Zhou Q, Gu JJ, Ling C *et al.* Exploiting multiple correlations among urban regions for crowd flow prediction. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 35(2): 338–352 Mar. 2020. DOI 10.1007/s11390-020-9970-y

Exploiting Multiple Correlations Among Urban Regions for Crowd Flow Prediction

Qiang Zhou, Jing-Jing Gu*, *Member*, *CCF*, Chao Ling, Wen-Bo Li, Yi Zhuang, and Jian Wang, *Senior Member*, *CCF*, *Member*, *ACM*, *IEEE*

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

E-mail: {zhouqnuaacs, gujingjing, lingchao, liwenbo, zhuangyi, wangjian}@nuaa.edu.cn

Received August 20, 2019; revised January 17, 2020.

Abstract Crowd flow prediction has become a strategically important task in urban computing, which is the prerequisite for traffic management, urban planning and public safety. However, due to variousness of crowd flows, multiple hidden correlations among urban regions affect the flows. Besides, crowd flows are also influenced by the distribution of Points-of-Interests (POIs), transitional functional zones, environmental climate, and different time slots of the dynamic urban environment. Thus, we exploit multiple correlations between urban regions by considering the mentioned factors comprehensively rather than the geographical distance and propose multi-graph convolution gated recurrent units (MGCGRU) for capturing these multiple spatial correlations. For adapting to the dynamic mobile data, we leverage multiple spatial correlations and the temporal dependency to build an urban flow prediction framework that uses only a little recent data as the input but can mine rich internal modes. Hence, the framework can mitigate the influence of the instability of data distributions in highly dynamic environments for prediction. The experimental results on two real-world datasets in Shanghai show that our model is superior to state-of-the-art methods for crowd flow prediction.

Keywords crowd flow prediction, multi-graph convolutional network, multiple correlations mining

1 Introduction

The accurate prediction of crowd flows is a fundamental basis for numerous applications in the field of urban computing, such as traffic management, urban planning, and public safety. With the rapid development of urbanization, 55% of the world's population lives in urban areas, a proportion that is expected to increase to 68% by 2050①. On the other hand, the ubiquitous GPS-embedded devices record the trajectories of these city dwellers. Such huge amounts of crowd flow data enable the new paradigm for the city management by providing tremendous sensing capabilities for understanding the city dynamics. For example, using big data with the City of Dublin, Irish Republic, IBM② has been forecasting traffic conditions to identify and solve the root causes of traffic congestion in the public bus network.

However, due to the variousness of crowd flows, multiple correlations which are obscure or hidden among urban regions affect urban flows in varying degrees. It is important to exploit multiple correlations between urban regions for fully mining the spatial information. Some previous researches mainly focused on predicting the crowd flows of gridded regions [1–3]. Although partitioning a city into grids is more quickly, the spatial dependencies of crowd flows in these rectangular regions are highly confusing and can hardly be modelled. Moreover, previous work on modeling spatial dependencies is concentrated on only neighborhood information [4, 5]

Regular Paper

Special Section on Learning and Mining in Dynamic Environments

This work was supported by the National Natural Science Foundation of China under Grant No. 61572253, and the Aviation Science Fund of China under Grant No. 2016ZC52030.

*Corresponding Author

①https://www.unwater.org/water-facts/urbanization/, Aug. 2019.

②http://analytics-magazine.org, Aug. 2019.

or some other superficial information of flows [6]. There are few researches on extracting multiple correlations from urban flows. Hence, the mining for spatial dependencies in urban flows is not enough. Fig.1 shows an example of a part of correlations among some regions. Although residential 1 is geographically isolated, we can find it may correlate to residential 2 that shares similar flow patterns. The two office zones share similar flow patterns too. Besides, the subway station is associated with different types of regions because of the high flow exchange. The office zone 1, which is adjacent to residential 2, also has high-flow exchange with the residential area.

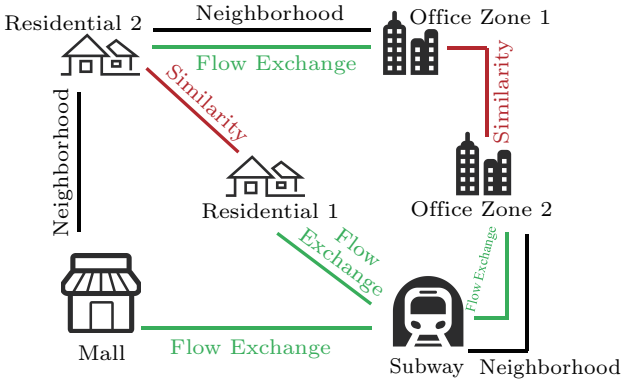


Fig.1. Example of a part of complex correlations among urban regions.

There also would be difficulties in predicting crowd flows in the highly dynamic environments of developing cities. Some existing crowd flow prediction methods [4, 7] not only use the data in recent time slots but even use the data of a year or two ago. However, because of rapid in-migration, unplanned urban expansions, and challenges in infrastructure financing, developing cities are always undergoing dramatical changes [8] so that the distribution of urban flows has already drifted over time. Besides, previous researches have mostly paid attention to the prediction for particular kinds of crowd flows. These specific methods usually take the features of flows into consideration, such as sharing bike system [9] and ride-hailing services [7], which could barely be transferred to other flow data.

In this work, we formulate the prediction of crowd flows as a spatiotemporal forecasting problem. A segmented region can be viewed as a node with some attributions [10] such as inflow and outflow. These attributions are associated with the historical observations of this node and the other related nodes [9]. To the best of our knowledge, previous researches mainly conduct their experiments for the region-level inflows and outflows forecasting [6, 11]. There are few experiments about crowd flow prediction between region pairs.

To address the challenges mentioned above, we propose a novel multi-graph deep learning framework for crowd flow prediction between urban regions of developing cities, called Multi-Graph Convolution Gated Recurrent Units (MGCGRU). In our work, we first study three types of pair-wise correlations among regions, including the geographical distance, the flow exchange, and the flow similarity, which are inferred from historical urban flows. Then, we take advantage of graph convolution to construct the multi-graph convolution operator, which can capture the multiple spatial dependencies adaptively. By replacing the matrix multiplications in gated recurrent units (GRU) with the multi-graph convolution operator, the MGCGRU model can capture both temporal and multiple spatial dependencies among the sequential data of crowd flows. This model is also convenient for multi-step ahead prediction. The main contributions of this paper are as follows.

• To exploit the interactions of crowd flows, we construct three different types of correlations among urban regions, including neighborhood information, flow exchange frequency, and flow similarity. This similarity can represent the correlations among urban regions relatively comprehensively. Then, we propose a novel prediction model called MGCGRU, which can capture these multiple spatial dependencies of urban crowd flows.

• For better adapting to the dynamic mobile data, we leverage the multiple spatial correlations and the temporal dependency to build a prediction framework that uses only a little recent data as the input but can mine rich internal modes. Hence, the framework can mitigate the influence of the instability of data distributions in highly dynamic environments, and enhance the memory bottleneck of predictor.

• We conduct an extensive experimental study on two real-world datasets in Shanghai, including the pair-wise flows of a dockless shared bike system. The results demonstrate the advantages of our MGCGRU model beyond the adaptations of several state-of-the-art approaches.

2 Related Work

2.1 Spatio-Temporal Prediction

As a fundamental issue of urban computing, crowd flow prediction is a long-standing problem for data-driven urban management [12]. There exist several

general methods like regression [11, 13] for flow prediction. Since crowd flows can be regarded as a kind of time series data, time series analysis methods such as ARIMA [14, 15] have been extensively studied for this task. Besides, some researchers aimed to predict travel speed [16] or traffic conditions on the road [17], which use the readouts of road sensors as training data. Divided by traffic types, the targets of spatio-temporal prediction include taxis [12], buses [18], sharing bikes [19, 20], and so on. However, because these researches often focus on a particular type of traffic, little work conducts their experimental studies on the different types of datasets.

2.2 Deep Learning for Crowd Flow Prediction

Deep learning has achieved numerous successes in many fields such as compute vision [21] and natural language understanding [22, 23]. In particular, recurrent neural networks (RNNs) such as gated recurrent units have been used successfully for sequence learning tasks like urban flow prediction [24], but they can only capture temporal dependencies in data. In order to resolve this problem, [25] has proposed a convolutional LSTM network for spatio-temporal prediction problems. However, this method can hardly capture non-Euclidean spatial dependencies. Although it is effective to leverage convolutional networks for flow prediction of grided regions, crowd flow prediction of irregular regions is actually of more realistic significance.

