Dynamic Graph Convolutional Network for Long-Term Traffic Flow Prediction with Reinforcement Learning

Hao Peng[a,*], Bowen Du[b], Mingsheng Liu[c], Mingzhe Liu[b], Shumei Ji[d], Senzhang Wang[e], Xu Zhang[f], Lifang He[g]

[a]*School of Cyber Science and Technology, Beihang University, Beijing 100083, China*
[b]*School of Computer Science and Engineering, Beihang University, Beijing 100083, China*
[c]*Shijiazhuang Institule of Railway Technology, Shijiazhuang 050041, China*
[d]*Shijiazhuang Iron & Steel Co. Ltd., Shijiazhuang 050031, China*
[e]*School of Computer Science and Engineering, Central South University, Changsha 410083, China*
[f]*National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China*
[g]*Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA 18015, USA*

Abstract

Exploiting deep learning techniques for traffic flow prediction has become increasingly widespread. Most existing studies combine CNN or GCN with recurrent neural network to extract the spatio-temporal features in traffic networks. The traffic networks can be naturally modeled as graphs which are effective to capture the topology and spatial correlations among road links. The issue is that the traffic network is dynamic due to the continuous changing of the traffic environment. Compared with the static graph, the dynamic graph can better reflect the spatio-temporal features of the traffic network. However, in practical applications, due to the limited accuracy and timeliness of data, it is hard to generate graph structures through frequent statistical data. Therefore, it is necessary to design a method to overcome data defects in traffic flow prediction. In this paper, we propose a long-term traffic flow prediction method based on dynamic graphs. The traffic network is modeled by dynamic traffic flow probability graphs, and graph convolution is performed on the dynamic graphs to learn spatial features, which are then combined with LSTM units to learn temporal features. In particular, we further propose to use graph convolutional policy network based on reinforcement learning to generate dynamic graphs when the dynamic graphs are incomplete due to the data sparsity issue. By testing our method on city-bike data in New York City, it demonstrates that our model can achieve stable and effective long-term predictions of traffic flow, and can reduce the impact of data defects on prediction results.

Keywords: Traffic Flow Prediction, Dynamic Graph, Graph Convolutional Policy Network, Spatio-Temporal Prediction, Reinforcement Learning

1. Introduction

With the development of deep learning technology, exploiting deep neural networks models for the task of traffic flow prediction has become increasingly popular [1]. The research on traffic flow prediction is based on various traffic data provided by relevant institutions and organizations. It collects statistics on the traffic flow in a certain area and predicts the traffic flow at various places or roads in the area in the future. In recent years, new models for traffic flow prediction have been developed, especially based on CNN (Convolutional Neural Network) [2, 3, 4, 5, 6] and GCN (Graph Convolutional Network) [7, 8, 9, 10]. The former uses convolution after contiguous or similar places are aggregated, and the latter models each place in the traffic network into a graph structure and then performs convolution operations.

Recent studies have shown that modeling the traffic network as a graph is more effective than converging the places in the network and then convolving it [11, 8]. The graph can represent the correlations between different areas in the transportation network, as well as the traffic flow information associated with each area [12]. As graph neural networks evolve, how to apply various graph neural networks for traffic flow prediction has attracted increasing research attention. However, it is worth noting that the traffic network is constantly changing. In the long-term prediction of traffic flow, the prediction performance will be less promising if the traffic network is modeled as a static graph without considering the dynamic changes of the traffic network [13, 14].

Another problem of long-term traffic flow prediction is the integrity and accuracy of the data source. In practical applications, the data collected by some institutions and organizations is not real-time [15]. As shown in Figure 1, sometimes the data

*Corresponding author
Email addresses: `penghao@act.buaa.edu.cn` (Hao Peng), `dubowen@buaa.edu.cn` (Bowen Du), `liums601001@sina.com` (Mingsheng Liu), `mzliu1997@gmail.com` (Mingzhe Liu), `jishumei@hbisco.com` (Shumei Ji), `szwang@csu.edu.cn` (Senzhang Wang), `zhangxu@cert.org.cn` (Xu Zhang), `lih319@lehigh.edu` (Lifang He)

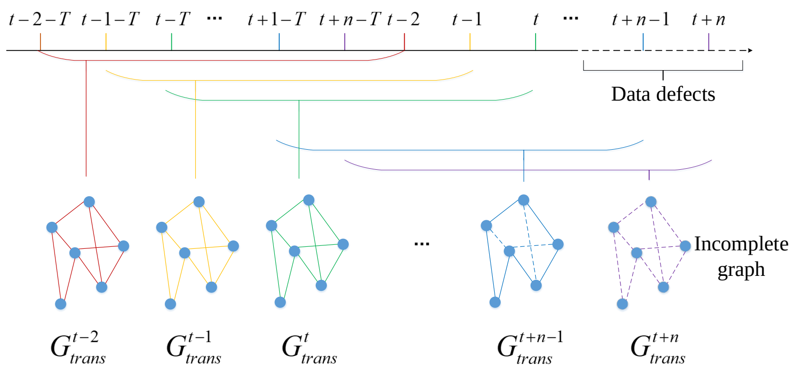


Figure 1: Data defects that may occur in practical applications of traffic flow prediction.

will be missing or the previous data will be collected after a period of time, which will have a great impact on the prediction of traffic flow [16, 17]. This situation is named as “data defects” in long-term traffic flow prediction. These defective data will eventually become available and completed with the continuous collection and synchronization of various institutions and organizations, but before they are available, direct use of these data to predict traffic flow will obviously cause large deviations. Therefore, it is necessary to design a method to ensure eliminating the effect caused by data defects in spatial and temporal relations in traffic flow prediction.

