:

$$A^{m+1} = \psi^m(A^{m-l+1}, ..., A^m), \quad (10)$$

where $l$ is the number of layers that $A^{m+1}$ relies on and $\psi^m$ is a coupled mapping function of our proposed method for obtaining the relationship matrix of next layer. In this work, $\phi^m$ is a fully connected neural network that learns with the whole framework. Previous method [46] initialize two adaptive submatrices randomly, and then update the relationship matrix by continuous training. However, such a training process often leads to the problems of difficult convergence and numerical instability [48]. In the following, we will solve the problem by decomposing original relationship matrix $A^0$ into two submatrices as the input of multi-step learning the representation of relationship matrix to ensure convergence and numerical stability.

First, we need to conduct the graph convolution on the relationship matrix to update representation $Z$ of stations.
The first layer of graph convolution can be defined as:

$$Z^1 = \sum_{i=0}^{k} (A^{0})^i Z^0 \theta^0_i,$$

where $Z^0 = X_t$, $A^{0}$ is the relationship matrix obtained by Equation (6) and $\theta^0$ is learnable parameters of graph convolution. The characteristic matrix $X_t \in R^{N \times d}$ is the representation of traffic flow at time step $t$.

However, due to the large number of stations in the traffic network, the computation cost of updating $N \times N$ relationship matrix and coupling function $\varphi$ is high. In order to solve this problem, Singular Value Decomposition is used to decompose $A^{0}$ into two small matrices to reduce the computational cost in advance:

$$A^{0} = E^{0}_1 E^{0}_2^{\top}.$$  \hspace{1cm} (12)

The dimensions of $E^{0}_1$ and $E^{0}_2$ are $N \times L$, $L$ is the number of embedding dimension. In this way, the number of parameters of each coupled mapping function $\varphi^*$ is reduced from $N \times N$ to $2 \times N \times L$. In the experiment, $L \ll N$. Thus, Equation (11) can be rewritten as:

$$Z^1 = \sum_{i=0}^{k} (E^{1}_1 E^{2}_2^{\top})^i Z^0 \theta^0_i.$$  \hspace{1cm} (13)

The submatrices $E^{1}_m$ and $E^{2}_m$ in the rest layers (except $E^{0}_1$ and $E^{0}_2$) are updated by the Equation (14):

$$E^{1}_m = \sum_{i=0}^{m-1} (E^{1}_i)W^{m-1} + b^{m-1},$$
$$E^{2}_m = \sum_{i=0}^{m-1} (E^{2}_i)W^{m-1} + b^{m-1},$$

where $W^{m-1}$ and $b^{m-1}$ are learnable parameters. It is worth noting that $W^{m-1}$ and $b^{m-1}$ are shared in the process of updating $E^{1}_m$ and $E^{2}_m$ since the two matrices affect each other and both of them come from the relationship matrix. The rank of matrices $E^{1}_1$ and $E^{2}_2$ is set to $L$, so the number of parameters in each coupled function is reduced to $L \times (L+1)$. Combining Equation (8) and (10), the updating formula of MCGC can be expressed as:

$$Z^{m+1} = \sum_{i=0}^{k} (A^{m})^i Z^{m} \theta^m_i,$$

$$A^m = \varphi(A^{m-1}, ..., A^{m-1}).$$  \hspace{1cm} (15)

It is worth noting that the $A^{0}$ in the first layer can be updated iteratively with the training process of the whole network. The second layer is updated by the single-step coupled mapping, and the remaining layers are updated by the multi-step coupled mapping function $\varphi$.

3) Multi-level aggregation

In order to comprehensively collect information from multiple graph convolutional layers, we use the attention mechanism to aggregate the features of all layers to select suitable information to achieve prediction tasks. After multi-step coupled graph convolution, the graph signal data can be expressed as $Z = \{Z^1, Z^2, ..., Z^m\}$, $Z \in R^{M \times N \times \beta}$, where $M$ represents the number of steps of multi-coupled convolution, and $\beta$ represents the feature hidden states dimensions of the stations. The attention score is implemented by a linear function, which is defined as follows:

$$\alpha^m = \frac{\exp(Z^mW_a + b_a)}{\sum_{m=1}^{M} \exp(Z^mW_a + b_a)},$$

$$h = \sum_{m=1}^{M} \alpha^m Z^m,$$

where $W_a$ and $b_a$ are the weight and bias of the linear function, $Z^m$ is a vector of the flatten of station representation $Z^m$, $\alpha^m$ is the attention score of each layer of graph convolution, and $h$ is the final representation of the stations aggregated by the multi-step graph convolution.