To handle the non-Euclidean structured data, researchers first introduced graph convolutional neural networks (GCN) in [26], which bridges the spectral graph theory and deep neural networks. After that, [27] proposes ChebNet which improves GCN with fast localized convolutions filters. GCN has been applied to semi-supervised classification [28] and image analysis [29]. Recently, DMVST-Net [12] has been proposed which uses graph embedding as an external feature for spatiotemporal prediction and consequently fails to use the demand values from related regions. The authors further improved this method by modeling the periodically shift problem with the attention mechanism [30]. Researchers of [6] proposed the multi-graph convolutional networks for bike flow prediction, but this method only uses the weighted sum of adjacency matrices as a spatial dependency and applies the traditional convolutional networks to the prediction problem. In [16], the authors modelled the traffic as a diffusion process and proposed DCRNN to combine the GCN with gated recurrent units that capture both spatial and temporal dependencies. However, these methods are all aimed at a particular kind of crowd flows, which could hardly be adapted to other flow prediction problems. Furthermore, MVGCN [4] was proposed, which can capture multi-hop temporal dependencies for crowd flow prediction. However, this method needs a long period of data, and it is hard to conduct it in highly dynamic urban environments. Moreover, researchers proposed ST-MGCN [7] to capture multiple spatial dependencies. It introduces the attention mechanism, which can better obtain long-term time dependencies in stable environments effectively. However, due to the great changes in data distribution over time in the highly dynamic environment, long-term time dependence would even make the performance worse and also increase the overhead. In addition, ST-MGCN has a more complicated calculation process due to recalculating once per prediction.

3 Preliminaries

3.1 Region Partition

To analyze the crowd flows of a city, we should first segment the city into small regions. As mentioned in Section 1, the regular gridded regions can hardly represent any particular urban functions. Furthermore, the morphological approach [31] can hardly identify the gathering places of a city. With the development of a city, the urban regions which have already become crowded places always attract crowd flows, even if their urban functions may change greatly. Inspired by these thoughts, we adopt the DBSCAN algorithm [32], a classic density-based clustering method, to cluster the starting points and the ending points of crowd flows to form urban regions.

As shown in Fig.2, we visualize the average hourly distribution of starting points and ending points in a sharing bike system in Shanghai during day time. The dots colored in red and blue denote high and low point densities respectively, the green denotes the medium point densities, and the transparent color denotes there is no crowd flow. From the figure, it can be seen that most areas of the city have bike flows, but there are also some areas like a part of universities where shared bikes are not allowed. Therefore, to achieve a meaningful region partition, the density-based clustering approach becomes a natural choice since it can easily avoid those forbidden areas and works well with the detection of clusters that have irregular shapes.

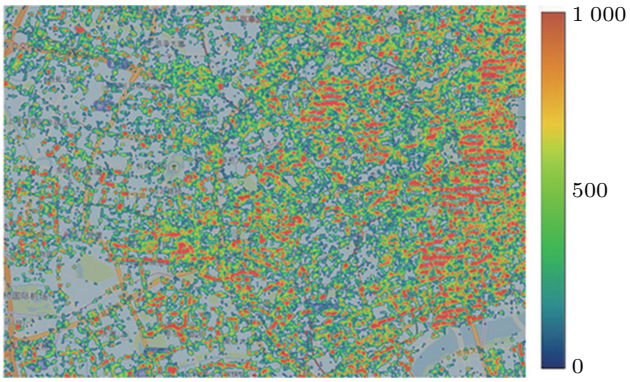
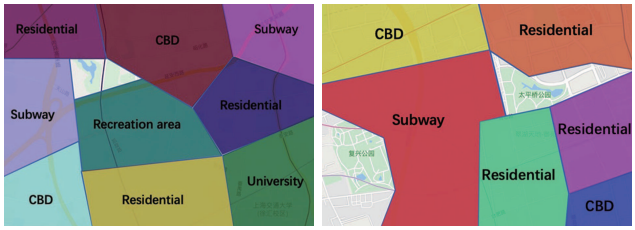


Fig. 2. Distribution of starting points and ending points in a sharing bike system in Shanghai.

We now show the details of region partition. We first put all the starting points and the ending points of crowd flows into a dataset. Then, we choose an arbitrary point in the dataset and find all its neighboring points within the range of a fixed distance (referred to as epsilon). For the chosen point, if the number of neighboring points is less than a threshold (referred to as minPts), it is temporally treated as an outlier and gets skipped. Otherwise, it forms clusters including all its neighboring points and itself. All of these neighboring points will perform the same expansion process until no points can be further included. We then repeat from choosing a new point and stop the process if all points are considered. After that, all points marked as outliers are removed and regions are formed by detecting the boundaries of clusters.

The examples of specific region partition are shown in Fig.3. By using proper parameters *epsilon* and *minPts*, each region is tagged as shown in the figure. We can observe that there are more uncoated parts in Fig.3(b) than in Fig.3(a), because the reachable area of taxis is smaller than that of bikes. Also, the area in the park is not pated into any region on both two datasets because there are few bikes or taxis.



(a) (b)

Fig.3. Examples of region partition. (a) Region partition result of MobikeSH. (b) Region partition result of TaxiSH.

3.2 Region-Level Urban Flow Prediction

With the partitioned regions, we are now ready to formalize the region-level urban flow prediction problem. The goal of urban flow prediction is to predict the future region-wise flows given previously observed ones. First of all, the whole city is represented as a weighted graph, denoted as $\mathcal{G} = (V, \varepsilon, \boldsymbol{A})$, where V and ε denote the set of nodes and edges, respectively, and $\boldsymbol{A} \in \mathbb{R}^{|V|\times|V|}$ is an adjacency matrix. Specifically, each partitioned region is defined as a node $v \in V$ and the relationships between regions are represented by edges ε. Each node v has P attributions called graph signals that represent the crowd flows related to the urban region, such as inflows, outflows, or just the flow to another region. Let $\boldsymbol{F}^{(t)} \in \mathbb{R}^{|V|\times P}$ represent the graph signals of all regions at the t-th interval. The crowd flow prediction problem is formulated as (1) which learns a function $f(\cdot)$ that maps h' historical graph signals to future h' graph signals, given a graph $\mathcal{G}$.

$$\{\boldsymbol{F}^{(t-h'+1)}, \boldsymbol{F}^{(t-h'+2)}, \ldots, \boldsymbol{F}^{(t)}; \mathcal{G}\} \xrightarrow{f(\cdot)} \{\boldsymbol{F}^{(t+1)}, \boldsymbol{F}^{(t+2)}, \ldots, \boldsymbol{F}^{(t+h)}\}. \quad (1)$$

Note that, the data ahead of $\boldsymbol{F}^{(t-h'+1)}$ is no longer used as inputs for prediction because we suppose the city as a highly dynamic environment and the recent data is much more valuable. Using only recent time for prediction, we can mitigate the influence of the instability of data distributions in such environments.

3.3 Framework

As aforementioned, we formalize the crowd flow prediction as a spatiotemporal prediction problem. We design a deep learning framework for solving it. As shown in Fig.4, the framework overview is composed of three major parts: data preparation, model construction, and learning and prediction.

In the data preparation stage, we first construct graph signals in different time slots $\boldsymbol{F}^{(t-h\prime)} \sim \boldsymbol{F}^{(t)}$ using the historical data of target urban flows. Considering the factors that may affect the flow rate, we need to fetch weather elements (e.g., weather, temperature, wind speed, visibility) and global date information (e.g., the time of the day, the day of the week). Then, we extract features from these extra factors, $E^{(t-h\prime)} \sim E^{(t)}$. Finally, we concatenate E with the graph signal attributes of $\boldsymbol{F}$ to form the input $\boldsymbol{F}^{(t-h\prime)\prime} \sim \boldsymbol{F}^{(t)\prime}$.

In the model construction stage, we exploit multiple correlations among urban regions from the existing crowd flow data to construct three related graphs,

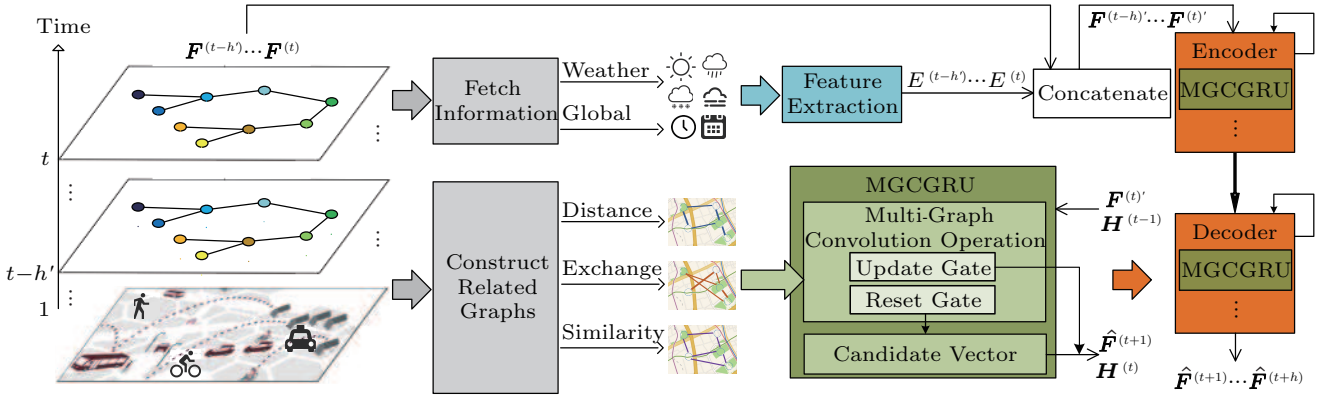


Fig.4. Framework overview of the prediction model based on Multi-Graph Convolution Gated Recurrent Units.

including region distance graph for representing the regional proximity, flow exchange graph for demonstrating the volume of regional crowd flows, and flow similarity graph for showing the similarity of the regional flow pattern. Based on the three graphs mentioned above, we propose a multi-graph convolution operator. Sequentially, we present a modified-GRU by replacing the matrix multiplications in gated recurrent units (GRU) with the multi-graph convolution operator, called MGCGRU, which can capture both temporal and multiple spatial dependencies among the sequential data of crowd flows.