To address the above problems, we have sufficient motivation to design a traffic flow spatio-temporal prediction model that accepts and generates dynamic graphs, based on the changing characteristics of traffic graph modeling over time. The generated dynamic graphs can improve the quality of the spatial features captured by the model, and more importantly, can overcome the adverse effects caused by data defects.

In this paper, we propose a long-term prediction model for traffic flow using dynamic graph generation to overcome data defects. In this model, the stations involved in the graph are based on recent real-time city bike data in New York City, where each station is a node in the graph. Based on the statistics of the traffic flow of each station within a certain period of time, a probability graph of the traffic flow transfer between each station is generated, and the inflow and outflow of traffic are taken as the features of the station node. Assuming that the data itself is defective (i.e., it takes a long time to obtain relevant data and calculate the traffic flow in the current time period), we use the GCPN model (Graph Convolutional Policy Network) [18] to perform reinforcement learning on existing traffic flow graphs, and utilize complete historical data as the learning environment. According to the characteristics of our traffic flow graph, the states, actions, rewards, etc. in the reinforcement learning process are designed to generate dynamic graphs sequence on the time series. We perform graph convolution operation on the generated graphs to extract spatial features, and use LSTM (Long-Short Term Memory) [19] to extract temporal features. Long-term prediction of traffic flow is achieved by generating dynamic graphs and adjusting the time interval between the LSTM input sequence and output.

The contributions of this work are summarized as follows:

- Model travel records on the traffic network as traffic flow transfer graphs and traffic flow probability graphs, and use dynamic graphs in the overall framework to reduce traffic flow prediction error.

- Aiming at the possible data defects in practical applications, propose to apply dynamic graph generation to the long-term prediction task of traffic flow. Make the generation of traffic flow transfer graph consistent with the Markov Decision Process, and utilize the related algorithms in the Graph Policy Convolutional Network to generate dynamic graphs through reinforcement learning.

- Design a spatio-temporal prediction network with dynamic graphs available. This deep learning network combines graph convolutional network (GCN) and LSTM. The former is used to learn the spatial features of the traffic network, and the latter is used to learn the temporal features of the traffic network.

- Use the real city bike data in New York City to conduct long-term traffic flow prediction experiments. Our dynamic graph method is better than other spatio-temporal prediction methods under any prediction period, especially when there are data defects. When data defects exist, the Root Mean Square Error (RMSE) of the prediction results of our dynamic graph method is reduced by up to 32.4% compared with other static graph methods.

The paper is structured as follows: Section 2 reviews the related work. Section 3 describes the technical details involved in the long-term traffic flow prediction methods. Experimental setup and results are discussed in Section 4. The conclusion of the paper is in Section 5.

2. Related Work

We review the related work from two aspects: *deep learning model of traffic flow prediction* and *dynamic graph generation*.

2.1. Deep learning model of traffic flow prediction

In recent years, with the development of deep learning technology, various neural network models have been applied to the task of traffic flow prediction [1, 20]. Since traffic flow prediction involves both spatial and temporal characteristics, research on traffic flow prediction is mainly focused on improving prediction performance from these two aspects. In terms of temporal feature extraction, temporal network models such as LSTM (Long-Short Term Memory) and GRU (Gated Recurrent Unit) are widely used. In terms of spatial feature extraction, the modeling methods of the traffic network in recent years can be divided into two types: Euclidean space (grid format) [3, 21, 22, 23, 24, 25] and non-Euclidean space (graph format) [8, 26, 27, 28, 29, 30].

In the research of traffic flow prediction using Euclidean space, Liu et al. [3] proposed the ConvLSTM model, which maps traffic flow data to a one-dimensional vector space, combines the one-dimensional spatial information vectors at different times into a matrix, and convolves through CNN and combines with LSTM to extract the spatio-temporal characteristics of traffic flow; Zhang et al. [22] proposed the ST-ResNet model, which divides cities into grid-based maps according to latitude and longitude, each grid represents an area, and counts the inflow and outflow in the area and converts it into a 2-channel matrix; Zhang et al. [21] divided the actual map into a grid map, modeled the grid map as a weighted directed graph, and then fed it into a convolutional neural network for learning after the graph embedding operation. Guo et al. [23] proposed RSTN, which combines CNN, LSTM and ConvLSTM modules through residual connections to capture spatio-temporal and extraneous dependencies. The traditional prediction problem is regarded as a learning residual function of travel density in each time interval. Chen et al. [31] proposed MGSTC, which explores multiple spatio-temporal correlations through multiple gated spatio-temporal CNN branches, and dynamically combines spatio-temporal features with external factors.

In the research of traffic flow prediction using non-Euclidean space modeling, Yu et al. [9] proposed the STGCN framework. To make full use of spatial information, a general graph structure was used to model the traffic network, where the stations in the traffic network were modeled as the nodes. The nodes in the graph have different observation values at different times, and they utilized the graph convolution and gated linear units to extract spatial and temporal features respectively; Chai et al. [28] proposed the use of multi-graph convolutional networks to predict traffic flow. By establishing multiple graphs based on distances, traffic interactions and correlations between stations in the traffic network and fusing these graphs, they learned the spatio-temporal features through graph convolution and encoding-decoding networks. Zhao et al. [8] proposed the T-GCN model, which modeled roads in traffic network as nodes in the graph, edges representing the connections between roads, and captured spatio-temporal features through graph convolutional network and gated recurrent unit.