B. TEMPORAL MODELING WITH CONVOLUTION GATE RECURRENT UNIT

As a variant of RNN, GRU can solve the problem of gradient vanish. In order to capture the temporal and spatial correlations of traffic flow simultaneously, we use the combination of coupled graph convolution and multi-step aggregation to replace the linear transformation in GRU. The multi-step coupled convolution gating recurrent unit (MCGRU) is defined as:

$$r^{(t)} = \sigma(\Theta_r \ast G[h^{(t)}, H^{(t-1)}] + b_r),$$
$$u^{(t)} = \sigma(\Theta_u \ast G[h^{(t)}, H^{(t-1)}] + b_u),$$
$$c^{(t)} = \tanh(\Theta_c \ast G[h^{(t)}, (r^{(t)} \odot H^{(t-1)})] + b_c),$$
$$H^{(t)} = u^{(t)} \odot H^{(t-1)} + (1 - u^{(t)}) \odot c^{(t)},$$

where $h^{(t)}$ and $H^{(t)}$ represent the output of the MCGC module and the GRU module at time step $t$, respectively. $\odot$ represents the Hadamard product, and $\sigma$ is the activation function. The reset gate $r^{(t)}$ helps to forget unnecessary information, and the update gate $u^{(t)}$ can control the output of the convolution GRU at time step $t$. $\Theta_r$, $\Theta_u$, and $\Theta_c$ are the corresponding filter parameters. $b_r$, $b_u$, and $b_c$ are bias.

C. MULTI-STEP TEMPORAL ATTENTION FOR FLOW PREDICTION

In order to select the useful information from the historical time steps to better update the traffic flow in the new future, we combine the multi-step time attention mechanism at the final decoding output. Fig. 4 shows the structure of the multi-step temporal attention module for flow prediction. The goal of the module is to update the decoder by learning the hidden state produced by the encoder to get a set of weight
vectors \( \{ \alpha_i \}_{i=t-P}^{t} \), where \( \alpha_{t-P} \) represents the importance of hidden state \( H_{t-P} \) to the prediction in time step \( t \). The implementation process of multi-step temporal attention is as follows. First, we use the final output of the convolution gate recurrent unit \( \{ H_i \}_{i=t-P}^{t-1} \) in the previous \( P \) time steps and the decoder output \( H_i \) of time step \( t \) as the module’s input of current prediction. And then, the input \( \{ H_i \}_{i=t-P}^{t-1} \) and \( H_t \) are flattened to the vectors \( \{ \tilde{H}_i \}_{i=t-P}^{t-1} \) and \( \tilde{H}_t \), respectively. Next, we conduct matrix product of \( \tilde{H}_i \) and \( \tilde{H}_t \) to get a weight vector \( \alpha \). This process corresponds to the “matmul” module in the Fig. 4. The \( \alpha \) represents the importance of each hidden state in the process of the future traffic flow prediction. The formula of computing the weight vector can be expressed as:

\[
\alpha_k = \tilde{H}_t \times (\tilde{H}_k)^T, \ k \in \{ t - P, t - 1 \}.
\]

And then, we normalize \( \alpha_k \) with softmax normalization by Equation (20):

\[
\bar{\alpha}_k = \frac{\exp(\alpha_k)}{\sum_{k=t-P}^{t-1} \exp(\alpha_k)}, \ k \in \{ t - P, t - 1 \}.
\]

Finally, the attention representation \( R_t \) is obtained by calculating the weight sum of \( \{ \bar{\alpha}_i \}_{i=t-P}^{t} \) and \( \{ H_i \}_{i=t}^{t+Q} \). This process can be expressed as:

\[
R_t = \sum_{k=t-P}^{t+Q} \bar{\alpha}_k H_i, \ i \in \{ t, t+Q \}.
\]

Similarly, we can get the attention representation from time step \( t + 1 \) to time step \( t + Q \): \( \{ R_i \}_{i=t+1}^{t+Q} \). Then, we integrate the attention output \( R_t \) with the original output of the decoder \( H_i \) by fusion mechanism to get the prediction result of each prediction timestep \( \hat{Y}_i \), the fusion mechanism can be expressed as follows:

\[
\hat{Y}_i = W_A R_i + W_H H_i, \ i \in \{ t + 1, t + Q \},
\]

where \( W_A \) and \( W_H \) are learning parameters.