In the learning and prediction stage, we utilize the Encoder-Decoder architecture to implement the MGCGRU model for multi-step ahead prediction. Both the encoder and the decoder are recurrent neural networks with MGCGRU. Given the prepared data $\boldsymbol{F}^{(t-h\prime)\prime} \sim \boldsymbol{F}^{(t)\prime}$, the encoder generates a final state through its network structure and transfers it to the decoder, which will then produce the ultimate prediction result.

4 Methodology

4.1 Encoding Region Correlations Using Related Graphs

Graph construction is the fundamental aspect of modeling the spatial dependency of urban regions. If the edges of related graphs cannot represent the correlations among urban regions accurately, it will result in the wrong learning direction of the model and generate a bad result of crowd flow prediction. Our study shows that the closer urban regions have stronger relations with each other. Besides, the regions with frequent exchanges or similar flow patterns seem to relate to each other. Based on these considerations, we encode three types of correlations among urban regions with graphs.

• *Region Distance Graph.* In urban areas, two regions near each other are affected by the same events, which means the fluctuation of urban flows will happen in nearby regions at the same time. Therefore, we construct the first related graph $\mathcal{G}^d = (V, \varepsilon^d, \boldsymbol{A}^d)$, which encodes the spatial proximity. We choose the geographic center as the representative of every irregular region, and then calculate the distance dis_{ij} between two arbitrary regions i and j. In particular, we use a base-e negative exponential function to model the geographic correlations among urban regions as (2). For controlling the sparsity of $\boldsymbol{A}^d$, there is a threshold of distance Th_d. The parameter b_d is adjusted for an appropriate value of $\boldsymbol{A}^d$.

$$A^d_{ij} = \begin{cases} e^{-\frac{dis^2_{ij}}{b_d}}, & \text{if } dis_{ij} \leqslant Th_d, \\ 0, & \text{if } dis_{ij} > Th_d. \end{cases} \tag{2}$$

• *Flow Exchange Graph.* The historical records of crowd flows can provide us the amount of exchange between urban regions. By common sense, those regions which are distant but with a tremendous amount of exchange can be related to each other. When the crowd flow patterns of one region change, it will appear that the crowd flows of another region tend to change too. Here, we define the flow exchange graph $\mathcal{G}^e = (V, \varepsilon^e, \boldsymbol{A}^e)$, which encodes how frequently two regions interact with each other. Specifically, we use the scaled sum of crowd flows between two regions to model this type of urban correlations as (3). $\mathcal{N}()$ is the normalization function. There is also a threshold Th_e

for controlling the sparsity of $\boldsymbol{A}^e$.

$$A^e_{ij} = \begin{cases} \mathcal{N}(F_{ij} + F_{ji}), & \text{if } \mathcal{N}(F_{ij} + F_{ji}) > Th_e, \\ 0, & \text{if } \mathcal{N}(F_{ij} + F_{ji}) \leqslant Th_e. \end{cases} \quad (3)$$

• *Flow Similarity Graph.* When we study the crowd flow patterns of a region, we can refer to other regions that have similar urban functions as this one. Hence, the regions with similar urban functions also share similar directed crowd flow patterns. Intuitively, we can approximate the similarity of urban regions by the correlation coefficients of crowd flows in each time slot. The flow similarity graph $\mathcal{G}^s = (V, \varepsilon^s, \boldsymbol{A}^s)$ is encoded as (4):

$$\begin{aligned} A^s_{ij} &= \begin{cases} (R_{ij})^{b_s}, & \text{if } (R_{ij})^{b_s} > Th_s, \\ 0, & \text{if } (R_{ij})^{b_s} \leqslant Th_s, \end{cases} \\ R_{ij} &= \frac{C_{ij}}{\sqrt{C_{ii} \times C_{jj}}}, \end{aligned} \quad (4)$$

where the correlation coefficients are calculated according to the covariance matrix $\boldsymbol{C}$. Also, there is the threshold Th_s and a parameter b_s for adjusting the result. The covariance matrix $\boldsymbol{C}$ is calculated as (5).

$$C_{ij} = \mathrm{E}[(F^i - \mathrm{E}(F^i))(F^j - \mathrm{E}(F^j))], \quad (5)$$

where E() means the expected value.

4.2 Modeling Spatio-Temporal Dependencies

Given the region correlations in Subsection 4.1, traditional methods such as convolutional neural networks (CNNs) are helpless for modeling such spatial dependencies encoded with graphs. Recently, the convolutional neural networks for graphs have been proposed [26]. In this work, we leverage the spectral graph convolution to build the multi-graph convolution operator for modeling the spatial dependencies, which can mine rich internal modes of urban flows. For convenience, we denote our multi-graph convolution operator as $\star_{\mathbb{A}}$ like the spectral formulation in [27], where $\mathbb{A} = \{\boldsymbol{A}^d, \boldsymbol{A}^e, \boldsymbol{A}^s\}$ represents the set of three constructed graphs. The normalized graph Laplacian $\boldsymbol{L}$ of graph $\mathcal{G}$ is utilized which is defined as (6).

$$\boldsymbol{L}_{\boldsymbol{A}} = \boldsymbol{D}_{\boldsymbol{A}}^{-\frac{1}{2}}(\boldsymbol{D}_{\boldsymbol{A}} - \boldsymbol{A})\boldsymbol{D}_{\boldsymbol{A}}^{-\frac{1}{2}} = \Phi_{\boldsymbol{A}}\Lambda_{\boldsymbol{A}}\Phi_{\boldsymbol{A}}^{\mathrm{T}}, \quad (6)$$

where $\boldsymbol{D}_{\boldsymbol{A}} \in \mathbb{R}^{|V|\times|V|}$ is the degree matrix of the adjacency matrix $\boldsymbol{A}$. $\Phi_{\boldsymbol{A}} \in \mathbb{R}^{|V|\times|V|}$ and $\Lambda_{\boldsymbol{A}} \in \mathbb{R}^{|V|\times|V|}$ are the matrix of eigenvectors and the diagonal matrix of eigenvalues with eigenvalue decomposition $\boldsymbol{L}$ respectively. By the definition above, the spectral graph convolution $\star_{\boldsymbol{A}}$ with the graph signal $\boldsymbol{F} \in \mathbb{R}^{|V|\times P}$ and a filter $\theta \in \mathbb{R}^K$ is defined as (7) in the Fourier domain.

$$\theta \star_{\boldsymbol{A}} \boldsymbol{F}_{:,p} = \Phi \cdot \sum_{k=0}^{K-1} (\theta_k \cdot \Lambda^k) \cdot \Phi^{\mathrm{T}} \cdot \boldsymbol{F}_{:,p}. \quad (7)$$

Based on (7), for modeling the multiple spatial dependencies described in Subsection 4.1 at the same time for crowd flow prediction, we present the multi-graph convolution operator $\star_{\mathbb{A}}$ with the graph signal $\boldsymbol{F} \in \mathbb{R}^{|V|\times P}$ and a filter $\theta \in \mathbb{R}^{|\mathbb{A}|\times K}$ as (8).

$$\theta \star_{\mathbb{A}} \boldsymbol{F}_{:,p} = \sum_{\boldsymbol{A}\in\mathbb{A}} (\Phi_{\boldsymbol{A}} \cdot \sum_{k=0}^{K-1} (\theta_{\boldsymbol{A},k} \cdot \Lambda_{\boldsymbol{A}}^k) \cdot \Phi_{\boldsymbol{A}}^{\mathrm{T}}) \cdot \boldsymbol{F}_{:,p}. \quad (8)$$

However, the multiplication with $\Phi_{\boldsymbol{A}}$ is computationally expensive when $|V|$ is large. We leverage the ChebNet [27] to overcome this problem as shown in (9).