It is worth noting that there have been some researches [32, 33] on the use of reinforcement learning in the field of traffic, but the scenarios they orient are not traffic flow predictions, but mostly optimization problems. Schultz et al. [32] developed a deep learning model for calibrating transportation simulators and reinforcement learning to solve the problem of optimal planning for travelers on the networks. Liu et al. [33] used DQN algorithm to optimize traffic light control strategies. There is no direct research on applying reinforcement learning to dynamic graph generation in traffic flow prediction.

2.2. Dynamic graph generation

With the widespread application of graph neural networks, knowledge graphs and other technologies, research is no longer limited to static graphs, but began to study dynamic graphs. In terms of dynamic graph generation, a more general framework is to learn the representation of dynamic graphs, and perform link prediction of graphs based on the learned representations [34, 35, 36]. Sankar et al. [35] proposed the DySAT model that predicted links by learning the node representation of a dynamic graph, and applied the attention mechanism to the dynamic graph to capture the changes of the same node at all times; Trivedi et al. [34] proposed the unsupervised DyRep model, describing the changing process of dynamic graphs as events, and encoding these events for learning to achieve link prediction.

Another novel approach is to generate graphs through reinforcement learning [37, 18]. Khan et al. [37] proposed to treat each node in the graph as an agent, which shared action space and state space. They perform parameter update through graph convolutional network and policy gradient algorithm. You et al. [18] proposed the GCPN model, which treated the entire graph as an agent, computed nodes embedding and predicted actions, and optimized the policy network through policy gradient algorithm.

In summary, there is currently no research that explores dynamic graph generation to the deep learning framework for long-term traffic flow prediction in the presence of data defects.

3. Model

3.1. Problem definition

Given traffic flow data, our main task is to make more accurate long-term predictions of traffic flow in the presence of **data defects**. The data defects mainly refer to: due to the delay in the statistics of transportation data by relevant institutions or organizations, and the coverage of transportation data calculated by different institutions or organizations are different, it takes a long time to obtain complete traffic flow data in a city. As shown in Figure 1, generating a graph based on traffic data requires data support within a long time T. If the data defect due to delay occurs at time t, the graph of time steps after t will be affected. And this part of the defective data will not be available in the short term. Therefore, since there is not enough data to support the moment when predictions are needed, historical traffic flow data are necessary to generate current travel records and traffic flow features that may occur in the short term, and then make further long-term forecasts.

In order to describe the problem more clearly, we define the following key concepts:

Definition 1: Traffic flow graph G^t. We use weighted graph $G^t = (V^t, E^t)$ to describe the topology between stations in the traffic network, where $V = \{v_1, v_2, \cdots, v_N\}$ is the set of station nodes (N is the number of stations), and E is the set of weighted edges. The weight of the edges varies depending on the content represented by the graph. We describe the graph G by the adjacency matrix $A \in \mathbb{R}^{N\times N}$. G is a dynamic graph, different times t correspond to different G^t and E^t. In our model, there are two types of traffic flow graphs: traffic flow transfer graph G^t_{trans} and traffic flow probability graph $G^t_{prob} = \{G^t_{in}, G^t_{out}\}$. The specific construction method of graph G will be introduced in the following section.

Definition 2: Traffic flow feature X. The traffic flow of each station in the traffic network at time t includes two parts, namely the inflow I^t and the outflow O^t. For a traffic network with N nodes, $I^t = [i^t_1, i^t_2, \cdots, i^t_N]$, $O^t = [o^t_1, o^t_2, \cdots, o^t_N]$, where $i^t_j, o^t_j \in \mathbb{R}$ are the inflow and outflow of the station j at time t. The traffic flow feature X^t at time t is $[I^t, O^t] \in \mathbb{R}^{N\times 2}$.

Definition 3: Long-term prediction. A traffic flow prediction task is to predict the traffic flow X^{t+n} at time step $t+n$ under the premise of known traffic flow $[X^1, \cdots, X^t]$. In this paper, we specify the task as a long-term prediction when the actual time length corresponding to n is greater than 1 day. Otherwise, the task is a short-term prediction.

Definition 4: Data defects. Assume that the correct traffic flow feature at time t is X^t. Data defect refers to the fact that the counted traffic flow $\tilde{X}^t$ at time t does not match the correct traffic flow X^t due to errors in data statistics, that is, $\tilde{X}^t \neq X^t$, which leads to the wrong traffic flow graph $\tilde{G}^t$.

According to the above definition, our problem can be described as two parts: (1) In the case of data defects in the time range $[t, t+n]$, according to the historical time range $[t-T, t+n-T]$ with complete data records $\{G^{t-T}, G^{t+1-T}, \cdots, G^{t+n-T}\}$, the traffic flow graph G^t can be autonomously generated $\{G^t, G^{t+1}, \cdots, G^{t+n}\}$. (2) In the time range $[t-m, t]$, according to the dynamic graph sequence $\{G^{t-m}, G^{t+1-m}, \cdots, G^t\}$ and the traffic flow feature sequence $\{X^{t-m}, X^{t+1-m}, \cdots, X^t\}$, predict the future long-term traffic flow X^{t+n}.

