Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Forecasting traffic flows in irregular regions with multi-graph convolutional network and gated recurrent unit*

Dewen SENG, Fanshun LV, Ziyi LIANG, Xiaoying SHI‡, Qiming FANG

School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China

E-mail: sengdw@hdu.edu.cn; 172050041@hdu.edu.cn; liangziyi2020@163.com;
shixiaoying@hdu.edu.cn; fangqiming@hdu.edu.cn

Received May 21, 2020; Revision accepted Feb. 17, 2021; Crosschecked Apr. 1, 2021; Published online July 29, 2021

Abstract: The prediction of regional traffic flows is important for traffic control and management in an intelligent traffic system. With the help of deep neural networks, the convolutional neural network or residual neural network, which can be applied only to regular grids, is adopted to capture the spatial dependence for flow prediction. However, the obtained regions are always irregular considering the road network and administrative boundaries; thus, dividing the city into grids is inaccurate for prediction. In this paper, we propose a new model based on multi-graph convolutional network and gated recurrent unit (MGCN-GRU) to predict traffic flows for irregular regions. Specifically, we first construct heterogeneous inter-region graphs for a city to reflect the relationships among regions. In each graph, nodes represent the irregular regions and edges represent the relationship types between regions. Then, we propose a multi-graph convolutional network to fuse different inter-region graphs and additional attributes. The GRU is further used to capture the temporal dependence and to predict future traffic flows. Experimental results based on three real-world large-scale datasets (public bicycle system dataset, taxi dataset, and dockless bike-sharing dataset) show that our MGCN-GRU model outperforms a variety of existing methods.

Key words: Traffic flow prediction; Multi-graph convolutional network; Gated recurrent unit; Irregular regions
https://doi.org/10.1631/FITEE.2000243 **CLC number:** TP391

1 Introduction

An intelligent transportation system is a significant part of a smart city, and traffic forecasting plays an important role in intelligent transportation control and management (Zhu et al., 2019). The goal of region-level traffic forecasting is to predict the future flow number of regions in a city given historical observation. Accurate traffic forecasting provides a scientific basis for traffic managers in controlling flow numbers.

This task is challenging because of the complicated spatio-temporal dependencies among regions:

(1) Spatial dependence: Complicated dependencies exist among different regions, and are influenced by not only the topological structure of the urban road network, but also the proximity and usage patterns of regions;

(2) Temporal dependence: Traffic flows in one region change dynamically over time. These flows not only have periodic patterns, but also are affected by the flow conditions of previous time steps.

Traditional time-series forecasting methods, such as auto-regressive integrated moving average (ARIMA) (Box et al., 2015), time-varying Poisson model (Moreira-Matias et al., 2013), and vector autoregressive model (Chandra and Al-Deek, 2009), consider only the temporal dependence and have relatively low prediction accuracy. With the boom of

---

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (No. 61903109) and the Zhejiang Provincial Natural Science Foundation of China (No. LY19F030021)
ORCID: Dewen SENG, https://orcid.org/0000-0003-0921-848X; Xiaoying SHI, https://orcid.org/0000-0003-0452-2503

deep learning techniques, researchers tried to adopt them for flow prediction. Long short-term memory (LSTM) and gated recurrent unit (GRU) models achieved better performance for short-term time-series prediction than traditional methods (Fu et al., 2016). However, they also focused only on temporal correlation.

To better capture spatio-temporal correlations, deep learning based models, such as the convolutional neural network (CNN) (Zhang et al., 2016) and residual neural network (Zhang et al., 2018), have been introduced. The city was divided into grids and the grid-level flows were predicted (Zhang et al., 2016, 2018). However, such methods focused on modeling the Euclidean correlations among regular grids. The city can be divided into meaningful regions according to the road network or administrative boundaries. These regions are always irregular with a complex topological structure. Grid-level prediction models failed to predict the demands of irregular regions, decreasing the effectiveness of the predicted results. In addition, most existing methods (Li et al., 2018; Yu B et al., 2018; Zhao et al., 2020) use only the distance to represent the spatial relationship, but ignore the multiple complex relationships among regions.

In this study, we propose a novel deep learning based prediction model, i.e., multi-graph convolutional network and gated recurrent unit (MGCN-GRU), to predict traffic flows in irregular regions. We view the traffic flows in a city from the graph perspective. Three heterogeneous inter-region graphs encoding various spatial dependencies are constructed first. A multi-graph convolutional network is proposed to fuse the complex non-Euclidean correlations among irregular regions. Then, a GRU is built to capture the dynamic temporal patterns. Hence, both the spatial and temporal patterns are effectively captured for region-level traffic flow prediction. Experimental results on three real-world datasets demonstrate that our model can effectively capture the spatio-temporal correlations from traffic data, and that our MGCN-GRU model outperforms a variety of existing methods.

Two contributions are made in this paper:

1. We encode multiple non-Euclidean pair-wise correlations among irregular regions using multiple graphs, and propose a novel prediction model to use these spatial correlations to predict traffic flows in irregular regions.

2. Three real-world traffic datasets (public bicycle system dataset, taxi dataset, and dockless bike-sharing dataset) are used to evaluate our model, and the experimental results show that our MGCN-GRU model outperforms a variety of existing methods.

2 Literature review

2.1 Traffic flow prediction

Traffic flow prediction is important for traffic management. Traffic-related datasets have been used to predict taxi demand (Moreira-Matias et al., 2013; Yao et al., 2018), traffic speed (Li et al., 2018; Wang et al., 2018; Yu B et al., 2018; Kim et al., 2019), and bicycle flows (Kaltenbrunner et al., 2010; Yoon et al., 2012). Traditional approaches use time-series forecasting techniques such as ARIMA, seasonal ARIMA (Williams and Hoel, 2003), and time-varying Poisson model (Moreira-Matias et al., 2013). In recent years, deep learning based methods have been applied as novel alternatives for traffic flow prediction. LSTM (Tian and Pan, 2015) and GRU (Fu et al., 2016) models have been adopted to predict short-term traffic flows, which perform better than traditional models. Yu R et al. (2017) augmented the LSTM model with stacked autoencoder accounting for accident features to forecast extreme traffic conditions. However, the above methods focus only on the temporal correlation.

To capture the spatial and temporal dependencies simultaneously, Zhang et al. (2016) partitioned a city into grids and proposed a deep neural network based prediction model including spatio-temporal and global components. CNN was used to capture spatio-temporal dependencies. Zhang et al. (2018) further proposed ST-ResNet, using a residual neural network to forecast the inflow number and outflow number for each region. Yao et al. (2018) proposed a deep multi-view spatial-temporal network to predict taxi demand. The above methods (Zhang et al., 2016, 2018; Yao et al., 2018) predict traffic flows in regular grids. However, partitioning the city into grids is inaccurate for flow prediction. Human activity regions are usually irregular, influenced by road networks or administrative boundaries. In our study, we use a multi-graph convolution to model the spatial relationships among irregular regions explicitly, so that the scope of prediction is no longer limited to regular grids.

2.2 Graph convolutional networks

Generalizing CNN to the graph convolutional network (GCN) can enable the handling of arbitrary graph-structured data, which has received widespread attention recently. GCN was first introduced by Bruna et al. (2014), in which convolutional layers were applied to the graph data. It was later extended by Defferrard et al. (2016) with fast localized convolutions to accelerate the computation process. Kipf and Welling (2017) proposed an efficient variant of CNN which can be used directly on graphs, and the network achieved good performance on graph node classification tasks. Seo et al. (2018) proposed a graph CNN to deal with structured sequence data.

GCN has been successfully used in many applications, including semi-supervised classification (Kipf and Welling, 2017), system recommendation (Monti et al., 2017; Ying et al., 2018), and traffic prediction. For traffic prediction, Zhao et al. (2020) proposed a temporal GCN (T-GCN), combining the GCN with GRU to predict the traffic speed on the road. Li et al. (2018) and Yu B et al. (2018) applied GCN to predict the traffic speed in road segments. These methods (Li et al., 2018; Yu B et al., 2018; Zhao et al., 2020) construct only one kind of graph (e.g., distance) to represent the spatial relationship. Chai et al. (2018) constructed multiple graphs to reflect the heterogeneous relationships among stations, and proposed a multi-graph CNN model to predict bike flows at the station level. As one graph may not be able to describe inter-region relationships comprehensively, we construct multiple graphs and further design a multi-graph convolutional network to capture the heterogeneous spatial relationships among regions. In addition, different from Chai et al. (2018), in which only the original adjacent matrices were calculated, we further set different thresholds for different graphs, and investigate the impact of the threshold on the model performance.