$$\begin{aligned} \theta \star_{\mathbb{A}} \boldsymbol{F}_{:,p} &= \sum_{\boldsymbol{A}\in\mathbb{A}} \sum_{k=0}^{K-1} (\theta_{\boldsymbol{A},k} \cdot \boldsymbol{L}_{\boldsymbol{A}}^k) \cdot \boldsymbol{F}_{:,p} \\ &= \sum_{\boldsymbol{A}\in\mathbb{A}} \sum_{k=0}^{K-1} (\widetilde{\theta}_{\boldsymbol{A},k} \cdot T_k(\widetilde{\boldsymbol{L}}_{\boldsymbol{A}})) \cdot \boldsymbol{F}_{:,p}, \end{aligned} \quad (9)$$

where $T_0(x) = 1$, $T_1(x) = x$, $T_k(x) = xT_{k-1}(x) - T_{k-2}(x)$ are the basis of the Cheyshev polynomial, and $\widetilde{\boldsymbol{L}}_{\boldsymbol{A}} = \frac{2}{\lambda_{\max}}\boldsymbol{L}_{\boldsymbol{A}} - \boldsymbol{I}$ represents the scaled Laplacian, $\lambda_{\max}$ denotes the largest eigenvalue of $\boldsymbol{L}_{\boldsymbol{A}}$. Therefore, the filter θ is now replaced with the Chebyshev coefficients $\widetilde{\theta} \in \mathbb{R}^{|\mathbb{A}|\times K}$.

To model the temporal dependency of urban crowd flows, we use the recurrent neural networks (RNNs) as the basis of our prediction method. Specifically, inspired by [16], we replace the matrix multiplications in GRU [33] with our multi-graph convolution operator $\star_{\mathbb{A}}$ as (10) called MGCGRU.

$$\begin{aligned} r^{(t)} &= \sigma(\theta_{(r)} \star_{\mathbb{A}} [\boldsymbol{F}^{(t)}, \boldsymbol{H}^{(t-1)}] + b_r), \\ u^{(t)} &= \sigma(\theta_{(u)} \star_{\mathbb{A}} [\boldsymbol{F}^{(t)}, \boldsymbol{H}^{(t-1)}] + b_u), \\ \widetilde{\boldsymbol{H}}^{(t)} &= \tanh(\theta_{(\widetilde{H})} \star_{\mathbb{A}} [\boldsymbol{F}^{(t)}, (r^{(t)} \odot \boldsymbol{H}^{(t-1)})] + b_{\widetilde{\boldsymbol{H}}}), \\ \boldsymbol{H}^{(t)} &= u^{(t)} \odot \boldsymbol{H}^{(t-1)} + (1 - u^{(t)}) \odot \widetilde{\boldsymbol{H}}^{(t)}, \end{aligned} \quad (10)$$

where $r^{(t)}, u^{(t)}$, and $\widetilde{\boldsymbol{H}}^{(t)}$ denote the reset gate, the update gate, and the candidate matrix of time slot t respectively. On the right side of the equal signs, $\boldsymbol{F}^{(t)}$ and $\boldsymbol{H}^{(t)}$ represent the input and the output of time slot t respectively. $\theta_{(r)}$, $\theta_{(u)}$, and $\theta_{(\widetilde{H})}$ are the parameters for the corresponding filters. Note that the parameters are replaced with the Chebyshev coefficients when training and can be trained using backpropagation, which will be faster than the original GCNs.

The related graphs are honestly large, but the adjacency matrices of related graphs are sparse. In the implementation, we load and calculate the graph matrices as sparse tensors using *tensorflow* package[③], which has lower space complexity than the traditional methods of storage. Therefore, we can load the large graphs of urban regions and apply our algorithm on computing platforms. In addition, it is helpful for breaking the memory bottleneck, as we use only several recent time flow matrices as inputs.

4.3 Training and Prediction

Because the crowd flow prediction problem is defined as a multi-step prediction problem, we apply the Encoder-Decoder architecture to our prediction model. As an example in Fig.5, there is a multi-layer MGCGRU model for h-step ahead crowd flow prediction. The structures of encoder and decoder are similar to traditional RNNs, which are constructed by multiple layers that consist of MGCGRU cells. We fetch the extra information to form $E^{(t-h)} \sim E^{(t+h)}$, and concatenate it with the historical crowd flow data $\boldsymbol{F}^{(t-h)} \sim \boldsymbol{F}^{(t+h)}$ for constructing the inputs data $\boldsymbol{F}'$, which is illustrated in Fig.4. The extra information includes weather conditions (temperature, humidity, wind speed, weather, visibility) and date information (the time of the day, the day of the week, and so on). Every MGCGRU cell produces a result as the input of its next layer, as well as a hidden state as the input of its next MGCGRU cell in the same layer. The encoder network compresses the information of historical graph signals into a hidden state vector. Then, the decoder network decodes the hidden state vector to perform urban flow predictions.

For training, the decoder takes the ground truths as inputs to improve the training effect. With the generated predictions of decoder, the entire network is trained by minimizing the loss function which represents the gap between predictions $[\hat{\boldsymbol{F}}^{(t+1)}, \hat{\boldsymbol{F}}^{(t+2)} \sim \hat{\boldsymbol{F}}^{(t+h)}]$ and ground truths $[\boldsymbol{F}^{(t+1)}, \boldsymbol{F}^{(t+2)} \sim \boldsymbol{F}^{(t+h)}]$ using backpropagation through time. For different types of crowd flows, the parameters of multiple correlations can be adjusted through training adaptively. Therefore, we can use MGCGRU on generic crowd flow data for prediction.

For prediction, the encoder is the same as that of training. In the decoder, we use the weather forecasting as extra weather information data. The ground truth data is also replaced with predictions $[\hat{\boldsymbol{F}}^{(t+1)}, \hat{\boldsymbol{F}}^{(t+2)} \sim \hat{\boldsymbol{F}}^{(t+h-1)}]$ generated by the model itself, as the earth-yellow dotted arrows in Fig.5.

5 Experiments

In this section, we present an extensive experimental study of our MGCGRU model, compared with six competitive algorithms. We conduct two sets of experiments on each of two different datasets to evaluate the effectiveness of the proposed MGCGRU model and study the parameter sensitivity. We also study the effects of spatial dependence modeling by using different related graphs.

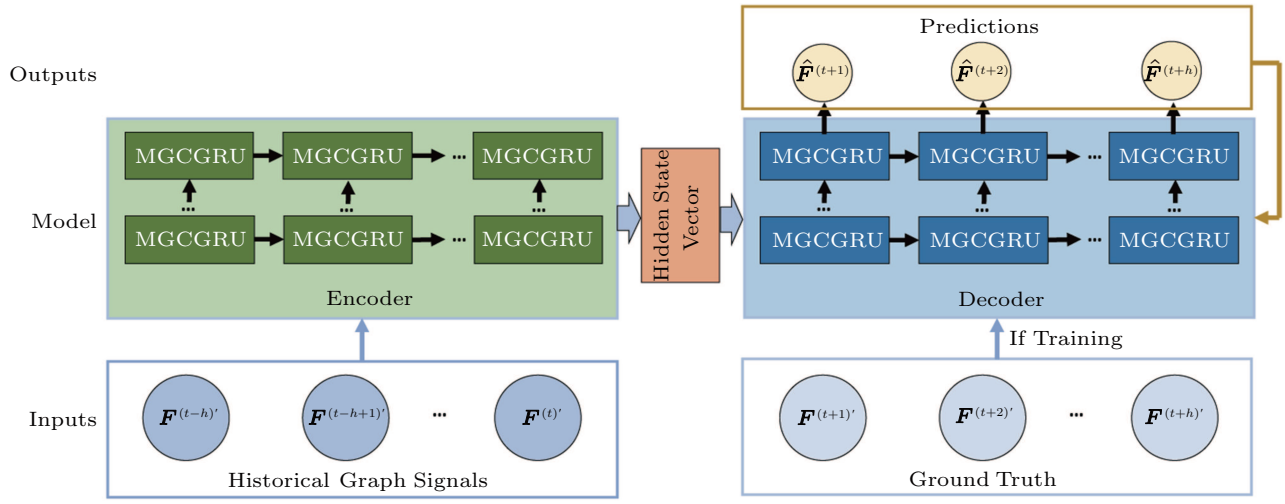


Fig.5. Encoder-decoder architecture for the MGCGRU model.

③https://www.tensorflow.org, Nov. 2019.

5.1 Setting

Datasets. We use two different datasets of crowd flows in Shanghai. The details are described as follows.

MobikeSH. The real-life Mobike data contains 957 357 367 riding records generated by 314 703 shared bikes from February 2017 to March 2018 in Shanghai. Each record specifies a bike ID, a pick-up location, a pick-up time, a drop-off location, and a drop-off time. We partition the main areas in Shanghai into 766 irregular regions based on the DBSCAN algorithm in Subsection 3.1 and build the related graphs using the methods in Subsection 4.1.