3.2. Framework overview

The overview of our dynamic graph convolutional network for long-term traffic flow prediction is shown in Figure 2. Our model can be roughly summarized into three parts:

Traffic flow probability graph modeling. We utilize graph to reflect the topological relationship between various stations for the traffic network of a city. The traffic flow probability graph we propose is a directed weighted graph. The stations in the traffic network correspond to the nodes in the graph, and the edge weights represent the probability that the outflow or inflow of the station node flows to its neighbor nodes. The graph structure generated based on the source data will be used for reinforcement learning to generate dynamic graphs, and we employ graph convolution to extract spatial features and add them to the prediction network.

Dynamic graph generation. As mentioned in previous sections, because the data in practical applications may have defects, we need the traffic flow graph to be able to generate short-term dynamic graph sequence under the guidance of historical complete data via reinforcement learning. The short-term dynamic graph completion makes the long-term prediction result more accurate. Dynamic graph generation is implemented by policy gradient algorithm, and the differences of the graph are used as reward signals for reinforcement learning to generate action sequences on the traffic flow transfer graph.

Spatio-temporal prediction network. After completing the short-term dynamic graph sequence, we obtain the sequence of spatial features through graph convolution. Then the spatial feature sequence is used as the input of the LSTM unit, and the long-term time feature is captured by adjusting the time interval between the output and input of the LSTM unit.

In summary, for the traffic flow prediction task with data defects, we model the topology structure of the traffic network, complete the obtained graph through reinforcement learning, and then combine the graph convolution and LSTM to capture spatio-temporal features for long-term traffic flow prediction. Since our proposed method is oriented to actual application scenarios, it can utilize the newly collected complete data to continuously update the model parameters. When the defective data is continuously collected over time until it is completely available, the collected data will first be used to update the parameters of the dynamic graph generation. Then the newly generated dynamic graph will be applied to the spatio-temporal prediction network. The following sections introduce the specific implementation methods of these parts in detail.

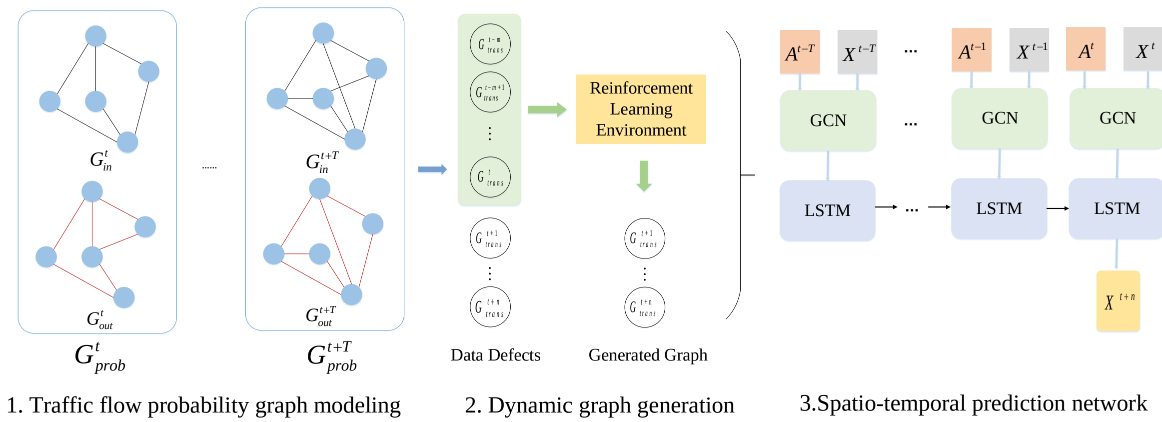


Figure 2: An overview of our spatio-temporal prediction network model for long-term traffic flow prediction

3.3. *Traffic flow graph modeling*

In recent research of traffic flow prediction, graph neural networks have been gradually applied to the spatial feature extraction for traffic networks. In a traffic network, there are several *Points of Information* (POI), e.g., a building, a park, or a bus stop. Traffic flow can be seen as the process of people starting from one POI to another POI through various transportation vehicles. To make the model interpretation more popular and consistent with the data used, we collectively refer to these POIs as *stations*.

In the research of using Euclidean space to model traffic networks, the stations are often alone or aggregated in a two-dimensional grid structure, which obviously ignores the topology information between the stations. In the task of traffic flow prediction, in order to better capture the spatial features, it is necessary to build a model of the traffic network in non-Euclidean space. For a traffic network, if we consider the stations in it as nodes in the graph, then for the traffic flow prediction task, the links between the nodes should be determined by the traffic flow. That is, if someone departs from station v_1 to station v_2, a potential link will be created between nodes v_1 and v_2. If there are more travel records between the two stations, it means that the traffic flow between the two stations is greater, and the weight of the links should be greater.