### D. LOSS FUNCTION

The goal of our proposed model is to minimize the root mean square error (RMSE) between the predicted values and the real values of \( Q \) time steps with Equation (23):

\[
RMSE = \sqrt{\frac{1}{Q} \sum_{i} (Y_i - \hat{Y}_i)^2},
\]

where \( Y_i \) and \( \hat{Y}_i \) represent the real traffic flow and forecast traffic flow of the stations, respectively, and \( Q \) represents the number of time steps that we plan to predict.

### V. EXPERIMENTS

In this section, we will elaborate on the details of the experiment from the following aspects: data set, parameter settings, evaluating metric, experimental result analysis, ablation experiments, etc., to verify the effectiveness of our proposed algorithm.

### A. DATA SET

| Dataset | NYC Taxi | NYC Bike |
|---------|----------|----------|
| Start time | 4/1/2016 | 4/1/2016 |
| End time | 6/31/2016 | 6/31/2016 |
| Training set (Days) | 63 | 63 |
| Validating set (Days) | 14 | 14 |
| Testing set (Days) | 14 | 14 |
| Time interval (min) | 30 | 30 |
| Research area (km) | 8.42 × 14.45 | 8.42 × 14.45 |
| Number of stations | 266 | 250 |

Two real data sets on NYC OpenData: NYC Taxi and NYC Bike are used to evaluate our proposed model. The description of the two data sets is as follows:

- **NYC Taxi**: The data set contains approximate 35 million taxi trip records in New York City. It contains the following information: time of getting on and off, latitude and longitude of each station, and stop time.

- **NYC Bike**: The data set contains approximately 35 million bike trip records in New York City. It contains the following information: start time, end time, start and end station IDs, and trip distance.
and longitude of getting on and off, and travel distance. In the experiment, we use the data of 63 days as the training set, and 14 days as the validating set, and 14 days as the testing set.

- **NYCBike**: This data set contains records of shared bicycle orders used by the people in New York City every day. The data set contains the following information: bicycle pick-up point, bicycle drop-off point, bicycle pick-up time, bicycle drop-off time and travel duration. The division of the training set, validating set and testing set is consistent with NYCTaxi.

The details of NYCTaxi and NYCBike data sets are shown in table 1.

### B. DATASET PREPROCESS

In the experiment, we transfer the traffic flow data to the graph with the method of Section IV-A1. For non-station transportation like taxis and sharing bikes, the locations where passengers arrive and leave are random, but they are usually concentrated on some specific areas. For example, there is a lot of demand for taxi orders at the gate of a school, which naturally forms a virtual station [49]. For non-station systems, discovering potential traffic stations can help to capture the characteristics of traffic flow. This paper adopts the method of Density Peak Clustering (DPC) in the literature [50] to explore the potential stations. We finally obtain 266 virtual stations on NYCTaxi dataset and 250 stations on NYCBike dataset. For other clustering methods such K-Means and DBSCAN etc., the distance of samples is recalcualted after each cluster division to find a new cluster center, which will bring a lot of computational cost. DPC can quickly find one or more density peaks, so as to allocate samples efficiently. At the same time, the calculation process of DPC is simple and suitable for cluster analysis of large-scale data such as traffic flow. Therefore, we chose DPC to mine the potential stations of NYCTaxi data. The details of DPC can be referred to the reference [50]. The spatial relationship between stations can be expressed by relationship matrix in experiment. Different from the previous adjacency matrix, which takes the physical distance between stations as the weight of edges, the relationship matrix takes the traffic flow similarities between stations as the weight of edges. The traffic similarities between stations can better express the relationship between stations. For example, there is a large of commute between two distant stations. If the physical distance between stations is used as the weight of edges, the association between the two stations will be weakened. However, the traffic flow similarities between stations can capture the spatial characteristics of those stations with long distance and high correlation. The details of constructing the relationship matrix refer to section III-A2.