3 Problem definition and framework overview

3.1 Problem definition

Our goal in traffic forecasting is to predict traffic flows over a certain period of time based on historical traffic data.

Definition 1 (Traffic flows) Traffic flows include two types, i.e., inflows and outflows. Supposing that we divide the city into N disjoint irregular regions, the inflows in all regions in the time interval t (e.g., 1 h) can be denoted as $\boldsymbol{I}^t=[\mathrm{ri}_1^t,\mathrm{ri}_2^t,\ldots,\mathrm{ri}_N^t]$, where ri_i^t stands for the inflows of region i in the time interval t. Similarly, the outflows in the time interval t can be denoted as $\boldsymbol{O}^t=[\mathrm{ro}_1^t,\mathrm{ro}_2^t,\ldots,\mathrm{ro}_N^t]$, and ro_i^t stands for the outflows of region i in the time interval t.

Definition 2 (Inter-region graph) The inter-region graph encodes the spatial relationship between irregular regions, and is defined as a weighted graph $G=(V, E)$. The nodes $v_i\in V$ in the graph represent the irregular regions, and $|V|=N$. The edges $(v_i, v_j)\in E$ encode the pair-wise relationships among regions. The weights of the edges encode the relationship strength between regions, and are represented by an adjacency matrix $\boldsymbol{A}\in\mathbb{R}^{N\times N}$.

Based on the above definitions, the traffic flow prediction problem for irregular regions can be formulated as follows: Given inputs with the history data $[(\boldsymbol{I}^0, \boldsymbol{O}^0), (\boldsymbol{I}^1, \boldsymbol{O}^1), \ldots, (\boldsymbol{I}^{t-1}, \boldsymbol{O}^{t-1})]$, a function $f(\cdot)$ that maps the historical inflows and outflows of all irregular regions to the inflows and outflows at the next time step is learned:

$$\left[(\boldsymbol{I}^0,\boldsymbol{O}^0),(\boldsymbol{I}^1,\boldsymbol{O}^1),\ldots,(\boldsymbol{I}^{t-1},\boldsymbol{O}^{t-1})\right]\xrightarrow{f(\cdot)}(\hat{\boldsymbol{I}}^t,\hat{\boldsymbol{O}}^t), \quad (1)$$

aiming to minimize $\sqrt{(\hat{\boldsymbol{I}}^t-\boldsymbol{I}^t)^2+(\hat{\boldsymbol{O}}^t-\boldsymbol{O}^t)^2}$, where $(\boldsymbol{I}^t, \boldsymbol{O}^t)$ and $(\hat{\boldsymbol{I}}^t,\hat{\boldsymbol{O}}^t)$ are the ground truth and predicted traffic flow numbers of the next time step t, respectively.

3.2 Framework overview

The system architecture of the proposed MGCN-GRU model is shown in Fig. 1. As seen from the right side of Fig. 1, we first use the historical time-series data as inputs, and a multi-graph convolutional network is designed to capture the spatial dependency for each time step. Details of MGCN are illustrated on the left side of Fig. 1. Based on the segmented irregular regions, we simplify the data from the spatial and temporal dimensions and calculate the inter-region graphs to encode pair-wise correlations between regions, including distance graph,

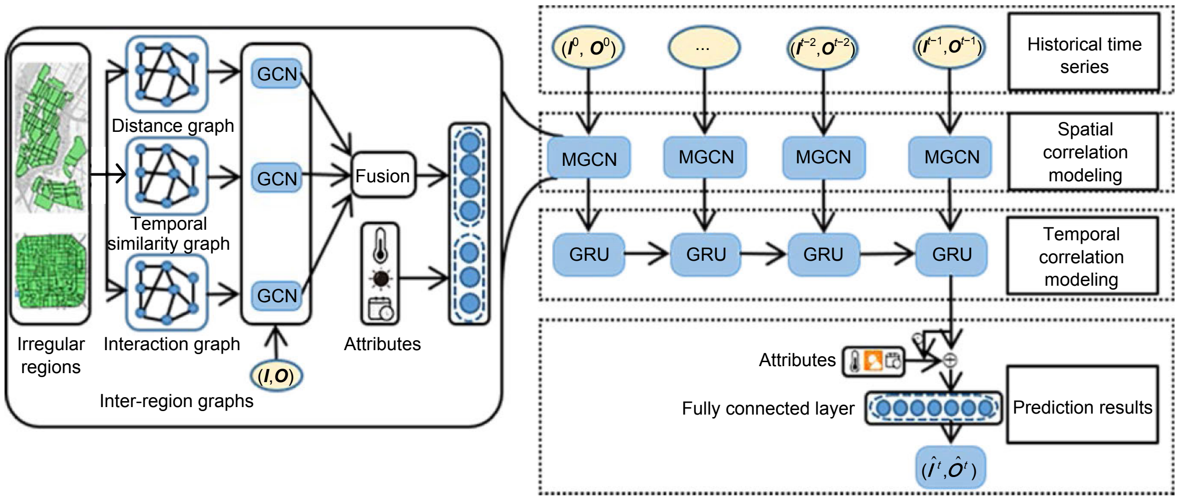


Fig. 1 Architecture of the proposed multi-graph convolutional network and gated recurrent unit (MGCN-GRU) model

temporal similarity graph, and interaction graph. The inter-region graphs are processed by the GCN model and combined with the additional time and weather attributes to obtain the multi-graph convolutional results. These calculation results are further fed into the GRU model. The temporal dynamic pattern is captured by transmitting information between units. The observations in different time steps are aggregated. Then, the additional attribute is concatenated with the output of the GCN model, and fed into the fully connected layer to generate the prediction results.

4 Method

In this section, we describe the proposed novel prediction model MGCN-GRU in detail.

4.1 Datasets and data pre-processing

Three real-world traffic datasets are used for model evaluation. Here, we first introduce the datasets used in our study.

1. Public bicycle system (PBS) dataset (https://www.citibikenyc.com/system-data). The PBS dataset was taken from the bike system in New York City (NYC), from Jan. 1, 2014 to Dec. 31, 2014. The PBS dataset includes two kinds of records, station records and trip records. Station records contain attributes of stations, such as station name, longitude, and latitude. Each trip record contains the start station, start time, end station, and end time.

2. Taxi dataset (https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page). The taxi dataset contains taxi trip records of NYC from Jan. 1, 2014 to Dec. 31, 2014. Each trip record captures the pick-up/drop-off timestamps and the corresponding Global Positioning System coordinates.

3. Dockless bike-sharing (DBS) dataset (https://www.biendata.com/competition/mobike/). The DBS dataset contains dockless bicycle usage records from May 10, 2017 to May 24, 2017 in Beijing. Each record represents the flow from a start location to a destination location, containing user ID, bike ID, start time, start location, and end location. Since the start and end locations are encoded by geohash, a decode function is adopted to obtain the latitude and longitude coordinates for locations.

The original datasets are pre-processed first, including region segmentation and spatial-temporal simplification.

1. Region segmentation

The first step is to divide the city into irregular regions. The region boundaries can be provided by users directly, or generated by an irregular map segmentation method (Yuan et al., 2015) if road network data is available. In our experiments, the region boundaries of NYC are generated by the map segmentation method, and the boundaries of Beijing are provided by the user. We obtain 94 and 363 irregular regions for NYC and Beijing, respectively. The segmented maps can be seen on the left side of Fig. 1.

2. Spatial-temporal simplification

The entire time period is split into time steps (e.g., hour), and the original traffic records are mapped into regions on an hourly basis. For PBS dataset, all stations are mapped into the corresponding regions. For taxi and DBS datasets, the start and end locations are mapped into regions. Hence, different traffic datasets can be abstracted as a uniform form, and we obtain the following simplified trajectories:

$$\begin{aligned}\mathbf{TR}_{\text{Simp}} = [&\text{startRegion}, \text{startDate}, \text{startHour}, \\ &\text{endRegion}, \text{endDate}, \text{endHour}],\end{aligned} \tag{2}$$

which means that a person departs from the startRegion on startDate of startHour and arrives at the endRegion on endDate of endHour. startRegion and endRegion are region IDs.