Based on the above ideas, a directed weighted graph is needed to model the traffic network. We first propose the concept of *traffic flow transfer graph*. For a traffic network with N stations, it is assumed that the sampling time interval of the traffic flow is T, and the traffic flow transfer graph corresponding to the time t is denoted as $G^t_{trans} = (V^t, E^t_{trans})$. We record the travel records provided by relevant institutions or organizations as $r^{\tau,\tau'}_{v_i.v_j}$, which means that someone in the current traffic network departs from station v_i at time τ and arrives at station v_j at time τ'. Then the edge weight between any two nodes in the traffic flow transfer graph G^t_{trans} should be the number of travel records from station v_i to station v_j in the range of time $(t-T, t]$, that is:

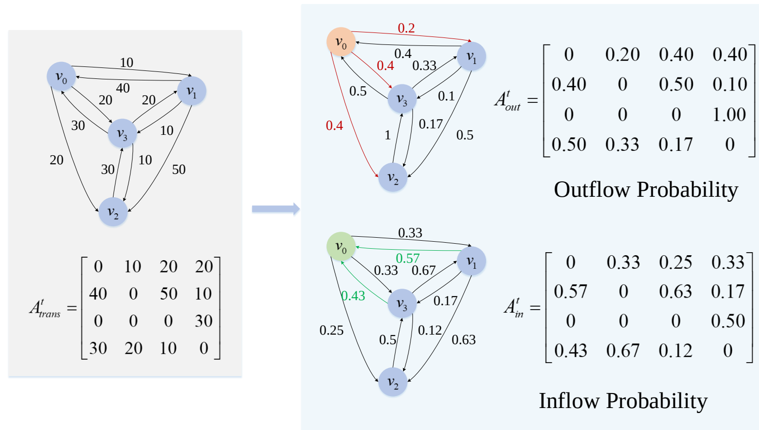
$$w^t_{trans}(v_i, v_j) = count_{\tau,\tau' \in (t-T,t]}\left(r^{\tau,\tau'}_{v_i.v_j}\right), \tag{1}$$

where $w^t_{trans}(v_i, v_j)$ represents the weight of the edge (v_i, v_j) and $count(\cdot)$ is the counting function. On this basis, we hope to better reflect the possibility that the traffic flow of each node is transferred to the neighbor nodes, so we propose a *traffic flow probability graph* model. It is worth noting that we need to consider the probability of outflow and inflow at the same time, and use the probability as the weight of the edge in the graph. We use a simple probability calculation method. In the outflow probability graph $G^t_{out} = (V^t, E^t_{out})$ and the inflow probability graph $G^t_{in} = (V^t, E^t_{in})$, for node v_i, the probability of traffic outflow and traffic inflow to node v_j is:

$$w^t_{out}(v_i, v_j) = \begin{cases} 0, & if \sum\limits_{v_k \in V} w^t_{trans}(v_i, v_k) = 0 \\ \frac{w^t_{trans}(v_i,v_j)}{\sum\limits_{v_k \in V} w^t_{trans}(v_i,v_k)}, & otherwise \end{cases} \tag{2}$$

$$w^t_{in}(v_i, v_j) = \begin{cases} 0, & if \sum\limits_{v_k \in V} w^t_{trans}(v_k, v_i) = 0 \\ \frac{w^t_{trans}(v_j,v_i)}{\sum\limits_{v_k \in V} w^t_{trans}(v_k,v_i)}, & otherwise \end{cases} \tag{3}$$

Figure 3 shows an example of generating the traffic flow transfer graph and traffic flow probability graph Taking node v_0 as an example, the probability of outflow and inflow to neighbor nodes are different in a specified period of time. The traffic



1. Traffic Flow Transfer Graph 2. Traffic Flow Probability Graph

Figure 3: Example of traffic flow transfer graph converted to traffic flow probability graph.

flow probability graph $G_{prob}^t = \{G_{out}^t, G_{in}^t\}$ can reflect the traffic transfer probability between stations in the current traffic network at time t, laying a foundation for subsequent work of dynamic graph generation and graph convolution.

3.4. Dynamic Graph Generation

In this section we will explain why dynamic graph generation is used in our model and introduce the approach we adopt for dynamic graph generation of traffic flow prediction tasks. We formulate the problem of graph generation as learning an RL (Reinforcement Learning) agent that iteratively adds substructures and edges to the traffic flow probability graph in a traffic-predictive environment.

First, although it is better to use the graph structure to obtain the topology information of the traffic network to extract the spatial features than to mesh the traffic network into grids, it is still flawed to simply use the static graph to model the traffic network. Obviously, the traffic network is changing over the time. Regardless of the topology information contained in the graph structure, it is difficult to capture the changing spatial features using static graphs. So dynamic graphs are a better choice. However, considering the practical application situation, if it requires effective traffic flow prediction in real time, it is hard to generate a graph model at regular intervals. Various travel record data are collected by a number of relevant institutions or organizations, and there will be problems of inconsistent standards, unsynchronized data, and even data missing. It takes time for complete data to be collected. If we ignore the data during this period, it will inevitably affect the results of long-term prediction of traffic flow. Therefore, we propose to add dynamic graph generation to our framework to preserve the integrity of the data as much as possible to reduce the error of long-term prediction.

For dynamic graph generation tasks, the advantage of reinforcement learning is that generating a graph does not need to give a complete graph sequence, only an intermediate state is needed to generate action. Meanwhile, reinforcement learning is capable of directly representing hard constraints and desired structures through the design of environment dynamics and reward function. Through the action sequence, it is feasible to generate a sequence of dynamic graphs at any time interval, which meets the requirements for dynamic graphs in our long-term traffic flow prediction framework. We use the GCPN model [18] for dynamic graph generation, and the environment corresponding to this model can be easily extended to graph generation tasks in other settings. Since reinforcement learning is not the main research content of this paper, we mainly introduce how to make the corresponding graph of the traffic network conform to the reinforcement learning environment of graph generation. For more details of GCPN, please refer to [18].