The time interval in the experiments is both 0.5 hours on two data sets. We use the Z-score standardization commonly used in previous work [48], [51] to normalize the traffic flow features of all stations. The feature dimensions $d$ of each station is 2, which represents the number of pick-up and drop-off in a corresponding station. Both the historical time steps and the predicted time steps are $12$. $\xi$ is the dimension of station features after decomposition by Equation (5), and its value is 20.

### C. BASELINE

In order to test the performance of our proposed model, we use the following baseline methods for comparison:

- **HA**: Historical Average (HA) predict the traffic flow in and out of the corresponding time slot by calculating the historical average flow value. For example, the flow of 10:00-10:30 am on Wednesday is the average value of the flow during the time period of 10:00-10:30 am on previous Wednesday.
- **XGBoost** [52]: XGBoost is a gradient boosted regression tree technique. By establishing $K$ regression trees, the predicted value of the tree group is as close to the true value as possible.
- **FC-LSTM**: FC-LSTM is a combination of fully connected network and LSTM [53].
- **DCRNN** [6]: DCRNN uses a bidirectional random walk on the graph to capture spatial dependence, and utilizes a seq2seq structure with schedule sampling to extract the temporal dependence of traffic flow.
- **STGCN** [45]: STGCN is a general framework for processing structured time series data. It can not only solve traffic flow prediction problems, but also can be applied to more general spatial-temporal sequence learning tasks.
- **STG2Seq** [54]: STG2Seq uses a GCN-based seq2seq method to model city-wide multi-step ride demand forecasting.
- **GWNNet** [46]: GraphWaveNet is a framework for efficiently obtaining spatial-temporal dependencies at the same time. The core idea of this framework is to fuse the expanded causal convolution with the graph convolution, and then each graph convolution layer can process the spatial dependence of each node information extracted by the expanded causal convolution at different granularities.
- **CCRNN** [48]: CCRNN provides a hierarchical coupling mechanism, which adaptively updates the adjacency matrix of each layer by associating the upper-layer adjacency matrix with the lower-layer adjacency matrix.

### D. EVALUATING METRIC

The evaluation indexes used in the experiment are Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Pearson Correlation Coefficient (PCC), and their calculation formula are as follows:

$$RMSE = \sqrt{\frac{1}{Q} \sum_{i=1}^{Q} (Y_i - \hat{Y}_i)^2}, \quad (24)$$

$$MAE = \frac{1}{Q} \sum_{i=1}^{Q} |Y_i - \hat{Y}_i|, \quad (25)$$

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
\[ p_{Y,Y} = \frac{\text{cov}(\hat{Y}, Y)}{\sigma_Y \sigma_Y} = \frac{E[(\hat{Y} - \mu_Y)(Y - \mu_Y)]}{\sigma_Y \sigma_Y}, \]  

(26)

where \( Y \) and \( \hat{Y} \) represent the real traffic flow and forecast traffic flow of the stations, respectively, \( Q \) represents the number of the time steps which we plan to predict, and \( \mu_Y \) and \( \mu_{\hat{Y}} \) are the means of the real traffic flow and forecast traffic flow, \( \sigma_Y \) and \( \sigma_{\hat{Y}} \) are the standard deviation of the real traffic flow and forecast traffic flow.

### E. EXPERIMENTAL SETTING

From the framework of the MCGCN model in Fig. 3, we can see that the model mainly contains three modules: MCCG, MCGRU, and MCGCAtt. In the experiment, the parameter setting of the model is shown in Table 2.

| Parameter                  | Value       |
|----------------------------|-------------|
| MCCG layers                | 3           |
| Aggregation layer          | 1           |
| Diffusion steps            | 3           |
| Adaptive matrix dimension  | 50          |
| Hidden states dimension    | 25          |
| MCGRU Layer                | 1           |
| Batch Size                 | 32          |
| Epoch                      | 200         |
| Optimizer                  | Adam        |
| Time interval of each time | 30 min      |
| Early stop                 | 20          |
| Learning rate (NYCTaxi)    | 0.0015      |
| Learning rate (NYCBike)    | 0.0005      |