Then, we aggregate trajectories to calculate the inflows and outflows. A record of $\mathbf{TR}_{\text{inflow}}$ represents the inflow number ending at region i on I_{Date} of I_{Hour}, which is represented as $\mathbf{TR}_{\text{inflow}}=[I_{\text{Date}}, I_{\text{Hour}}, i, \text{num}]$. From this, we can obtain the number of inflows ri_i^t in region i during the t^{th} time step. A record of $\mathbf{TR}_{\text{outflow}}$ represents the outflow number starting from region i on O_{Date} of O_{Hour}, which is represented as $\mathbf{TR}_{\text{outflow}}=[O_{\text{Date}}, O_{\text{Hour}}, i, \text{num}]$. From this, we can obtain the number of outflows ro_i^t in region i during the t^{th} time step. Based on ri_i^t and ro_i^t, $\boldsymbol{I}^t$ and $\boldsymbol{O}^t$ are calculated. For the taxi and PBS datasets, $\boldsymbol{I}^t$ and $\boldsymbol{O}^t$ are obtained. For the DBS dataset, only $\boldsymbol{O}^t$ is calculated since the end time is not available.

4.2 Construction of multiple inter-region graphs

After calculating the regional inflows and outflows, we build different inter-region graphs to represent various spatial relationships among regions that can help improve our prediction accuracy. Specifically, they are distance graph, temporal similarity graph, and interaction graph.

1. Distance graph

Since nearby regions may share similar usage patterns, the distance graph is constructed. The edge weight is the distance between two regions. The element in the adjacency matrix of the distance graph is calculated as follows:

$$A_{\text{d}}(i,j) = \begin{cases} \text{dist}(i,j), & \text{if } i \neq j, \\ 0, & \text{if } i = j, \end{cases} \tag{3}$$

where $\text{dist}(i,j)$ is the distance between the centroids of region i and region j. The adjacency matrix $\boldsymbol{A}_{\text{d}}$ is further normalized to [0, 1] and transformed to a 0/1 matrix based on the predefined distance threshold thres_{d}. If $A_{\text{d}}(i,j)\le\text{thres}_{\text{d}}$, which means that the distance between region i and region j is very short, then $A_{\text{d}}(i,j)=1$; otherwise, $A_{\text{d}}(i,j)=0$.

2. Temporal similarity graph

Historical records contain regions' usage characteristics. Temporal demand correlations between regions are calculated to generate the temporal similarity graph. Specifically, the historical usage numbers of each region in each time step (1 h) are calculated. Taking the PBS dataset as an example, each region has a time series of 17 520 (365×24×2) hourly traffic flow values for a one-year period. The time series for region i can be expressed as $\boldsymbol{h}_i = [\text{ri}_i^0, \text{ri}_i^1, \ldots, \text{ri}_i^{8759}, \text{ro}_i^0, \text{ro}_i^1, \ldots, \text{ro}_i^{8759}]$. Then, the similarity between every two regions is calculated by the Pearson correlation coefficient (PCC) based on the hourly time series between region i and region j. An element in the adjacency matrix of the temporal similarity graph $A_{\text{tSimi}}(i,j)$ is calculated as follows:

$$A_{\text{tSimi}}(i,j) = \begin{cases} \text{PCC}(\boldsymbol{h}_i, \boldsymbol{h}_j), & \text{if } i \neq j, \\ 0, & \text{if } i = j, \end{cases} \tag{4}$$

where $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ are the hourly time series for region i and region j, respectively. The adjacency matrix $\boldsymbol{A}_{\text{tSimi}}$ is further normalized to [0, 1] and transformed to a 0/1 matrix based on the predefined threshold $\text{thres}_{\text{tSimi}}$. If $A_{\text{tSimi}}(i, j)\ge\text{thres}_{\text{tSimi}}$, which means that region i and region j have similar temporal usage patterns, then $A_{\text{tSimi}}(i,j)=1$; otherwise, $A_{\text{tSimi}}(i,j)=0$.

3. Interaction graph

The bi-directional flows provide plenty of information. If the flow number between two regions is very large, these two regions tend to affect each other in their dynamic traffic flow patterns. Therefore, an interaction graph is constructed to indicate whether two regions have interacted with each other frequently according to the historical records. The interactions include bi-directional flows between two regions over a period of time. By aggregating $\mathbf{TR}_{\text{Simp}}$, we can calculate the flow number $\text{fn}(i,j)$ starting from region i and ending at region j and the flow number $\text{fn}(j, i)$ starting from region j and ending at region i during the whole time period. Then, the elements in the adjacency matrix of interaction graph $A_{\text{inter}}(i,j)$ are calculated. To handle the graph convolution in a unified form, we set the entries on the main diagonal as 0:

$$A_{\text{inter}}(i,j)=\begin{cases}\text{fn}(i,j)+\text{fn}(j,i), & \text{if } i\neq j,\\ 0, & \text{if } i=j.\end{cases} \tag{5}$$

The adjacency matrix $\boldsymbol{A}_{\text{inter}}$ is further normalized to [0, 1] and transformed to a 0/1 matrix based on the predefined interaction threshold $\text{thres}_{\text{inter}}$. If $A_{\text{inter}}(i, j)\geq\text{thres}_{\text{inter}}$, which means that the interaction between region i and region j is strong, then $A_{\text{inter}}(i,j)=1$; otherwise, $A_{\text{inter}}(i,j)=0$.

It should be noted that we set the diagonal elements of all adjacency matrices as 0 in this phase. An identity matrix will be added to the adjacency matrices in the following phase, to ensure that the connection from region i to itself is the strongest.

4.3 Multi-graph convolutional network

After constructing the inter-region graphs, we propose a multi-graph convolutional network to fully exploit different graphs that contain useful spatial correlation information and to capture the spatial dependency. The multi-graph convolutional network can fuse features of various spatial relationships with some additional attributes for different time steps. The process contains three steps:

Step 1: Each graph for the $(t')^{\text{th}}$ time step is processed by a GCN model $f(\boldsymbol{X}^{t'},\boldsymbol{A})$:

$$f_{\text{d}}(\boldsymbol{X}^{t'},\boldsymbol{A}_{\text{d}})=\sigma\left(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}_{\text{d}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{W}_{\text{d}}\boldsymbol{X}^{t'}\right), \tag{6}$$

$$f_{\text{tSimi}}(\boldsymbol{X}^{t'},\boldsymbol{A}_{\text{tSimi}})=\sigma\left(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}_{\text{tSimi}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{W}_{\text{tSimi}}\boldsymbol{X}^{t'}\right), \tag{7}$$

$$f_{\text{inter}}(\boldsymbol{X}^{t'},\boldsymbol{A}_{\text{inter}})=\sigma\left(\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}_{\text{inter}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{W}_{\text{inter}}\boldsymbol{X}^{t'}\right), \tag{8}$$

where $\boldsymbol{X}^{t'}=\left[\boldsymbol{I}^{t'},\boldsymbol{O}^{t'}\right]$ is the model input at time t' ($t'\in[0, t-1]$). Such input can be regarded as the traffic flow feature for each node at t'. The elements of the feature vector of each node are the inflow number and outflow number in the $(t')^{\text{th}}$ time step for a region. Thus, the dimension of the feature vector is 2. $\tilde{\boldsymbol{A}}=\boldsymbol{A}+\boldsymbol{I}_N$ is an adjacency matrix with self-loops, where $\boldsymbol{I}_N$ is an identity matrix. For example, $\tilde{\boldsymbol{A}}_{\text{d}}=\boldsymbol{A}_{\text{d}}+\boldsymbol{I}_N$ is the distance graph. Adding an identity matrix $\boldsymbol{I}_N$ can ensure that the distance correlation from region i to itself is the strongest. Similarly, we set $\tilde{\boldsymbol{A}}_{\text{tSimi}}=\boldsymbol{A}_{\text{tSimi}}+\boldsymbol{I}_N$ for the temporal similarity graph and $\tilde{\boldsymbol{A}}_{\text{inter}}=\boldsymbol{A}_{\text{inter}}+\boldsymbol{I}_N$ for the interaction graph. $\tilde{\boldsymbol{D}}$ is a diagonal matrix with $\tilde{D}_{ii}=\sum_j\tilde{A}_{ij}$. $\boldsymbol{W}_{\text{d}}$, $\boldsymbol{W}_{\text{tSimi}}$, and $\boldsymbol{W}_{\text{inter}}$ are the trainable weight matrices. $\sigma(\cdot)$ represents the activation function, and we use tanh in the experiments.

Step 2: The additional attribute $\mathbf{attr}^{t'}$ for each time step is calculated, since the traffic flow in each time step has unique influence factors. The additional attributes include the date attribute $\mathbf{attr}^{t'}_{\text{date}}$ (the day of the week, 7-dimensional), hour attribute $\mathbf{attr}^{t'}_{\text{hour}}$ (the time of the day, 24-dimensional), weather attribute $\mathbf{attr}^{t'}_{\text{wea}}$, and temperature attribute $\mathbf{attr}^{t'}_{\text{temp}}$. The weather attribute is divided into eight categories, including sun, cloud, light rain, moderate rain, heavy rain, light snow, moderate snow, and heavy snow. The temperature attribute is divided into eight levels, from 10 °F to 90 °F. Every 10 °F corresponds to one level. All these attributes use one-hot encoding, and are concatenated as a binary vector $\mathbf{attr}^{t'}=[\mathbf{attr}^{t'}_{\text{date}},\mathbf{attr}^{t'}_{\text{hour}},\mathbf{attr}^{t'}_{\text{wea}},\mathbf{attr}^{t'}_{\text{temp}}]$.