In order to reduce the difficulty of dynamic graph generation, we use the traffic flow transfer graph G_{trans}^t instead of the traffic flow probability graph G_{prob}^t for generation. After generating a new traffic flow transfer graph, we can get the traffic flow probability graph G_{prob}^t by Equations (2) and (3).

Details of RL Process. We take the dynamic graph generation procedure as Markov Decision Process $MDP < S, A, P, R, \gamma >$, where $S = \{s_i\}$ is the set of states that consists of all possible intermediate and final graphs, $A = \{a_i\}$ is the set of actions that describe the modification made to current graph at each time step, P is the transition dynamics that specifies the possible outcomes of carrying out an action $p(s_{t+1}|s_t, \ldots, s_0, a_t)$. $R(s_t)$ is a reward function that specifies the reward after reaching state s_t, and γ is the discount factor. The dynamic graph generation procedure can be viewed as a trajectory $(s_0, a_0, r_0, \cdots, s_n, a_n, r_n)$, where s_n is the final generated graph. Meanwhile, a modification of a graph at each time

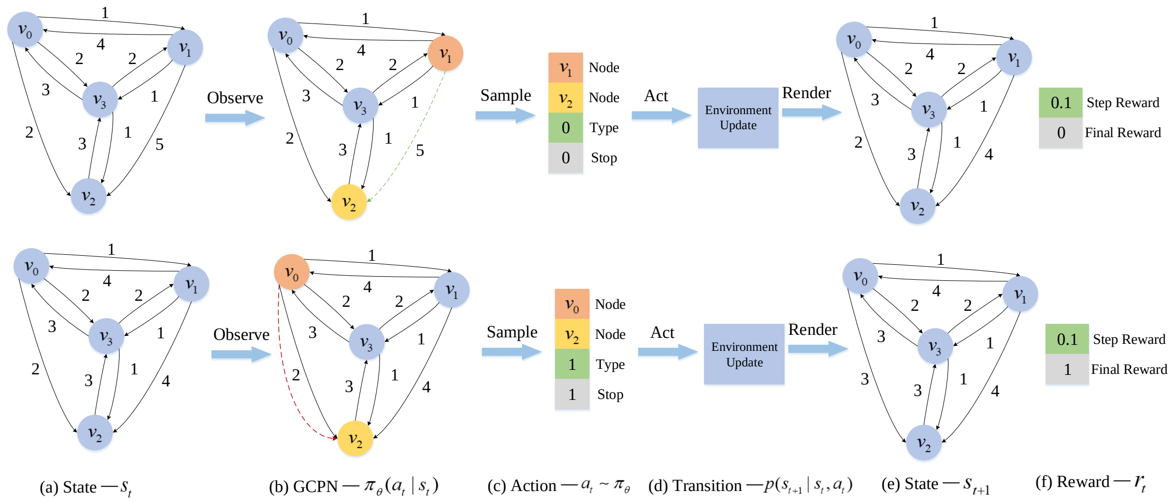


Figure 4: The main steps of dynamic graph generation. (a) is the traffic flow transition graph corresponding to the current state s_t. (b) GCPN performs graph convolution and calculates the node embeddings to generate the policy π_θ. (c) a_t is the action quaternion obtained by the policy π_θ, which is used in (d) state transition to obtain a new state (e) s_{t+1}, and calculate the reward function (f) r_t.

step can be viewed as a state transition distribution: $p(s_{t+1}|s_t,\ldots,s_0) = \sum_{a_t} p(a_t|s_t,\ldots,s_0)p(s_{t+1}|s_t,\ldots,s_0,a_t)$, where $p(a_t|s_t,\ldots,s_0)$ refers to a policy network π_θ. Figure 4 shows the main 6 steps of dynamic graph generation.

- **State space:** The state of the environment s_t is the intermediate generated graph G_t at the time step t. And the state is fully observable by the RL agent. Update the current state according to the predicted action. If the action's $type$ is 1, a new edge is added or edge weight is added by 1 between the two nodes; if the action's $type$ is 0, the edge weight between the two nodes is reduced by 1 or the edge disappears. Note that it is not possible to perform an action of link disappearing between two nodes that are not linked. These infeasible actions will be rejected and the state remains unchanged.

- **Action space:** We design actions similar to the link prediction work in network science. In the traffic flow transfer graph, the generation and disappearance of links correspond to the increase and decrease of the traffic flow between any two stations. We describe the action as a quaternion $a_t = (v_1, v_2, type, stop)$, where v_1 and v_2 represent the station nodes, and $type \in \{0,1\}$ corresponds to whether the link disappears (0) or is newly added (1), $stop \in \{0,1\}$ indicates whether the current action can stop the learning process.

- **State transition:** The traffic flow transition graph G^t_{trans} at time t is the state s_t in the graph generation environment. The definition of G^t_{trans} is consistent with Section 3.3, that is, the entire graph is observed by the reinforcement learning agent. In the specific generation task, the start graph used for graph generation should be a traffic flow transfer graph that can be statistically obtained from data without defects.