TaxiSH. The trajectory data is taxi GPS data in Shanghai from Apr. 1st, 2015 to Apr. 30th, 2015. There are 10 053 taxis in total. Each taxi generates a record every ten seconds. A record includes: taxi_ID, taxi condition, time, speed, direction, and location. The area partition and graph construction methods are the same with that of MobikeSH.

Baselines. We compare the proposed MGCGRU model with the following six algorithms for crowd flow forecasting.

• *Historical Average* (*HA*). HA directly uses the average crowd flow of previous time slots as the prediction. Specifically, the prediction of MobikeSH is based on aggregated data from the same time in previous weeks, while the prediction of TaxiSH is based on the data from the recent time due to the short time range.

• *Vector Auto-Regressive* (*VAR*) [14]. VAR is the multi-variate extension of the auto-regressive model, which can model the correlation among regions. We implement the model using the *statmodel* python package. The number of lages used is 5.

• *Gradient Boosting Regression Trees* (*GB-RT*) [34]. GB-RT is implemented using the *sklearn* python package. The optimal parameters are achieved by the grid search.

• *Feed Forward Neural Network* (*FNN*). We build a 3-layer feed forward neural network with L2 regularization. The inputs are the previous t timesteps and the output is the next timestep.

• *Long Short-Term Memory* (*FC-LSTM*) [35]. FC-LSTM is a Recurrent Neural Network with fully connected LSTM hidden units. There are three layers with some LSTM units. The inputs and output are the same as those of FNN.

• *Diffusion Convolutional Recurrent Neural Network* (*DCRNN*) [16]. DCRNN is a graph-convolution-based model for traffic forecasting. It uses the road network for building a non-euclidean region-wise relationship and models the spatiotemporal dependency by integrating graph convolution into the GRU.

• *Spatiotemporal Multi-Graph Convolution Network* (*ST-MGCN*) [7]. ST-MGCN is a deep learning model for ride-hailing demand forecasting. It can capture non-Euclidean correlations among regions using multi-graph convolution in the spatial modeling procedure. Furthermore, it augments the recurrent neural network with a contextual gating mechanism to incorporate global contextual information in the temporal modeling procedure.

Implementation. In the experiment, we aggregate crowd flows data into one-hour windows for MobikeSH and 30-minute windows for TaxiSH. Then, we randomly select 10% of the data as the validation data and 20% as the test data. The rest are used for training.

For all sequence-forecasting methods except ST-MGCN, the length of inputs data h' is set to 6. The length of inputs data of ST-MGCN h'' is set to 20. The extra information is added into graph signals as the inputs of every baseline except HA. HA directly uses weather information and date information in prediction. Only the time segments that share the same weather and date information with the target time segment are used for prediction. The Min-Max normalization method is used to scale the data (including the extra information) into the range $[-1, 1]$. In the evaluation, we re-scale the predicted value back to the normal values and compare them with ground truth data.

The neural network based models are all implemented using TensorFlow and trained via backpropagation and Adam optimization. For FNN, FC-LSTM, DCRNN, and ST-MGCN, the number of hidden units is set to 64 for each embed layer by default. Other hyperparameters are chosen following the default setting.

Metrics. We evaluate the performance by two metrics, namely mean absolute error (MAE) and root mean square error (RMSE), which are defined as follows:

$$MAE = \frac{1}{|T|}\sum_{t\in T}\frac{\sum_{i,j=1}^{M}|\widehat{\boldsymbol{F}}_{i,j}^{(t)} - \boldsymbol{F}_{i,j}^{(t)}|}{M^2},$$

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{t\in T}\frac{\sum_{i,j=1}^{M}(\widehat{\boldsymbol{F}}_{i,j}^{(t)} - \boldsymbol{F}_{i,j}^{(t)})^2}{M^2}},$$

where T denotes the set of forecasting time fragments and $\widehat{\boldsymbol{F}}_{i,j}^{(t)}$ and $\boldsymbol{F}_{i,j}^{(t)}$ denote the predicted and the ground truth crowd flows, respectively. Both of the two metrics are widely used in the regression tasks.

5.2 Performance Comparison

Table 1 shows the comparison of different approaches for two types of graph signals forecasting on both datasets. The lowest errors of each method in different tests are indicated in bold. We observe that MGCGRU performs the best in all our tests. Compared with the second-best method ST-MGCN, MGCGRU decreases the errors by (0.83%, 1.01%, 0.44%, 5.16%) under MAE and (1.64%, 1.94%, 0.39%, 8.21%) under RMSE on four different tests. Compared with the DCRNN model, MGCGRU decreases the errors by (2.06%, 3.16%, 3.44%, 6.85%) under MAE and (3.95%, 5.80%, 2.19%, 12.64%) under RMSE. We can find that MGCGRU has greater improvement in the tests of "in-out" graph signals. The values in these two datasets are much higher than those in the other two. Because the relevancy between the entire in-out flows and related regions is larger than that between the components of flows and related regions, MGCGRU performs better for these high crowd flow prediction problems. Among four tests, ST-MGCN, DCRNN, and MGCGRU achieve better performance than almost all other metrics, which suggests the importance of spatiotemporal correlations modeling. ST-MGCN and our MGCGRU model outperform DCRNN, which emphasizes the effectiveness of using multiple graphs. ST-MGCN performs a little worse than MGCGRU. As the data distribution changes over time greatly, the attention mechanism in ST-MGCN may not apply to the highly dynamic environments well. RNN-based methods, especially FC-LSTM, perform better than the other baselines in most cases, indicating that temporal dependency modeling has a beneficial effect. By contrast, FNN does not consider temporal dependency, resulting in that its performance is unstable and just passable. GBRT, as a tree-ensemble model, performs well on the MobikeSH dataset, even better than FC-LSTM. However, GBRT could hardly deal with the high sparsity of TaxiSH, therefore the performance of this method is unsatisfying with this dataset.

In general, the difference between the prediction results of the two datasets is not much. As mentioned previously, the MobikeSH dataset has a longer time duration range than TaxiSH. Thus the results of MobikeSH are more credible when we measure the performance of our methods. Therefore, for clarity, we use MobikeSH as the default dataset for the following experiments.

5.3 Results on Multi-Step Prediction

To further evaluate the performance of the proposed method, we show the multi-step prediction results based on MAE and RMSE over the MobikeSH dataset in Fig.6. The multi-step prediction models, including FC-LSTM, DCRNN, and MGCGRU, are all implemented with the Encoder-Decoder architecture. We set the length of inputs data h' to 6, and predict the next six timesteps after the inputs, to train the models. Besides, we also evaluate the ST-MGCN model. The length of ST-MGCN inputs data h'' is set to 20. To forecast a new flow matrix using ST-MGCN for multi-step ahead prediction, we concatenate the previous prediction result with the original inputs to reconstruct new inputs and recalculate once per prediction.

As a result, our proposed MGCGRU model outperforms the other three methods as the step number (timestep) varies from 1 to 6. FC-LSTM performs the worst because it only uses the flow data in recent times and ignores the correlations among urban regions. Thus, much useful information is wasted. With the increase of step number, the MAE and the RMSE of FC-LSTM are not always going up. That means FC-LSTM can hardly model the distribution of flow data when the step number is big enough. Moreover, we can observe that the MAE and the RMSE of DCRNN are very close

Table 1. Comparisons with Baselines on 2 Datasets Based on 2 Metrics: RMSE and MAE

Method	MobikeSH				TaxiSH			
	Top20		In-Out		Top20		In-Out	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
HA	0.543 7	0.937 7	7.440 3	11.387 5	0.084 4	1.395 4	1.611 3	11.545 0
VAR	0.367 3	0.915 6	4.436 8	8.600 9	0.063 7	1.057 1	1.119 1	5.420 5
GBRT	0.326 4	0.835 7	4.467 6	7.119 1	0.060 1	0.927 3	1.534 2	3.763 6
FNN	0.361 0	0.900 3	6.111 3	9.211 7	0.049 5	1.062 6	1.138 7	7.835 3
FC-LSTM	0.326 3	0.865 1	6.109 2	9.219 1	0.048 0	1.049 3	1.107 5	7.057 6
DCRNN	0.315 2	0.838 6	3.648 8	6.044 4	0.046 5	0.825 9	1.084 4	1.919 0
ST-MGCN	0.311 3	0.818 9	3.569 3	5.806 5	0.045 1	0.811 0	1.065 1	1.826 4
MGCGRU	**0.308 7**	**0.805 5**	**3.533 4**	**5.693 6**	**0.044 9**	**0.807 8**	**1.010 1**	**1.676 5**

Note: For the results, the smaller the better.

to these of MGCGRU when the timestep is 1. However, DCRNN is not robust along with the increase of step number, because it also ignores the influential multiple spatial dependencies, meaning that DCRNN does not work well for long-range crowd flow prediction. Furthermore, the performance of ST-MGCN is only second to that of MGCGRU, as the highly dynamic environment may influence the effect of attention mechanism.