Step 3: Multiple inter-region graphs are fused by weight matrices, and the fusion operation is defined as follows:

$$\begin{aligned}\boldsymbol{G}^{t'}_{\text{fusion}}=&\boldsymbol{W}'_{\text{d}}\odot f_{\text{d}}(\boldsymbol{X}^{t'},\boldsymbol{A}_{\text{d}})+\boldsymbol{W}'_{\text{tSimi}}\odot f_{\text{tSimi}}(\boldsymbol{X}^{t'},\boldsymbol{A}_{\text{tSimi}})\\&+\boldsymbol{W}'_{\text{inter}}\odot f_{\text{inter}}(\boldsymbol{X}^{t'},\boldsymbol{A}_{\text{inter}}),\end{aligned} \tag{9}$$

where $\boldsymbol{W}'_{\text{d}}$, $\boldsymbol{W}'_{\text{tSimi}}$, and $\boldsymbol{W}'_{\text{inter}}$ are the weight matrices, and $\odot$ is an element-wise multiplication operator. The fusion result $\boldsymbol{G}^{t'}_{\text{fusion}}$ is further concatenated with the additional attributes by weight matrix $\boldsymbol{W}'_{\text{attr}}$, to generate the multi-graph convolution result $\mathbf{MGCN}^{t'}=[\boldsymbol{G}^{t'}_{\text{fusion}},\boldsymbol{W}'_{\text{attr}}\cdot\mathbf{attr}^{t'}]$. $\mathbf{MGCN}^{t'}$ represents the input of the GRU neural network.

4.4 GRU neural network

The GRU neural network uses a gated mechanism to memorize as much long-term information as possible, which is effective for various tasks. Compared with LSTM, the GRU model has a relatively simple structure and fewer parameters, and needs a shorter training time. Therefore, we choose the GRU model to capture the temporal dependency based on the multi-graph convolution results.

The GRU model obtains the traffic status at time t' by taking the hidden status at time $t'-1$ and the multi-graph convolution result $\mathbf{MGCN}^{t'}$ as the input. Let $\boldsymbol{h}^{t'-1}$ denote the hidden state at time $t'-1$ and $\boldsymbol{h}^{t'}$ denote the output state at time t'. The specific calculation process of the GRU is shown below:

$$\boldsymbol{u}^{t'} = \sigma\left(\boldsymbol{W}_u[\mathbf{MGCN}^{t'}, \boldsymbol{h}^{t'-1}] + \boldsymbol{b}_u\right), \tag{10}$$

$$\boldsymbol{r}^{t'} = \sigma\left(\boldsymbol{W}_r[\mathbf{MGCN}^{t'}, \boldsymbol{h}^{t'-1}] + \boldsymbol{b}_r\right), \tag{11}$$

$$\boldsymbol{c}^{t'} = \tanh\left(\boldsymbol{W}_c[\mathbf{MGCN}^{t'}, \boldsymbol{r}^{t'} \odot \boldsymbol{h}^{t'-1}] + \boldsymbol{b}_c\right), \tag{12}$$

$$\boldsymbol{h}^{t'} = \boldsymbol{u}^{t'}\boldsymbol{h}^{t'-1} + (1-\boldsymbol{u}^{t'})\boldsymbol{c}^{t'}, \tag{13}$$

where $\boldsymbol{r}^{t'}$ is the reset gate which controls how to combine the new input with the previous memory. When $\boldsymbol{r}^{t'}$ is close to $\mathbf{0}$, the hidden state is forced to ignore the previous hidden state and reset with the current input. $\boldsymbol{u}^{t'}$ is the update gate deciding how much information will be carried over from the previous hidden state to the current hidden state. $\boldsymbol{c}^{t'}$ is the memory content stored at time t', $\boldsymbol{W}_u$, $\boldsymbol{W}_r$, and $\boldsymbol{W}_c$ are the weight matrices, and $\boldsymbol{b}_u$, $\boldsymbol{b}_r$, and $\boldsymbol{b}_c$ are the bias vectors.

The output of the GRU model is further combined with the additional attributes by the weight matrix $\boldsymbol{W}_{\text{attr}}$, and processed by a fully connected layer to obtain the prediction result $(\hat{\boldsymbol{I}}^t, \hat{\boldsymbol{O}}^t)$, which is defined as follows:

$$(\hat{\boldsymbol{I}}^t, \hat{\boldsymbol{O}}^t) = \operatorname{sigmoid}\left(\boldsymbol{W}_{\text{pred}} \cdot [\boldsymbol{h}^{t-1}, \boldsymbol{W}_{\text{attr}} \cdot \mathbf{attr}^t]\right), \tag{14}$$

where $\boldsymbol{W}_{\text{pred}}$ is the weight matrix for final prediction.

4.5 Loss function

During the training process, the goal is to minimize the error between the predicted value $\hat{y}_t$ and the ground truth value y_t. We employ the smooth L1 loss function $\text{Smooth}_{\text{L1}}$ as the objective function, which has been verified as a robust loss function for regression:

$$\text{Smooth}_{\text{L1}} = \begin{cases} 0.5(y_t - \hat{y}_t)^2, & \text{if } |y_t - \hat{y}_t| \le 1, \\ |y_t - \hat{y}_t| - 0.5, & \text{otherwise.} \end{cases} \tag{15}$$

The loss function $\text{Smooth}_{\text{L1}}$ combines the desirable properties of the squared-error loss $(y_t-\hat{y}_t)^2$ near zero and the absolute-error loss $|y_t - \hat{y}_t|$ when $|y_t - \hat{y}_t|$ is greater than 1. We will compare its performance with those of the squared-error loss and absolute-error loss in Section 5.2.

5 Evaluation

In this section, we evaluate our prediction model based on real-world traffic datasets. We first introduce the experimental setting and then give the experimental results.

5.1 Experimental setting

1. Parameter setting

The datasets are split into training (70%), validation (20%), and testing (10%) sets. We use the past 10-h history data to predict the traffic flow in the next hour. The parameters are chosen based on their performance on the validation set. We set the training epoch as 300 and learning rate as 0.0005 for all datasets. The batch size is 64, 48, and 11 for the taxi, PBS, and DBS datasets, respectively. The neural network based models are implemented using PyTorch and trained via backpropagation and Adam optimization. They are trained and evaluated on a Windows server with AMD Ryzen 5-2600 CPU, 64 GB memory, and NVIDIA GeForce GTX 1080 Ti GPU. The offline training process takes about 40 min, while the inference process takes just a few seconds, which is suitable for real-time traffic flow prediction.

2. Evaluation metrics

During the evaluation process, we re-scale the predicted value back to the normal value $\hat{y}_t$, and compare it with the ground truth value y_t. The evaluation metrics for testing datasets include the root mean square error (RMSE) and mean absolute error (MAE):

$$\mathrm{RMSE}=\sqrt{\frac{1}{N}\sum_{t}(y_t-\hat{y}_t)^2}, \tag{16}$$

$$\mathrm{MAE}=\frac{1}{N}\sum_{t}\left|y_t-\hat{y}_t\right|, \tag{17}$$

where N is the number of all predicted values.

3. Baselines

We compare the performance of our MGCN-GRU model with those of the following baselines:

(1) History average (HA) model, which uses the average traffic flow number in historical periods as the prediction.

(2) ARIMA model, which is a widely used time-series prediction model.

(3) LSTM model, which is effective for modeling long-range dependencies in temporal sequences.

(4) GRU model, which is similar to LSTM with a relatively simple structure and fewer parameters.

(5) MGCN model, which uses only MGCN to capture the spatial correlation while ignoring the temporal correlation.

5.2 Experimental results

5.2.1 Prediction accuracy comparison between different loss functions

In the training process, we adopt the absolute-error loss, squared-error loss, and smooth L1 loss functions to optimize the parameters, and compare the prediction accuracies of different loss functions. Table 1 shows the prediction results. We observe that the smooth L1 loss function has a lower RMSE/MAE than the other two loss functions for all datasets. This indicates that the smooth L1 loss function is better for flow prediction tasks than the absolute-error loss and squared-error loss functions.