- **Reward:** Reward is used to guide the behavior of reinforcement learning agent, including intermediate reward and final reward. The final rewards is defined as a sum of structural rewards and adversarial rewards. Both are determined by the difference between the adjacency matrix corresponding to the intermediate state graph and the target graph. The intermediate rewards include step-wise validity rewards and adversarial rewards. If the transferred state can reduce the difference from the target graph, a small positive reward will be allocated, otherwise a small negative reward will be allocated. When it is determined that the action is a terminating action, both an intermediate reward and a final reward are allocated. Since our model assumes that no new stations are added during the change of the traffic network, we use a simpler method to measure the difference between the current state s_t and the final target graph state s_f by the similarity of the graph adjacency matrix. We also use the Generative Adversarial Network (GAN) framework [38] to define the adversarial rewards:

$$\underset{\theta}{min}\,\underset{\phi}{max}\, V(\pi_\theta, D_\phi) = \mathbb{E}_{x\sim p_{data}}[logD_\phi(x)] + \mathbb{E}_{x\sim\pi_\theta}[logD_\phi(1-x)], \tag{4}$$

where π_θ refers to the policy network, D_ϕ is the discriminator network, x means an input graph, p_{data} represents the underlying data distribution which defined either over final graphs (for final rewards) or intermediate graphs (for intermediate rewards). Meanwhile, the structural rewards is defined by the similarity measure. The calculation method of similarity and structural reward function is shown in Equations 5 and 6.

$$sim(s_t, s_f) = \frac{\vec{s_t} \cdot \vec{s_f}}{|\vec{s_t}| \cdot |\vec{s_f}|}, \tag{5}$$

$$r(t) = \begin{cases} \beta, & \text{if } sim(s_t, s_f) < \epsilon \\ \lambda \cdot (sim(s_t, s_f) - sim(s_{t-1}, s_f))), & \text{otherwise} \end{cases} \tag{6}$$

where s_t and s_f are the flattened vectors of adjacency matrices of the graph corresponding to the current state and the final state, λ is the reward coefficient, β is the final reward whose value is much larger than the intermediate reward, and ϵ is the threshold for judging whether the action is terminated.

After defining the generating environment of the traffic flow transfer graph, we can use the intermediate graph G^t_{trans} as the input of the GCPN, and the output obtained is the action a_t at the current time step t. GCPN mainly includes two parts: computing node embedding and action prediction. In [18], the node embedding is computed by graph convolutional neural network [39], and then node v_1, node v_2, action type and terminal type in the action quaternion is mapped by MLP layer by layer to give predictions of each component. The policy gradient algorithm uses the Proximal Policy Optimization (PPO) [40, 41] algorithm.

The trained GCPN can generate dynamic traffic graphs within a certain time range based on the current state. These generated dynamic traffic graphs are based on historical traffic flow transfer patterns and do not be affected by data defects. It should be noted that the graph convolution operation in GCPN is different from the graph convolution operation in the spatio-temporal prediction network in Section 3.5. The purpose of the graph convolution in this section is to calculate the node embedding of the traffic flow transfer graph G^t_{trans}, while the graph convolution in Section 3.5 is to capture the spatial features of the traffic network utilizing the traffic flow probability graph G^t_{prob}. It is worth emphasizing again that the role of GCPN is only to generate dynamic graphs to improve the accuracy of long-term prediction. The main framework of the long-term traffic flow prediction is still the spatio-temporal prediction network in Section 3.5.

In addition, in practical applications, if there is a temporary station cancellation in the current traffic system, we constraint the generated dynamic traffic graph so that the indegrees and outdegrees of the cancelled station are 0. But for temporary increasement of the station, the traffic flow transfer graph and traffic flow probability graph need to be rebuilt and the model need to be retrained.

3.5. *Spatio-temporal prediction network*

To implement long-term prediction of traffic flow, we have proposed the use of dynamic graph models to fully extract spatio-temporal features. In this section we will introduce the framework of our spatio-temporal prediction network. The framework of our spatio-temporal prediction network is shown in Figure 5. The length of the input sequence is determined according to the predicted duration T. For each group of traffic flow probability graphs and traffic flow features in the sequence, we extract spatial feature by Graph Convolution Network. Then based on the obtained spatial feature sequence, LSTM is used to extract temporal features, and the final output is the traffic flow features $\hat{X}^{t+T}$ after the predicted period T corresponding to the current sequence$[X^{t-n}, \cdots, X^{t-1}, X^t]$.

First, we extract spatial features through Graph Convolutional Neural Networks (GCN) [39]. Graph convolutional neural network is a very important type of graph neural network, which can process first-order neighborhood information on non-Euclidean space. The input of the GCN model requires the adjacency matrix and node features of the graph. In our problem, they correspond to the adjacency matrix A^t_{prob} of the traffic flow probability graph G^t_{prob} and the traffic flow feature X^t of the nodes, respectively. It is worth noting that the traffic flow probability graph contains two parts, the inflow probability graph G^t_{in} and the outflow probability graph G^t_{out}, so our graph convolution operation will combine the topological structure of the two graphs to learn the spatial features. We use the more common model of single-layer graph convolution operations. The specific forward propagation method is:

$$H^t_{out} = \tilde{D}_{out}^{-\frac{1}{2}} \tilde{A}^t_{out} \tilde{D}_{out}^{-\frac{1}{2}} X^t W_0, \tag{7}$$

$$H^t_{in} = \tilde{D}_{in}^{-\frac{1}{2}} \tilde{A}^t_{in} \tilde{D}_{in}^{-\frac{1}{2}} X^t W_0, \tag{8}$$

$$H^t_s = \sigma\left((H^t_{out} || H^t_{in}) W_1\right), \tag{9}$$

where $\tilde{A}^t = A^t + I$ is the self-connection added to the adjacency matrix, I is the identity matrix; $\tilde{D}$ is the degree matrix of A^t, that is, $\tilde{D}_{ii} = \sum_j \tilde{A}^t_{ij}$; X^t is the traffic flow feature at time t; "||" is the concatenation operation, which combines the convolution results of the inflow probability graph and the outflow probability graph, and then performs a linear transformation