Firstly, in the multi-step coupled graph convolution module, the number of layers of graph convolution is set to 3, and the number of diffusion steps K of each layer of graph convolution is 3. When learning the relationship matrix between different layers, the dimensions of matrices produced by decomposing relationship matrix is set to 50. Then we use a multi-level aggregation layer to aggregate the representations of stations generated by different layers in MCGC to get the final representation of stations. In the MCGRU module, each MCGRU block is composed of a layer of GRU, and there are 12 identical MCGRU blocks in the encoder-decoder to extract the traffic flow features at 12 historical time steps. The span of each time step is 30 minutes, such as 0: 00 am to 0: 30 am, 0: 30 am to 1: 00 am. The learning rates of Taxi and Bike are set to 0.0015 and 0.0005, respectively. The running environment used in this experiment is the Linux system host based on Ubuntu 18.04 with an 11G memory Nvidia RTX 2080Ti, which is in the Pytorch 1.0.4 framework.

### F. CONVERGENCE ANALYSIS

Fig. 5 shows the loss of training and validating of the MCGCN model using different historical time steps in the multi-step attention mechanism of the encoder. From the figure, we can observe that the overall variation trend of the errors of the models with different temporal attention steps is similar on the training and validating sets. As shown in the Fig. 5a and Fig. 5c, the error curve first drops and then tends to be stable during the training process. After about 40 epochs, the error curve is stable and does not change dramatically, which indicates that the training has reached convergence. A similar trend occurs in the validating process. We can observe from the figures that the convergence error of the model reaches the minimum when the number of time steps is set to 12. The results in Fig. 5 shows that a different number of time steps will affect the convergence of the model. In the analysis of the later results, we set time steps to 12.

### G. RESULTS ANALYSIS

Table 3 shows the comparison results of different baseline models on the data sets NYCTaxi and NYCBike. From the table, we can see that the RMSE, MAE, and PCC of the MCGCN model are improved by 5.55%, 5.62% and 0.47%, respectively, comparing with the model of the best baseline on the NYCTaxi dataset. On the NYCBike data set, the RMSE, MAE, and PCC of the MCGCN model are improved by 2.51%, 1.54% and 1.32%, respectively, compared with the second-best model. This result shows the effectiveness of the multi-step coupled graph convolution and time attention mechanism.

The model’s HA, XGBoost, and FC-LSTM only use temporal correlation to model the data, but do not employ the spatial information to improve the prediction results. From Table 3, HA, XGBoost, FC-LSTM models obtains the poor prediction results overall. For the NYCTaxi dataset, although the RMSE of STGCN is higher than XGBoost, the Pearson product-moment correlation coefficients of the STGCN model are better than XGBoost, because STGCN can autonomously learn the correlation between temporal and spatial. Both graph WaveNet and MCGCN models use adaptive relationship matrices, but MCGCN still performs better than graph WaveNet, which maybe benefit from the multi-step coupling strategy in MCGCN. MCGC module can update the relationship matrix flexibly by learning features from relationship representation of the previous layers, to improve the prediction effect. Multi-step attention mechanism can select more useful information from different historical time steps to update the output of the decoder, which may be the reason that MCGCN performs better than CCRNN.
Xiaohui Huang et al.: Preparation of Papers for IEEE TRANSACTIONS and JOURNALS

H. ABLATIONS

In the experiment, we conducted ablation experiments on our proposed model by removing or changing some modules. Specifically, there are 5 variants of our model: 1) ConvAtt: There is no coupling graph convolution in MCGC module, but existing MCGCAtt module; 2) CConvAtt: In MCGC module of CConvAtt, only one-step coupling graph convolution is used to update the relationship matrix, and a MCGCAtt module is used to capture the temporal dependence; 3) CConv: There is only single step coupling to update the relationship matrix, and no MCGCAtt module; 4) MCConv: Multi-step coupling convolution to update the relationship matrix, and without multi-step attention mechanism; 5) MCGCN: Updating the relationship matrix with multi-step coupled graph convolution in MCGC module, and using MCGCAtt module to capture the temporal dependence.

We have carried out experiments on the above five variants, and the results are shown in Fig. 6. From the figures, we can see that the MCGCN model shows the best performance in the two datasets. Multi-step coupled graph convolution can comprehensively consider the influence of relationship representation of previous layers when updating the current
relationship representation. In terms of time dependence, a multi-step attention mechanism can extract valuable information from the hidden state of historical time steps to update the state of future time steps. By integrating these two components, MCGCN can get the best prediction results.