5.2.2 Selection of the threshold

Threshold influences the final prediction results. Therefore, choosing an optimal threshold value for different inter-region graphs is important. We use {start: step: end} to define the search space. For example, {0.1: 0.1: 0.4} represents the search space {0.1, 0.2, 0.3, 0.4}.

Fig. 2 shows the change of prediction precision with different threshold values based on the taxi dataset. We first select the threshold using a rough search space {0: 0.1: 1.0} for all graphs (Fig. 2a), and then optimize the search space according to the results generated from the first round (Fig. 2b). The RMSE predicted by the original adjacency matrix without a threshold mechanism is shown as a horizontal line in Fig. 2a. For all graphs, we find that using the original adjacency matrices without a threshold mechanism cannot obtain the best results. In addition, considering all correlations among regions ($\mathrm{thres_d}$=1, $\mathrm{thres_{tSimi}}$=0, $\mathrm{thres_{inter}}$=0) (that is to say, when all elements in the adjacency matrices are set as 1), the RMSE is very large. Therefore, the pre-definition of threshold is important. For the distance graph, it can be seen that the error is small when the threshold is [0, 0.2] (left of Fig. 2a). Then, we refine the search space as {0: 0.02: 0.20}, and finally select $\mathrm{thres_d}$ as 0.06 (left of Fig. 2b). For the temporal similarity graph, the error is small when the threshold is [0.8, 1.0] (center of Fig. 2a), and the search space is refined as {0.8: 0.02: 1.00}. We finally select $\mathrm{thres_{tSimi}}$ as 0.90 (center of Fig. 2b). The refined search space is set as {0.01: 0.01: 0.20} for the interaction graph (right of Fig. 2b), and $\mathrm{thres_{inter}}$ is selected as 0.14.

Fig. 3 shows the threshold selection results based on the PBS dataset. We adopt the same rough search space as that in the taxi dataset for the first-round selection (Fig. 3a). For the distance graph, the error is small when the threshold is [0, 0.2] (left of Fig. 3a). The refined search space is {0: 0.02: 0.20}, and finally $\mathrm{thres_d}$ is selected as 0.04 (left of Fig. 3b). For the

Table 1 Prediction accuracy comparison between different loss functions

Loss function	RMSE			MAE		
	PBS dataset	Taxi dataset	DBS dataset	PBS dataset	Taxi dataset	DBS dataset
Absolute-error function	3.2506	45.4916	11.8972	1.8023	24.5878	3.4953
Squared-error function	3.2541	45.0784	11.1388	1.8037	24.3731	3.2802
Smooth L1 function	**3.1846**	**44.7117**	**10.6384**	**1.7818**	**24.1932**	**3.0705**

RMSE: root mean square error; MAE: mean absolute error. Best results are in bold temporal similarity graph, the error is small when the threshold is [0.8, 1.0] (center of Fig. 3a). The refined search space is {0.8: 0.02: 1.00}, and $\mathrm{thres}_{\mathrm{tSimi}}$ is selected as 0.90 (center of Fig. 3b). For the interaction graph, the error is small when $\mathrm{thres}_{\mathrm{inter}}$ is 0.13 (right of Fig. 3b). Similarly, we obtain $\mathrm{thres}_{\mathrm{d}}$=0.20, $\mathrm{thres}_{\mathrm{tSimi}}$=0.70, and $\mathrm{thres}_{\mathrm{inter}}$=0.10 based on the DBS dataset (Fig. 4).

The final values of $\mathrm{thres}_{\mathrm{tSimi}}$ are relatively large, while the values of $\mathrm{thres}_{\mathrm{d}}$ and $\mathrm{thres}_{\mathrm{inter}}$ are small based on the three datasets. We further visualize the distributions of element value in different adjacency matrices of graphs based on different datasets (Figs. 5–7). The x axis represents the value interval and the y axis represents the number of elements in different intervals. As seen from Figs. 5–7, we find that the element

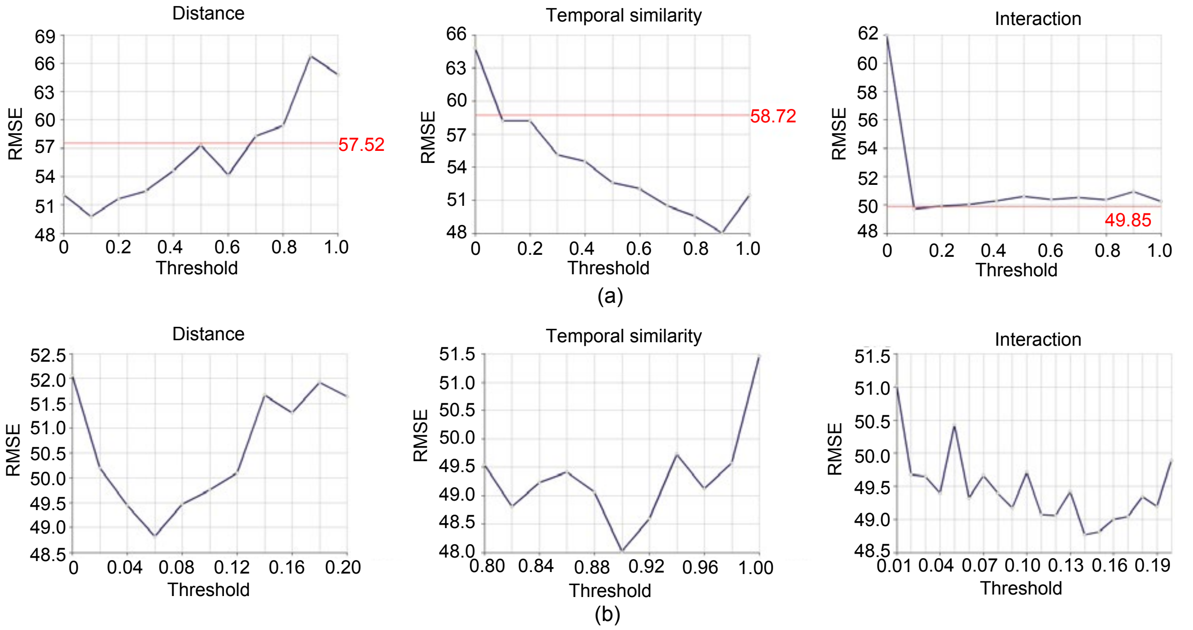


Fig. 2 Selection of different thresholds based on the taxi dataset: (a) rough search space; (b) optimized search space

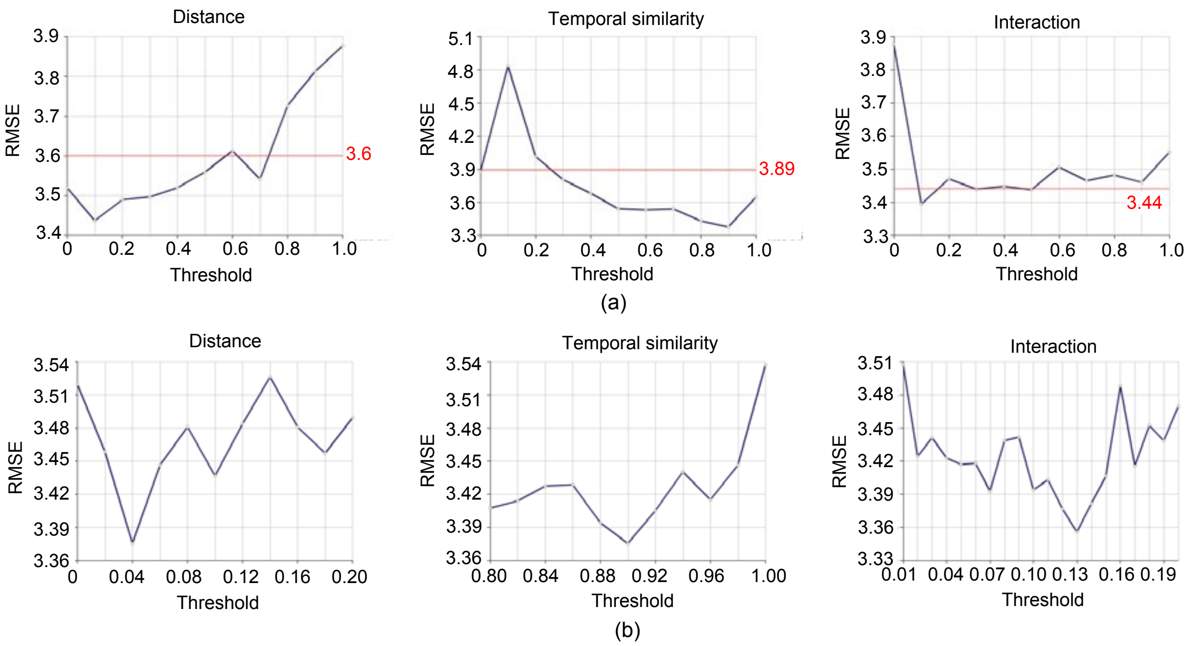


Fig. 3 Selection of different thresholds based on the PBS dataset: (a) rough search space; (b) optimized search space