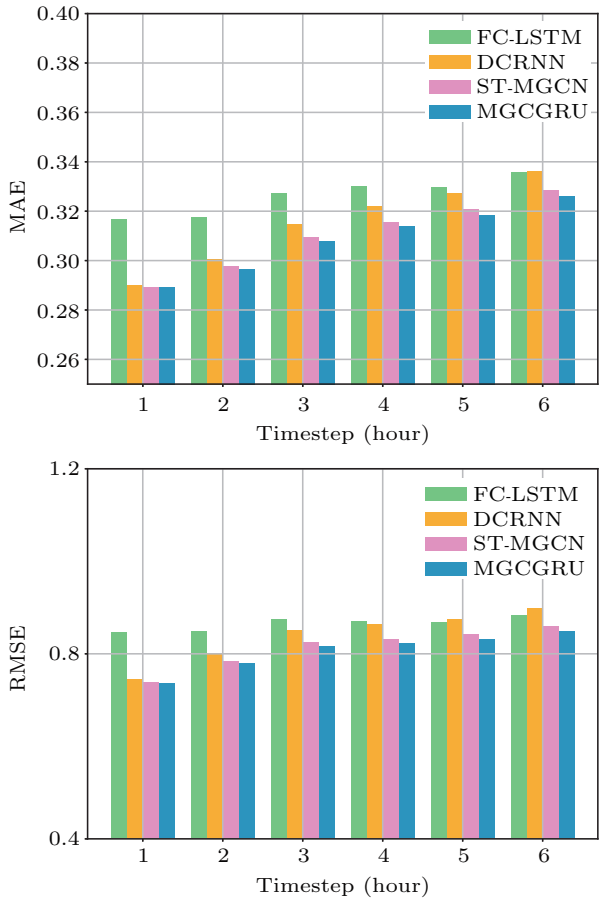


Fig.6. Step-wise comparisons on the Mobike test set.

5.4 Effects of Different Related Graphs

To investigate the effect of multiple related graphs, we only use a single graph for predicting the next six timesteps after the inputs on the MobikeSH dataset. These graphs include 1) the region distance graph, 2) the flow exchange factor graph, and 3) the flow similarity factor graph. The result is shown in Table 2. The lowest errors of each setting for different timesteps are indicated in bold. Overall, the results of single-graph models are similar to each other. Our MGCGRU model outperforms these single-graph models consistently, indicating the effectiveness of each type of region relationship. These spatial dependencies help the proposed method make a more accurate prediction because they all bring valuable prior knowledge into the model. More specifically, when the step number is 1, the flow similarity factor seems to be able to present a better result. One possible reason is that the regions sharing similar flow patterns provide more knowledge than the other two kinds of correlations when we predict the flows of the most recent time. With the step number increasing, the results of using the region distance graph are better than the other two, which means the neighborhoods are more important for long-range crowd flow prediction.

5.5 Result Insights of Representative Regions

To better understand how the model performs for different regions, Fig.7 illustrates the prediction results and the ground truth of two completely different regions from 9:00 a.m. of December 9(th) to 10:00 a.m. of December 11(th). Overall, the value of ground truths changes over time following a certain rule but varies sharply in a short range of time. A good prediction should fit a certain rule but get rid of the effect of sudden changes.

Looking into Fig.7, we have the following observations. 1) DCRNN, MGCGRU, and MGCGRU perform better than FC-LSTM. FC-LSTM only makes use of the temporal dependency of several recent times, which could hardly model the complex crowd flows like that in Fig.7. 2) MGCGRU and ST-MGCN generate better prediction results than DCRNN on morning peaks in both Fig.7(a) and Fig.7(b). This is because MGCGRU also incorporates the flow features of regions into

Table 2. Effect of Spatial Correlation Modeling on the Mobike Dataset Based on MAE

Timestep	Region Distance	Flow Exchange	Flow Similarity	MGCGRU
1	0.291 9	0.292 0	0.291 0	**0.289 2**
2	0.300 6	0.301 6	0.300 9	**0.296 6**
3	0.313 5	0.314 5	0.315 0	**0.308 1**
4	0.319 7	0.321 0	0.321 8	**0.314 1**
5	0.324 9	0.326 5	0.326 8	**0.318 4**
6	0.333 6	0.335 5	0.335 4	**0.326 1**

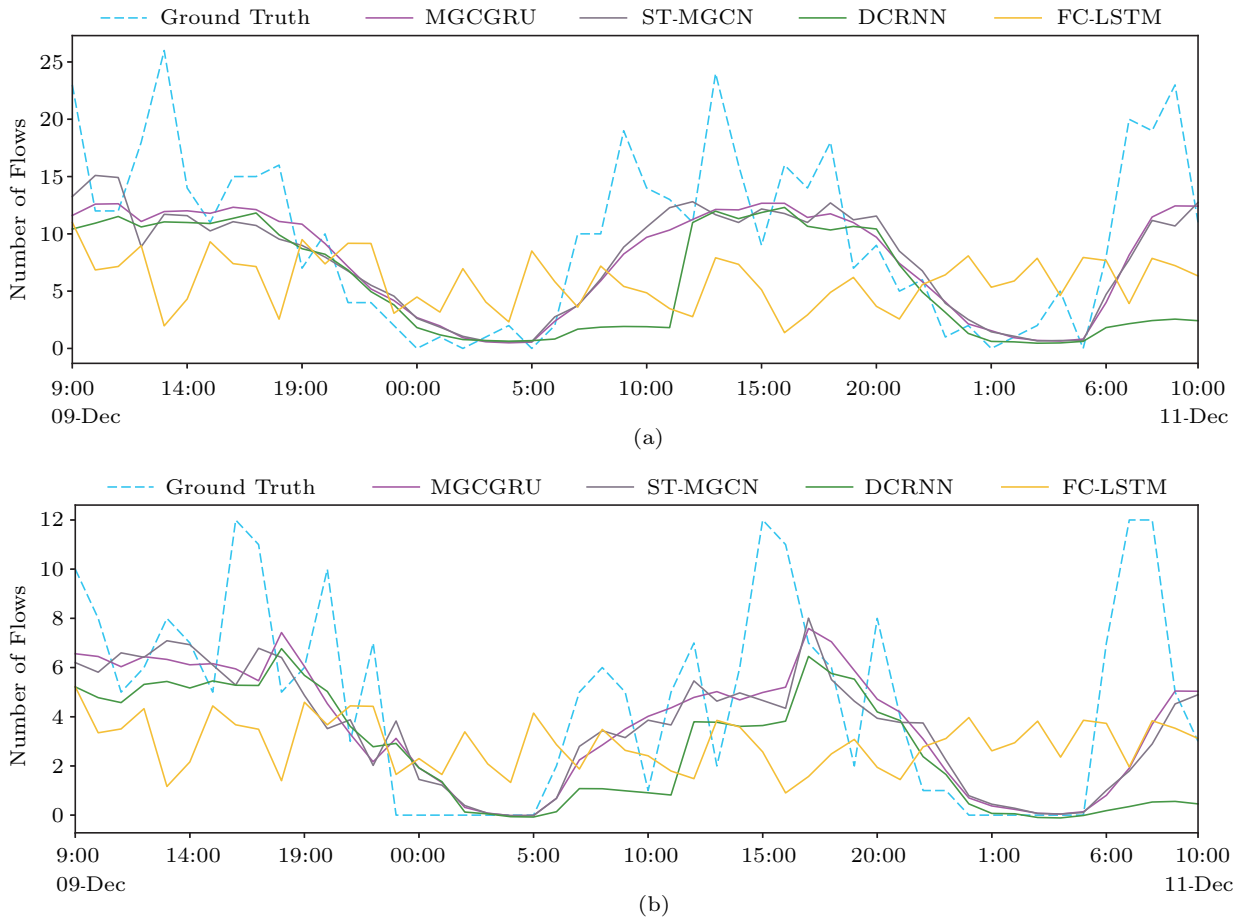


Fig.7. Crowd flows of particular regions. (a) Around Xijiao School and Shanghai Nanshan Hospital. (b) Around residential areas named Zixiang and Lingguang Garden.

the spatial dependency modeling, which enhances this method for modeling large crowd flows like morning or evening peaks. From the records of 6:00 a.m. and 10:00 a.m. in Fig.7, MGCGRU always fits the uptrend of flows on morning peaks more quickly than DCRNN. 3) To analyze crowd flows or rebalance public traffic resources, we tend to pay more attention to large flows. As shown in Fig.7, MGCGRU is more likely to make a better prediction for large flows. For example in the morning of December 11th, the value of flows increases suddenly, and MGCGRU still makes a better prediction than the other methods, thereby MGCGRU also performs well to adapt to the sudden changes of crowd flows. 4) The complex structure of ST-MGCN may bring in some instability. We can observe that the results of MGCGRU are smoother than those of ST-MGCN in Fig.7(b), which means MGCGRU is more robust. The smooth prediction results can better reflect the patterns of crowd flows, providing a meaningful outcome for urban flow analysis.