In order to further verify the validity of the MCGCN model, we also compared the results of various variants in different prediction time steps. We only show the results of four specific time steps: the first time step (0.5 hours), the fifth time step (2.5 hours), the ninth time step (4.5 hours), and the last time step (6 hours) due to the space limitation in Fig. 7, we can see from the figures that the MCGCN model is superior to other variant methods at every time step on both data sets NYCTaxi and NYCBike, which once again proves the superiority of the MCGCN model in single-step prediction. On the NYCBike data set, the results of the MCGC model are close to MCGCN at each time step, but MCGCN still has an advantage over MCGC, which maybe benefit from the information of time dependence extracted by the multi-step attention mechanism.

VI. CONCLUSION
This work proposed a multi-step coupled graph convolution with temporal-attention for traffic flow prediction. Specifically, we designed a multi-step coupled graph convolution module to dynamically update the relationship of the stations in each graph convolution layer by learning relationship matrices of the previous layers. And then, a multi-level aggregation module is used to aggregate these layers to get the final output of MCGC. In terms of temporal features, we established a multi-step temporal attention mechanism to dynamically extract the dependencies of multiple historical time steps to better predict future traffic flow. We conducted experiments on the NYCTaxi and NYCBike data sets and the experimental results showed that the MCGCN model was superior to the baseline methods. Furthermore, the ablation experiments once again verified the effectiveness of the modules of MCGCN.

However, the multi-step coupled graph convolution module involves more matrix operations (for example, the calculation of the relationship matrix), which will bring additional computational cost. At the same time, only one type of adjacency matrix is used in the graph volume product, which can only capture the flow characteristics of a single traffic mode. In the future work, we intend to redesign a novel construction method of adjacency matrix to reduce the computational overhead in multi-step coupled graph convolution. At the same time, a variety of adjacency matrices are designed to
capture more characteristics of traffic flow, and appropriate fusion methods are used to integrate these characteristics to improve the final prediction accuracy.

ACKNOWLEDGMENT

The authors would like to thank the editors and the anonymous reviewers whose insightful comments have helped to improve the quality of this paper considerably.

REFERENCES

A. REFERENCES

[1] J. Ke, S. Feng, Z. Zhu, H. Yang, and J. Ye, “Joint predictions of multi-modal ride-hailing demands: a deep multi-task multigraph learning-based approach,” Transportation Research Part C: Emerging Technologies, vol. 117, pp. 1–15, 2021.

[2] X. Wang, X. Guan, J. Cao, N. Zhang, and H. Wu, “Forecast network-wide traffic states for multiple steps ahead: A deep learning approach considering dynamic non-local spatial correlation and non-stationary temporal dependency,” Transportation Research Part C: Emerging Technologies, vol. 119, pp. 1–29, 2020.

[3] J. Li, F. Guo, A. Sivakumar, Y. Dong, and R. Krishnan, “Transferability improvement in short-term traffic prediction using stacked LSTM network,” Transportation Research Part C: Emerging Technologies, vol. 124, 2021.

[4] J. Zhang, X. Long, D. Qi, R. Li, X. Yi, and T. Li, “Predicting citywide crowd flows using deep spatio-temporal residual networks,” Artificial Intelligence, pp. 147–166, 2018.

[5] Z. Zhao, W. Chen, X. Wu, C. Y. P. Chen, and J. Liu, “LSTM network: a deep learning approach for short-term traffic forecast,” IET Intelligent Transport Systems, pp. 68–75, 2017.

[6] Y. Li, R. Yu, C. Shahabi, and Y. Liu, “Diffusion convolutional recurrent neural network: Data-driven traffic forecasting,” arXiv preprint arXiv:1707.01926, pp. 1–17, 2017.

[7] B. Yu, H. Yin, and Z. Zha, “Spatio-Temporal graph convolutional networks: a deep learning framework for traffic forecasting,” in Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3634–3640.

[8] T. Alghamdi, K. Elgazzar, M. Bayoumi, T. Sharaf, and S. Shah, “Forecasting traffic congestion using ARIMA modeling,” in Proceedings of the 15th International Wireless Communications and Mobile Computing Conference, 2019, pp. 1–7.

[9] R. E. Engle, “Autoregressive conditional heteroskedasticity with estimates of the variance of national unity inflation,” Econometrica, vol. 50, no. 3, pp. 987–1008, 1982.