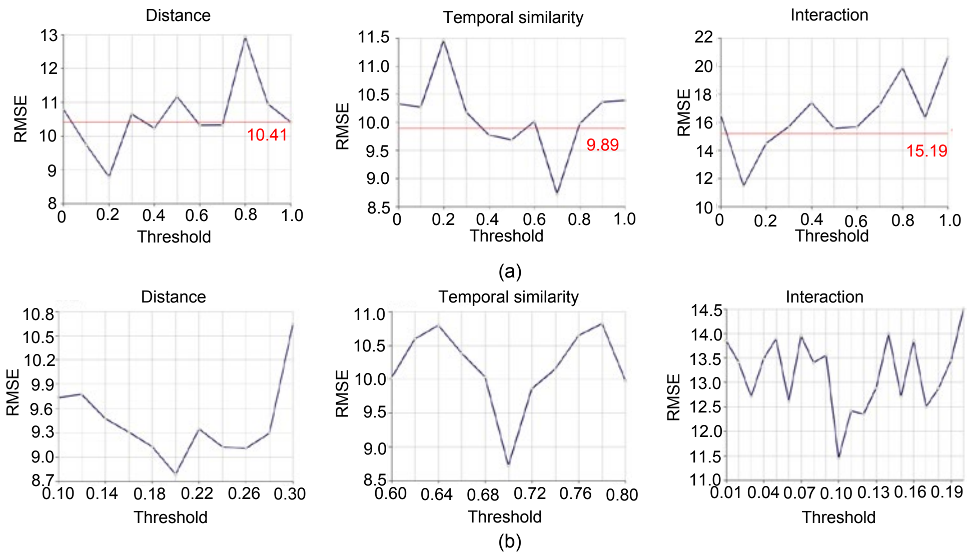


Fig. 4 Selection of different thresholds based on the DBS dataset: (a) rough search space; (b) optimized search space

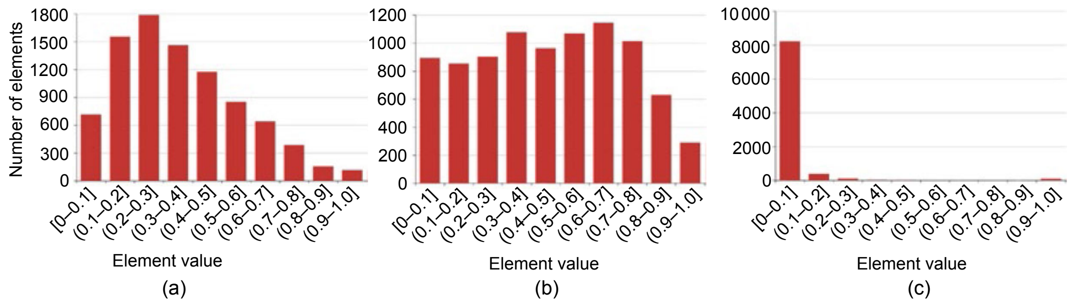


Fig. 5 Distribution of the element value in different graphs based on the taxi dataset: (a) distance; (b) temporal similarity; (c) interaction

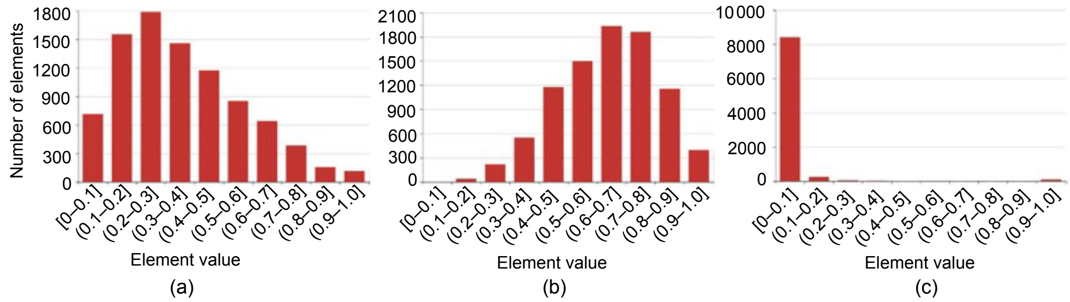


Fig. 6 Distribution of the element value in different graphs based on the PBS dataset: (a) distance; (b) temporal similarity; (c) interaction

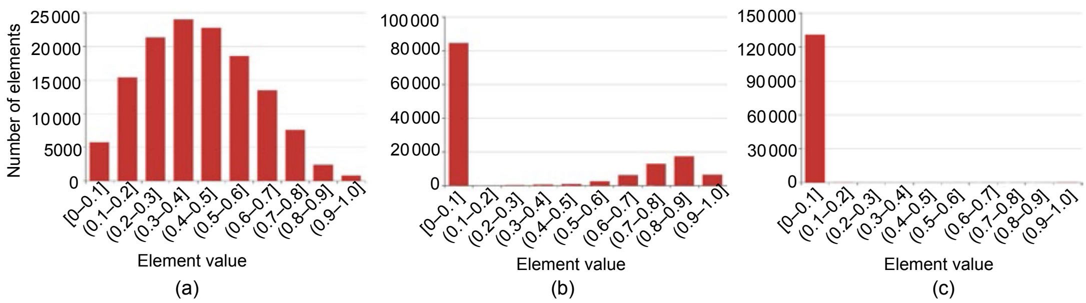


Fig. 7 Distribution of the element value in different graphs based on the DBS dataset: (a) distance; (b) temporal similarity; (c) interaction

value distributions in the distance graph are relatively dispersed. The values of $thres_d$ are relatively small, indicating that only adjacent regions are taken into consideration. For the temporal similarity graph, the element value distributions are dispersed based on the PBS and taxi datasets. The value of the interval [0, 0.1] is large for the DBS dataset. Because many regions have no bicycle flows, the temporal similarity values between these regions are 0. The values of $thres_{tSimi}$ are relatively large for all datasets, which means that only the regions with the most similar temporal usage patterns are considered. For the interaction graph, most values concentrate on the interval of [0, 0.1], which indicates that most regions have little interaction with others. The values of $thres_{inter}$ are equal to or larger than 0.1, indicating that only regions with stronger interactions are considered.

5.2.3 Model comparison

Table 2 demonstrates the results of MGCN-GRU and other baseline models based on these three datasets. The proposed MGCN-GRU model outperforms other models on both metrics on different datasets, proving the effectiveness of our model for traffic flow forecasting tasks. We summarize several findings from Table 2:

1. The traditional methods such as HA and ARIMA have relatively low prediction precision, which suggests that the non-neural network based methods are unsuitable for complex spatio-temporal prediction tasks.

2. The neural network based methods such as MGCN, LSTM, and GRU have better prediction precision than those of the HA and ARIMA models. For example, the RMSE of the GRU model shows a reduction of approximately 55.4%, 29.2%, and 32.8% compared with those of the HA model based on the PBS, taxi, and DBS datasets, respectively. The performance of LSTM is close to that of GRU using temporal dependency in the time series.

3. The proposed MGCN-GRU model performs the best among various baselines, which captures the graph-based spatial features and the temporal dependency simultaneously. For example, the RMSE shows a reduction of approximately 36.1%, 38.5%, and 40.3% compared with those of the MGCN model based on the PBS, taxi, and DBS datasets, respectively. The MGCN model has lower prediction accuracy, mainly because it models only the spatial correlations and ignores important temporal features. Compared with the GRU model, which captures only the temporal dependency, the RMSE of the MGCN-GRU model is decreased by approximately 7.35%, 12.1%, and 20.7% based on the PBS, taxi, and DBS datasets, respectively, indicating that MGCN can capture the hidden spatial correlation among regions to further improve the performance.

5.2.4 Effectiveness of multiple inter-region graph fusion

To verify the effectiveness of multiple inter-region graphs and additional attributes, we compare five models with different adjacency matrices and additional attributes. All models adopt GRU to capture the temporal correlation. GCN_dis, GCN_inter, and GCN_tSimi are the models using only one graph (distance, interaction, and temporal similarity graphs, respectively) and do not use additional attributes for prediction. MGCN_noAttr is the model fusing the three graphs without additional attributes, and MGCN-GRU is the proposed model.