5.6 Effect of Region Partition

To evaluate the impacts of the radius of neighborhood *epsilon* and the minimum number of points required to form a dense region $MinPts$, we vary *epsilon* from 4 to 12 and $MinPts$ from 20 to 40. The ranges of the two parameters can be set according to the actual demands of prediction tasks. In fact, we should not make the urban regions too small or too large. If the regions are small, the graph will become too complex, which will result in high space complexity. In contrast, the correlations between large regions are usually ambiguous. Therefore, we choose the suitable epsilon and MinPts, which conduces to the proper region size.

For presenting the results lucidly, we vary a parameter and fix the other one. As shown in Fig.8, we can find that the results of region partition will affect the prediction a lot, because the changes on *epsilon* or $MinPts$ may lead to a very different partition result. In our prediction task, when $epsilon = 7$ and $MinPts = 30$, our model achieves the best results.

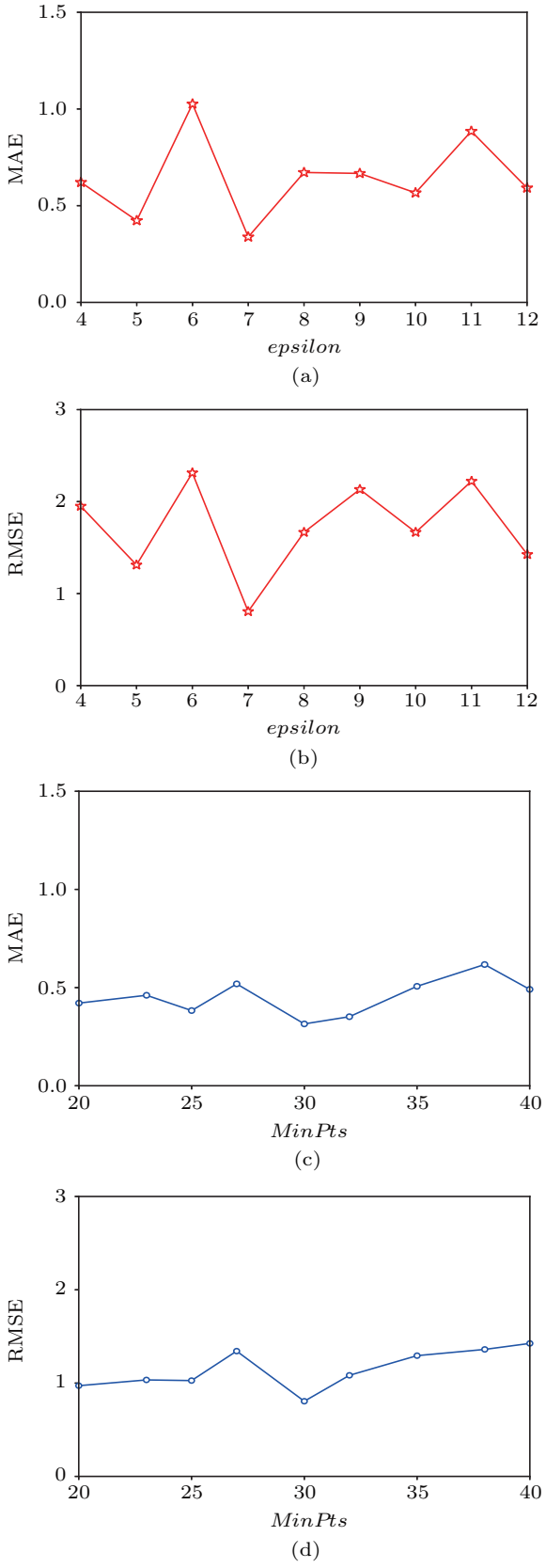


Fig. 8. Impacts of varying DBSCAN parameters *epsilon* and *MinPts*. (a) $MinPts = 30$. (b) $MinPts = 30$. (c) $epsilon = 7$. (d) $epsilon = 7$.

5.7 Effect of Model Parameters

To evaluate the impacts of the number of layers l, we vary l from 1 to 5, fix the rest parameters to their default values and test the prediction errors of different timesteps in Fig.9(b). Similarly, we vary k from 1 to 5, and show the results in Fig.9(d). We observe that along with the increase of l and k, the MAE decreases, but just a little bit. With the increase of l and k, the complexity of the model increases sharply, and the fitting capability of the model is stronger. At the same time, the risk of over-fitting increases unconsciously, and the model will take a lot more time for training. From Fig.9(b)and Fig.9(d), we can also find that the average of MAE decreases more slowly when l and k are increasing.

To obtain higher efficiency-cost ratios, we fit the curves of the training time and MAE when l and k vary. As shown in Fig.9(a) and Fig.9(c), the curves of two parameters are all convex downward. Hence, the best l and k can be found between 1 and 5. Because l and k can only be integers, the possible value of best l and k must be the points in Fig.9(a) and Fig.9(c) respectively. As a result, the model will have the highest efficiency-cost ratio when $l = 3$ and $k = 2$, because the points of $l = 3$ and $k = 2$ are under the curves of the training time and MAE, and they are the farthest away from the curves.

6 Conclusions

In this paper, we first leveraged the density-based clustering method to segment the city into functional regions and studied the region-level urban flow prediction problem, in particular for flows in highly dynamic environments like developing cities. To address the problem, we proposed a deep learning based method for predicting different types of crowd flows named Multi-Graph Convolution Gated Recurrent Units, which can capture the temporal and multiple spatial dependencies. Specifically, we first captured the multiple spatial dependencies such as neighborhood information, flow exchange frequency, and flow similarity adaptively using our multi-graph convolution operator. Then, we captured the temporal dependency by applying the encoder-decoder architecture in our model. Finally, we evaluated our MGCGRU model on two different real-world datasets in Shanghai. The experimental results demonstrated that the proposed MGCGRU outperformed state-of-the-art methods.

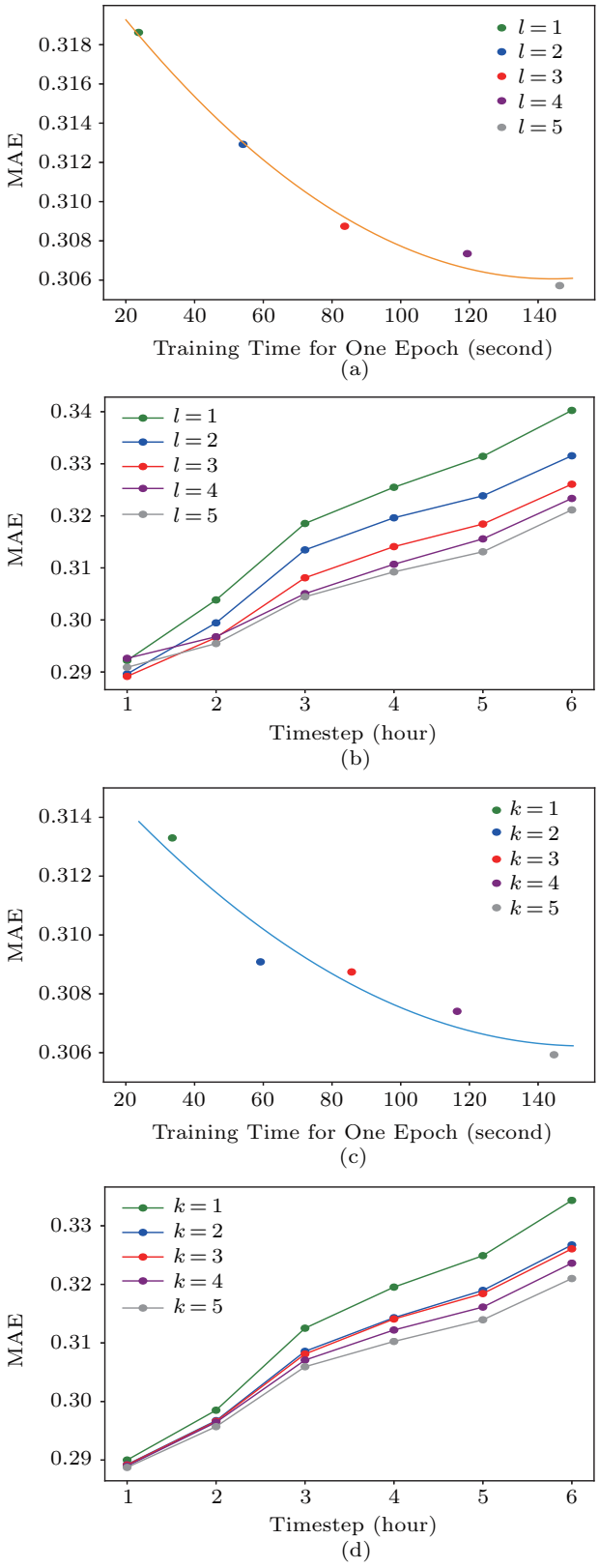


Fig.9. Effect of the number of layers l and the reception fields k. (a) Impacts of varying l on the MAE and the training time. (b) Impacts of varying l on the MAE of different timesteps. (c) Impacts of varying k on the MAE and the training time. (d) Impacts of varying k on the MAE of different timesteps.