[10] C. Ding, J. Duan, Y. Zhang, X. Wu, and G. Yu, “Using an ARIMA-GARCH modeling approach to improve subway short-term ridership forecasting accounting for dynamic volatility,” IEEE Transactions on Intelligent Transportation Systems, pp. 1054–1064, 2018.

[11] T. Bollerslev, “Generalized autoregressive conditional heteroskedasticity,” Econometrica, vol. 59, no. 3, pp. 307–327, 1986.

[12] X. Lin and Y. Huang, “Short-term high-speed traffic flow prediction based on ARIMA-GARCH model,” Wireless Personal Communications, vol. 117, pp. 3421–3430, 2021.

[13] X. Xiao and H. Duan, “A new grey model for traffic flow mechanics,” Engineering Applications of Artificial Intelligence, vol. 88, pp. 1–12, 2020.

[14] M. Bai, Y. Lin, M. Ma, P. Wang, and L. Duan, “Prept: Traffic congestion prediction in smart cities with relative position congestion tensor,” Neurocomputing, vol. 444, pp. 147–157, 2021.

[15] X. Wang, Q. Zhu, P. Li, X. Chen, and Q. Niu, “Fine-grained predicting urban crowd flows with adaptive spatio-temporal graph convolutional networks,” Neurocomputing, vol. 446, pp. 95–105, 2021.

[16] Z. Zhang, Y. Li, H. Song, and H. Dong, “Multiple dynamic graph based traffic speed prediction method,” Neurocomputing, vol. 461, pp. 109–117, 2021.

[17] X. Dong, T. Lei, S. Jin, and Z.-S. Hou, “Short-term traffic flow prediction based on XGBoost,” in Proceedings of the IEEE 7th Data Driven Control and Learning Systems Conference, 2018, pp. 854–859.

[18] W. Wu, Y. Xia, and W. Jin, “Predicting bus passenger flow and prioritizing influential factors using multi-source data: Scaled stacking gradient boosting decision trees,” IEEE Transactions on Intelligent Transportation Systems, pp. 2510–2523, 2021.

[19] S. R. Chandra and H. Al-Deek, “Predictions of freeway traffic speeds and volumes using vector autoregressive models,” Journal of Intelligent Transportation Systems, pp. 53–72, 2009.

[20] Y. Xie, Y. Zhang, and Z. Ye, “Short-term traffic volume forecasting using kalman filter with discrete wavelet decomposition,” Computer-Aided Civil and Infrastructure Engineering, pp. 326–334, 2007.

[21] J. Ye, J. Zhao, K. Ye, and C. Xu, “How to build a graph-based deep learning architecture in traffic domain: A survey,” IEEE Transactions on Intelligent Transportation Systems, pp. 1–21, 2020.

[22] R. Law, G. Li, D. Fong, and X. Han, “Tourism demand forecasting: A deep learning approach,” Annals of Tourism Research, vol. 75, pp. 410–423, 2019.

[23] X. Ma, H. Zhong, Y. Li, J. Ma, and Y. Wang, “Forecasting transportation network speed using deep capsule networks with nested LSTM models,” IEEE Transactions on Intelligent Transportation Systems, vol. PP, no. 99, pp. 1–12, 2020.

[24] B. Du, X. Hu, L. Sun, J. Liu, Y. Qiao, and W. Lv, “Traffic demand prediction based on dynamic transition convolutional neural network,” IEEE Trans. Intell. Transp. Syst., pp. 1237–1247, 2021.

[25] Y. Lee, H. Jeon, and K. Sohn, “Predicting short-term traffic speed using a deep neural network to accommodate citywide Spatio-Temporal correlations,” IEEE Transactions on Intelligent Transportation Systems, pp. 1435–1448, 2021.

[26] R. Ma, S. Chen, and M. H. Zhang, “Time series relations between parking garage occupancy and traffic speed in macroscopic downtown areas - a data driven study,” Journal of Intelligent Transportation Systems, pp. 423–438, 2021.

[27] H. Cui, G. Yuan, N. Liu, M. Xu, and H. Song, “Convolutional neural network for recognizing highway traffic congestion,” Journal of Intelligent Transportation Systems, vol. 24, pp. 279–289, 2020.