Fig. 8 shows a comparison of model performances. Compared to the models using a single graph, models fusing the three graphs (MGCN_noAttr, MGCN-GRU) perform consistently better. When adding the additional attributes, the prediction

Table 2 Prediction accuracy comparison between different models

Model	RMSE			MAE		
	PBS dataset	Taxi dataset	DBS dataset	PBS dataset	Taxi dataset	DBS dataset
HA	7.7134	71.7902	19.9643	3.8163	34.4045	5.3395
ARIMA	5.2191	88.2771	23.7872	2.7086	45.3036	6.8241
MGCN	4.9874	72.6974	17.8231	2.6484	36.5174	6.2653
LSTM	3.4460	50.8887	13.5471	1.9809	28.7526	5.3938
GRU	3.4371	50.8379	13.4207	1.9629	28.3321	5.4848
MGCN-GRU	**3.1846**	**44.7117**	**10.6384**	**1.7818**	**24.1932**	**3.0705**

Best results are in bold

accuracy of the MGCN-GRU model is further improved. In addition, the model performance depends on the datasets and the selected graph using one graph, although the differences are not obvious. By adopting the proposed method, analyzers do not need to select which graph to use, since the multi-graph convolutional network can extract useful information from all inter-region graphs and improve the prediction accuracy.

5.2.5 Visualization of the prediction results

To further show the usability of our method, we visualize the predicted and real results through heat maps. We first compare the predicted outflows/inflows and real outflows/inflows based on the taxi dataset on different days. Each region is rendered with a color related to its flow number. A gradient color scheme (purple-red-yellow-green) is used to encode the flow magnitude. A purple region indicates a region with a large number of flows. We can see from Figs. 9 and 10 that the predicted values are very close to the real ones. The predicted popularity of regions has a similar trend compared to the ground truth. We can find the regions with large flow numbers from the heat maps. On a workday morning peak, region 1 has the largest numbers of outflows and inflows (Figs. 9a and 9c), since it contains a lot of office buildings and the busiest transportation hub (Grand Central Terminal) in NYC. On the night of Christmas day, many regions have obvious inflows or outflows. As seen from Fig. 10c, region 2 has the largest number of inflows. Because region 2 contains many nightlife places, it is reasonable to suppose that many people will go there to enjoy life at night.

We also visualize the flow differences through heat maps. A diverging color scheme (blue-yellow-red) is used to encode the difference in the inflow number and outflow number. If the inflow number is equal to the outflow number, the region is rendered yellow. If the inflow number is larger than the outflow

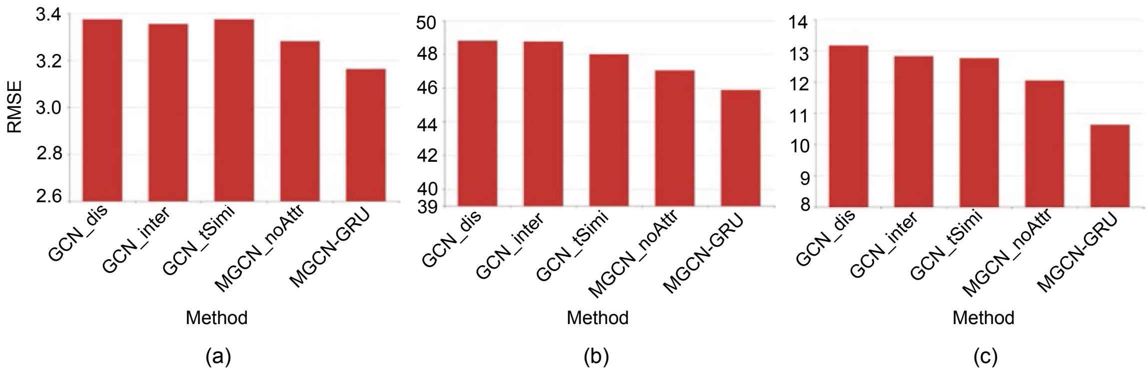


Fig. 8 Model performance comparison using different inter-region graphs and additional attributes: (a) PBS dataset; (b) taxi dataset; (c) DBS dataset

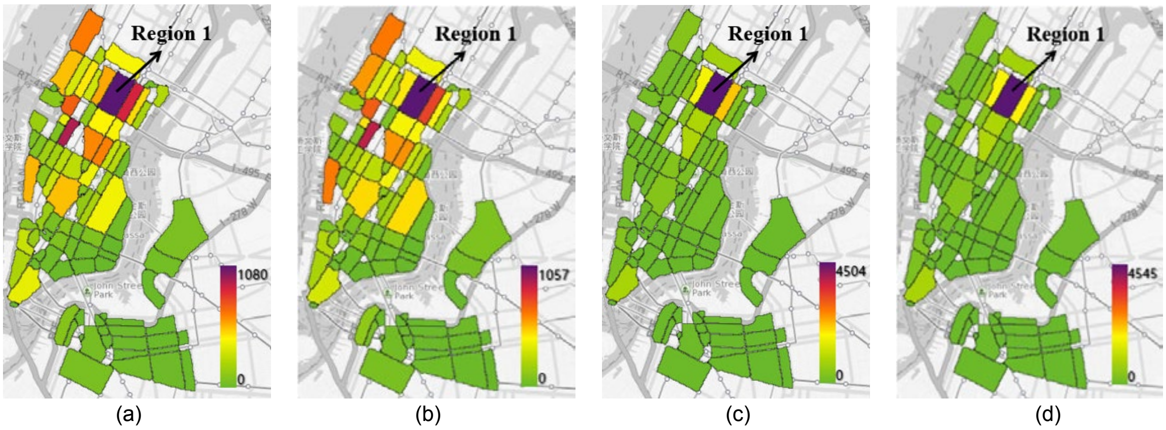


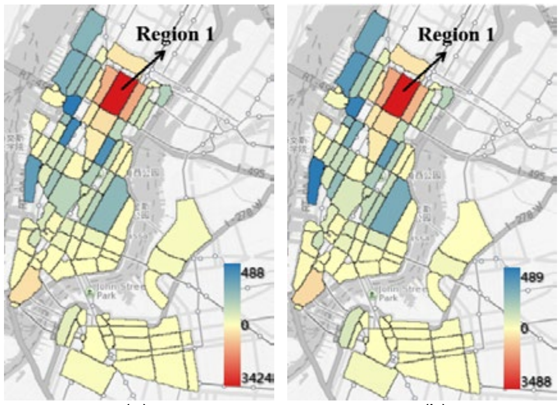
Fig. 9 Visualization of the prediction and real results based on the taxi dataset on 7:00 to 8:00, Apr. 23, 2014 (workday): (a) predicted outflows; (b) real outflows; (c) predicted inflows; (d) real inflows (References to color refer to the online version of this figure)

number, the region is rendered red. Otherwise, the region is rendered blue. Generally, the predicted flow differences (Figs. 11a and 12a) are similar to those of the real ones (Figs. 11b and 12b). The prediction for region 3 is not accurate (Figs. 12a and 12b). We can also know the region congestion conditions from Figs. 11 and 12. For example, a lot of people enter region 1 on workday morning (Fig. 11a). On the night of Christmas day, the nightlife area (region 2) has more inflows (Fig. 12a). Therefore, our method can help the traffic manager grasp the regional traffic conditions in advance.

6 Conclusions and future work

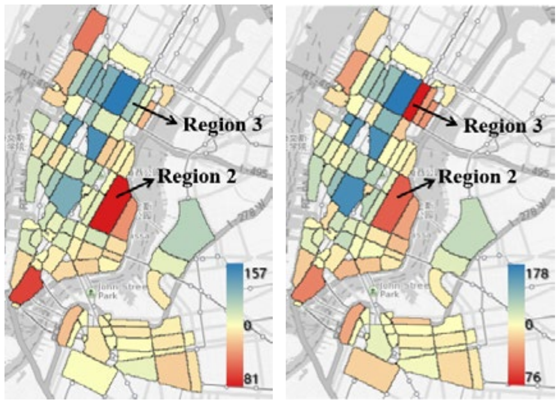
In this paper, we have proposed a novel deep learning based prediction model called MGCN-GRU, which combines the multi-graph convolutional network and GRU to forecast traffic flows in irregular regions. The multi-graph convolutional network has been proposed to model the heterogeneous spatial relationships among irregular regions. The GRU has been used to capture the dynamic temporal dependence of traffic flows and eventually fulfils the traffic flow prediction tasks. Compared with various previous methods, the MGCN-GRU model can successfully capture the spatio-temporal dependencies for irregular regions, and obtains the best prediction results on the three real-world traffic datasets.