By exploiting multiple correlations, our method can mine rich internal modes without complex mechanisms. As a result, MGCGRU can achieve the best performance with relatively lower complexity in highly dynamic environments. For future work, we will investigate how to efficiently fuse the various influences in our model for improvement.

References

[1] Zheng Y, Capra L, Wolfson O, Yang H. Urban computing: Concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology*, 2014, 5(3): Article No. 38.

[2] Zhang J B, Zheng Y, Qi D K. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *Proc. the 31st AAAI Conference on Artificial Intelligence*, February 2017, pp.1655-1661.

[3] Zheng Z, Yang Y, Liu J *et al.* Deep and embedded learning approach for traffic flow prediction in urban informatics. *IEEE Transactions on Intelligent Transportation Systems*, 2019, 20(10): 3927-3939.

[4] Sun J, Zhang J, Li Q *et al.* Predicting citywide crowd flows in irregular regions using multi-view graph convolutional networks. arXiv:1903.07789, 2019. https://arxiv.org/abs/1903.07789, August 2019.

[5] Du B, Peng H, Wang S *et al.* Deep irregular convolutional residual LSTM for urban traffic passenger flows prediction. *IEEE Transactions on Intelligent Transportation Systems*. doi:10.1109/TITS.2019.2900481.

[6] Chai D, Wang L, Yang Q. Bike flow prediction with multi-graph convolutional networks. In *Proc. the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, November 2018, pp.397-400.

[7] Geng X, Li Y, Wang L *et al.* Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proc. the 33rd AAAI Conference on Artificial Intelligence*, January 2019, pp.3656-3663.

[8] Ramaswami A, Russell A G, Culligan P J *et al.* Meta-principles for developing smart, sustainable, and healthy cities. *Science*, 2016, 352(6288): 940-943.

[9] Ai Y, Li Z, Gan M *et al.* A deep learning approach on short-term spatiotemporal distribution forecasting of dockless bike-sharing system. *Neural Computing and Applications*, 2019, 31(5): 1665-1677.

[10] Shuman D I, Narang S K, Frossard P, Ortega A, Vandergheynst P. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 2013, 30(3): 83-98.

[11] Li Y X, Zheng Y, Zhang H C, Chen L. Traffic prediction in a bike-sharing system. In *Proc. the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, November 2015, Article No. 33.

[12] Yao H, Wu F, Ke J, Tang X, Jia Y, Lu S, Gong P, Ye J, Li Z. Deep multi-view spatial-temporal network for taxi demand prediction. In *Proc. the 32nd AAAI Conference on Artificial Intelligence*, February 2018, pp.2588-2595.

[13] Holmgren J, Aspegren S, Dahlstroma J. Prediction of bicycle counter data using regression. *Procedia Computer Science*, 2017, 113: 502-507.

[14] Kumar S V, Vanajakshi L. Short-term traffic flow prediction using seasonal ARIMA model with limited input data. *European Transport Research Review*, 2015, 7(3): Article No. 21.

[15] Abadi A, Rajabioun T, Ioannou P A. Traffic flow prediction for road transportation networks with limited traffic data. *IEEE Transactions on Intelligent Transportation Systems*, 2015, 16(2): 653-662.

[16] Li Y, Yu R, Shahabi C, Liu Y. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *Proc. the 6th International Conference on Learning Representations*, April 2018.

[17] Cheng A Y, Jiang X, Li Y F, Zhang C, Zhu H. Multiple sources and multiple measures based traffic flow prediction using the chaos theory and support vector regression method. *Physica A: Statistical Mechanics and its Applications*, 2017, 466: 422-434.

[18] Achar A, Bharathi D, Kumar B A *et al*. Bus arrival time prediction: A spatial Kalman filter approach. *IEEE Transactions on Intelligent Transportation Systems*. doi:10.1109/TITS.2019.2909314.

[19] Liu J M, Sun L L, Li Q, Ming J C, Liu Y C, Xiong H. Functional zone based hierarchical demand prediction for bike system expansion. In *Proc. the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2017, pp.957-966.

[20] Liu J M, Sun L L, Chen W W, Xiong H. Rebalancing bike sharing systems: A multi-source data smart optimization. In *Proc. the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2016, pp.1005-1014.

[21] Srivastava N, Mansimov E, Salakhutdinov R. Unsupervised learning of video representations using LSTMs. arXiv:1502.04681, 2015. https://arxiv.org/abs/1502.04681, August 2019.

[22] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv:1409.0473, 2014. https://arxiv.org/abs/1409.0473, August 2019.

[23] Cho K, van Merrienboer B, Gulcehre C *et al*. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078, 2014. https://arxiv.org/abs/1406.1078, August 2019.

[24] Thirumalai C, Koppuravuri R. Bike sharing prediction using deep neural networks. *JOIV: International Journal on Informatics Visualization*, 2017, 1(3): 83-87.

[25] Shi X J, Chen Z R, Wang H, Yeung D Y, Wong W K, WOO W C. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Proc. the 2015 Annual Conference on Neural Information Processing Systems*, December 2015, pp.802-810.

[26] Bruna J, Zaremba W, Szlam A *et al*. Spectral networks and locally connected networks on graphs. arXiv:1312.6203, 2013. https://arxiv.org/abs/1312.6203, August 2019.

[27] Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proc. the 2016 Annual Conference on Neural Information Processing Systems*, December 2016, pp.3844-3852.

[28] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks. arXiv:1609.02907, 2016. https://arxiv.org/abs/1609.02907, August 2019.

[29] Zhang X, He L, Chen K, Luo Y, Zhou J, Wang F. Multi-view graph convolutional network and its applications on neuroimage analysis for Parkinson's disease. arXiv:1805.08801, 2018. https://arxiv.org/abs/1805.08801, August 2019.

[30] Yao H, Tang X, Wei H, Zheng G, Yu Y, Li Z. Modeling spatial-temporal dynamics for traffic prediction. arXiv:1803.01254, 2018. https://arxiv.org/abs/1803.01254, August 2019.

[31] Yuan N J, Zheng Y, Xie X *et al*. Discovering urban functional zones using latent activity trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 2015, 27(3): 712-725.

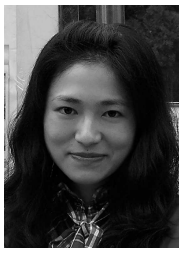
[32] Erman J, Arlitt M F, Mahanti A. Traffic classification using clustering algorithms. In *Proc. the 2nd Annual ACM Workshop on Mining Network Data*, September 2006, pp.281-286.

[33] Cho K, van Merrienboer B, Bahdanau D *et al*. On the properties of neural machine translation: Encoder-decoder approaches. arXiv:1409.1259, 2014. https://arxiv.org/abs/1409.1259, August 2019.

[34] Friedman J H. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 2001, 29(5): 1189-1232.

[35] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks. In *Proc. the 2014 Annual Conference on Neural Information Processing Systems*, December 2014, pp.3104-3112.

Qiang Zhou received his B.E. degree in software engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, in 2016. He is currently working toward his Ph.D. degree in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing. His current research interests include urban computing, data mining, deep learning and spatio-temporal prediction.



Jing-Jing Gu received her B.E. degree in computer science, and Ph.D. degree in computer science and technology from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, in 2005 and 2011, respectively. She is currently an associate professor at the College of Computer Science and Technology, NUAA. Her current research interests include mobile data mining, urban computing, and intelligent system.

Chao Ling is a Master student in Nanjing University of Aeronautics and Astronautics, Nanjing, majoring in computer science and technology. His research interests mainly include machine learning and spatio-temporal data mining.

Wen-Bo Li is a Bachelor student in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing. His research interests include computer vision, machine learning, and algorithms.

Yi Zhuang received her Ph.D. degree in computer science and technology from the Department of Computer Science, Nanjing University of Aeronautics and Astronautics, Nanjing, in 1981. Now she is a professor and Ph.D. supervisor of the College of Computer Science and Technology at Nanjing University of Aeronautics and Astronautics, Nanjing. Her research interests include network distributed computing, information security and dependable computing.

Jian Wang is a professor of the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing. He got his Ph.D. degree in computer application technology from Department of Computer Science and Technology, Nanjing University, Nanjing, in 1998. His interest includes applied cryptography, system security, etc.