[28] A. Krizhevsky, I. Sutskever, and E. G. Hinton, “Imagenet classification with deep convolutional neural networks,” Communications of the ACM, pp. 84–90, 2017.

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, E. S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” computer vision and pattern recognition, pp. 1894–1903, 2015.

[30] R. W. Liu, M. Liang, J. Nie, W. Y. B. Lim, Y. Zhang, and M. Guizani, “Deep learning-powered vessel trajectory prediction for improving smart traffic services in maritime internet of things,” IEEE Transactions on Network Science and Engineering, 2022.

[31] Y. Zhang, S. Wang, B. Chen, J. Cao, and Z. Huang, “Trafficgan: Network-scale deep traffic prediction with generative adversarial nets,” Trans. Intell. Transp. Syst., vol. 22, no. 1, pp. 219–230, 2021.

[32] S. Wang, M. Zhang, H. Miao, and P. S. Yu, “Mt-streets: Multi-task spatial-temporal networks for multi-scale traffic prediction,” in Proceedings of the 2021 SIAM International Conference on Data Mining (SDM), SIAM, 2021, pp. 504–512.

[33] B. Du, H. Peng, S. Wang, M. Z. A. Bhiyani, L. Wang, Q. Gong, L. Liu, and J. Li, “Deep irregular convolutional residual lstm for urban traffic passenger flows prediction,” IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 3, pp. 972–985, 2019.

[34] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, “Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting,” in Proceedings of the 33th AAAI conference on artificial intelligence, vol. 33, pp. 3656–3663.

[35] H. Wang, M. Zhao, X. Xie, W. Ye, and M. Guo, “Knowledge graph convolutional networks for recommender systems,” arXiv preprint arXiv:1904.12575, pp. 1–7, 2019.

[36] D. Nathani, J. Chauhan, C. Sharma, and K. Maul, “Learning attention-based embeddings for relation prediction in knowledge graphs,” in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 4710–4723.

[37] M. Luo, B. Du, K. Klemmer, H. Zhu, and H. Wen, “D3P: Data-driven demand prediction for fast expanding electric vehicle sharing systems,” Proceedings of the ACM on Interactive Mobile Wearable and Ubiquitous Technologies, vol. 4, pp. 1–21, 2020.

[38] K. Guo, Y. Hu, Z. Qian, H. Liu, K. Zhang, Y. Sun, J. Gao, and B. Yin, “Optimized graph convolution recurrent neural network for traffic prediction,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, pp. 1138–1149, 2021.

[39] Q. Song, R. Bing, J. Hu, H. Niu, and M. Gao, “Graph attention convolutional network - spatiotemporal modeling for urban traffic prediction,”
XIAOHUI HUANG received the BEng degree and the master’s degree from Jiangxi Normal University, Nanchang, P.R. China, in 2005 and 2008, respectively, and the PhD degree in the Harbin Institute of Technology, P.R. China, in 2014. Since Dec, 2014, he has been at the School of Information Engineering, East China Jiaotong University, Nanchang, China, where he is currently an associate professor of Computer Science. His research interests are in the areas of machine learning, deep learning and clustering algorithm.

YUNMING YE received the B.S. degree in software engineering from University of Electronic Science and Technology of China, Zhongshan Institute, in 2020. He is currently pursuing the M.S. degree in computer technology at East China Jiaotong University, Nanchang, China. He is currently a researcher of the Institute for Data Science and Deep Learning, East China Jiaotong University. He has won the second prize of Jiangxi Provincial Data Innovation Application Competition 2020. His research interests include deep learning, and traffic flow prediction.

XIAOFEI YANG received a B.Sc. from Suihua University in 2011. And he received M.Sc. and Ph.D. degrees from Harbin Institute of Technology in 2014 and 2019, respectively. Currently, he is a research fellow at the University of Macau. His research interests are in the areas of deep learning, remote sensing, and graph mining.

LIYAN XIONG received the B.S. degree in computer science from Jiangxi Normal University, Nanchang, China, in 1997 and the M.S. degree in computer application from East China Jiaotong University, Nanchang, in 2007. She is now a professor of the school of information engineering, East China Jiaotong University, and a graduate supervisor of the Institute of data science and deep learning, East China Jiaotong University. Her research interests include database system, natural language processing, decision support system and data mining.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/