In future, we aim to further improve the model performance. We plan to add the spatial- temporal attention mechanism in the network structure to better



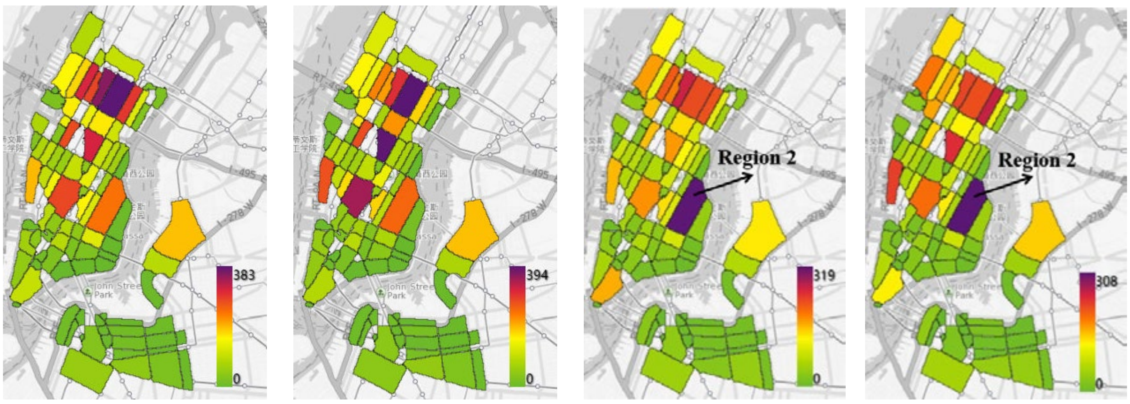
(a) (b)

Fig. 11 Visualization of flow differences based on the taxi dataset on 7:00 to 8:00, Apr. 23, 2014 (workday): (a) predicted flow differences; (b) real flow differences (References to color refer to the online version of this figure)



(a) (b)

Fig. 12 Visualization of flow differences based on the taxi dataset on 24:00 Dec. 25, 2014 to 1:00 Dec. 26, 2014: (a) predicted flow differences; (b) real flow differences (References to color refer to the online version of this figure)



(a) (b) (c) (d)

Fig. 10 Visualization of the prediction and real results based on the taxi dataset on 24:00 Dec. 25, 2014 to 1:00 Dec. 26, 2014: (a) predicted outflows; (b) real outflows; (c) predicted inflows; (d) real inflows (References to color refer to the online version of this figure)

capture the spatial dependency and temporal dynamics. In addition, it would be meaningful to consider more additional attributes, such as social events, in the prediction model.

Contributors

Dewen SENG designed the research. Fanshun LV and Ziyi LIANG processed the data and implemented the system. Xiaoying SHI drafted the manuscript. Dewen SENG, Xiaoying SHI, and Qiming FANG revised and finalized the paper.

Compliance with ethics guidelines

Dewen SENG, Fanshun LV, Ziyi LIANG, Xiaoying SHI, and Qiming FANG declare that they have no conflict of interest.

References

Box GEP, Jenkins GM, Reinsel GC, 2015. Time Series Analysis: Forecasting and Control. John Wiley & Sons, New York, USA.

Bruna J, Zaremba W, Szlam A, et al., 2014. Spectral networks and locally connected networks on graphs. Proc Int Conf on Learning Representations, p.1-14.

Chai D, Wang LY, Yang Q, 2018. Bike flow prediction with multi-graph convolutional networks. Proc 26th ACM SIGSPATIAL Int Conf on Advances in Geographic Information Systems, p.397-400. https://doi.org/10.1145/3274895.3274896

Chandra SR, Al-Deek H, 2009. Predictions of freeway traffic speeds and volumes using vector autoregressive models. *J Intell Transp Syst*, 13(2):53-72. https://doi.org/10.1080/15472450902858368

Defferrard M, Bresson X, Vandergheynst P, 2016. Convolutional neural networks on graphs with fast localized spectral filtering. Proc 30th Int Conf on Neural Information Processing Systems, p.3844-3852.

Fu R, Zhang Z, Li L, 2016. Using LSTM and GRU neural network methods for traffic flow prediction. Proc 31st Youth Academic Annual Conf of Chinese Association of Automation, p.324-328. https://doi.org/10.1109/yac.2016.7804912

Kaltenbrunner A, Meza R, Grivolla J, et al., 2010. Urban cycles and mobility patterns: exploring and predicting trends in a bicycle-based public transport system. *Perv Mob Comput*, 6(4):455-466. https://doi.org/10.1016/j.pmcj.2010.07.002

Kim Y, Wang P, Mihaylova L, 2019. Scalable learning with a structural recurrent neural network for short-term traffic prediction. *IEEE Sens J*, 19(23):11359-11366. https://doi.org/10.1109/jsen.2019.2933823

Kipf TN, Welling M, 2017. Semi-supervised classification with graph convolutional networks. Proc 5th Int Conf on Learning Representations, p.1-10.

Li YG, Yu R, Shahabi C, et al., 2018. Diffusion convolutional recurrent neural network: data-driven traffic forecasting. Proc 6th Int Conf on Learning Representations, p.1-10.

Monti F, Bronstein MM, Bresson X, 2017. Geometric matrix completion with recurrent multi-graph neural networks. Proc 31st Int Conf on Neural Information Processing Systems, p.3697-3707.

Moreira-Matias L, Gama J, Ferreira M, et al., 2013. Predicting taxi–passenger demand using streaming data. *IEEE Trans Intell Transp Syst*, 14(3):1393-1402. https://doi.org/10.1109/tits.2013.2262376

Seo Y, Defferrard M, Vandergheynst P, et al., 2018. Structured sequence modeling with graph convolutional recurrent networks. Proc 25th Int Conf on Neural Information, p.362-373. https://doi.org/10.1007/978-3-030-04167-0_33

Tian YX, Pan L, 2015. Predicting short-term traffic flow by long short-term memory recurrent neural network. IEEE Int Conf on Smart City/SocialCom/SustainCom, p.153-158. https://doi.org/10.1109/smartcity.2015.63

Wang P, Kim Y, Vaci L, et al., 2018. Short-term traffic prediction with vicinity Gaussian process in the presence of missing data. Sensor Data Fusion: Trends, Solutions, Applications, p.1-6. https://doi.org/10.1109/sdf.2018.8547118

Williams BM, Hoel LA, 2003. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results. *J Transp Eng*, 129(6):664-672.

Yao HX, Wu F, Ke JT, et al., 2018. Deep multi-view spatial-temporal network for taxi demand prediction. Proc 32nd AAAI Conf on Artificial Intelligence, p.2588-2595.

Ying R, He RN, Chen KF, et al., 2018. Graph convolutional neural networks for web-scale recommender systems. Proc 24th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining, p.974-983. https://doi.org/10.1145/3219819.3219890

Yoon JW, Pinelli F, Calabrese F, 2012. Cityride: a predictive bike sharing journey advisor. Proc 13th Int Conf on Mobile Data Management, p.306-311. https://doi.org/10.1109/mdm.2012.16

Yu B, Yin HT, Zhu ZX, 2018. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. Proc 27th Int Joint Conf on Artificial Intelligence, p.1-7. https://doi.org/10.24963/ijcai.2018/505

Yu R, Li YG, Shahabi C, et al., 2017. Deep learning: a generic approach for extreme condition traffic forecasting. Proc SIAM Int Conf on Data Mining. p.777-785. https://doi.org/10.1137/1.9781611974973.87

Yuan NJ, Zheng Y, Xie X, et al., 2015. Discovering urban functional zones using latent activity trajectories. *IEEE Trans Knowl Data Eng*, 27(3):712-725. https://doi.org/10.1109/tkde.2014.2345405

Zhang JB, Zheng Y, Qi DK, et al., 2016. DNN-based prediction model for spatio-temporal data. Proc 24th ACM SIGSPATIAL Int Conf on Advances in Geographic Information Systems, p.92. https://doi.org/10.1145/2996913.2997016

Zhang JB, Zheng Y, Qi DK, et al., 2018. Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artif Intell*, 259:147-166. https://doi.org/10.1016/j.artint.2018.03.002

Zhao L, Song YJ, Zhang C, et al., 2020. T-GCN: a temporal graph convolutional network for traffic prediction. *IEEE Trans Intell Transp Syst*, 21(9):3848-3858. https://doi.org/10.1109/tits.2019.2935152

Zhu L, Yu FR, Wang YG, et al., 2019. Big data analytics in intelligent transportation systems: a survey. *IEEE Trans Intell Transp Syst*, 20(1):383-398. https://doi.org/10.1109/tits.2018.2815678