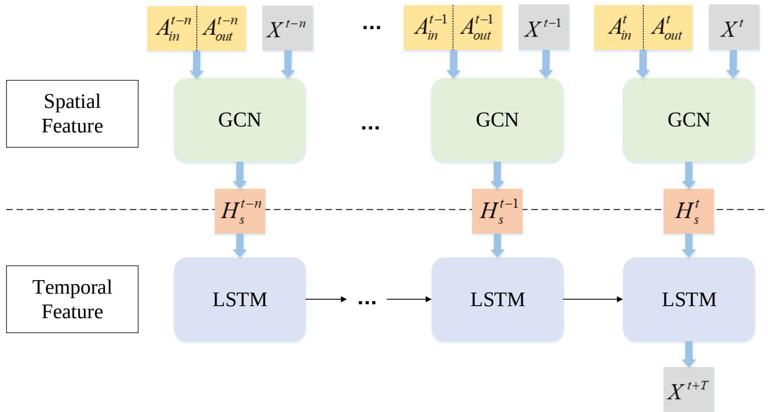


Figure 5: The framework of our spatio-temporal prediction network.

through a fully connected layer. W_0 is the parameter matrix of the graph convolution layer, and W_1 is the parameter matrix of the fully connected layer. Finally, a non-linear transformation is performed by the activation function σ to obtain the spatial feature H_s^t that aggregates the adjacent vertex features of the inflow and outflow probability graphs. The activation function can be *sigmoid* or *ReLU*, etc. but *LeakyReLU* is chosen in our model. The above-mentioned graph convolution operation can be regarded as the convolution of a two-channel adjacency matrix. This method combines the two aspects of outflow and inflow in the topological relationship to capture more comprehensive spatial features

After extracting the spatial features of the traffic network through graph convolution, we also need to extract the temporal features to achieve long-term traffic flow prediction. In simple terms, it is necessary to learn a sequence of spatial features over a regular period of time, and capture the features of spatial features that change over time. So after graph convolution we add the LSTM layer to our spatio-temporal prediction network. From another perspective, although it is able to generate dynamic graph sequences through the GCPN model in Section 3.4, the results obtained by GCPN cannot be used as the results of traffic flow prediction. The original intention of adding dynamic graph generation to traffic flow prediction is for scenarios with data defects. The purpose of generating dynamic graphs is to reduce the error of long-term prediction in the case of data defects. The generated dynamic graphs cannot be used as predicted results. In other words, even in the task of short-term prediction, the results obtained by reinforcement learning cannot compete with supervised learning. Therefore, LSTM is still very important in learn temporal feature for our proposed model.

In our spatio-temporal prediction network model, whether it is a graph that can be obtained from the original data or a graph that needs to be generated by GCPN, it is necessary to extract the spatial feature H_s^t through the graph convolution layer. Therefore, the input of the LSTM layer is the spatial feature sequence $[H_s^{t-n}, \cdots, H_s^{t-1}, H_s^t]$ within a certain period of time, and for different prediction period length T, the output is the traffic flow feature X^{t+T} at time $t + T$.

Since the traffic flow features of weekdays and weekends are generally different, when it is detected that there is a large difference between the traffic flow of weekdays and weekends in the dataset, the data of weekdays and weekends should be trained separately. In other words, there are independent GCPN, GCN and LSTM parameters for weekdays and weekends.

The loss function used in the training process is the Mean Square Error (MSE) loss function, which is used to calculate the difference between the predicted traffic flow $\hat{Y}_t$ and the target traffic flow Y_t. The goal of model training is to minimize the loss function. The calculation method is shown in the Equation (10).

$$loss(Y_t, \hat{Y}_t) = (Y_t - \hat{Y}_t)^2, Y_t, \hat{Y}_t \in \mathbb{R}^{N\times 2}. \tag{10}$$

For the entire network framework, the length of time it can predict depends on the time interval between the spatial features of the LSTM's input and output, and its prediction error will be affected by the quality of the spatial features, the time interval within the sequence, and the sequence length. In the case of data defects, blindly increasing the prediction time and the length of the prediction sequence will inevitably increase the impact of data defects. Therefore, in order to make long-term traffic flow prediction results long-term and accurate enough, both dynamic graph generation and LSTM are indispensable.

4. Experiment

4.1. Datasets

The dataset used is city bike data from New York City [1]. The data set contains cycling records of city bikes with cycling time of more than 1 minute. The key fields in each record include start time and date, end time and date, start and end station names, IDs, latitude and longitude, etc. We use data for a year of 2019, that is, data between 2019.1.1-2019.12.31, including a total of 20,551,697 cycling records and 974 stations. These stations are all start stations or end stations that have appeared in the data set. We summarize all the stations information in the dataset to establish a traffic network for city bikes, and treat the cycling of bikes as traffic flow, and test the performance of our proposed model on this traffic network. Since these data have been recorded for a long time, we assume that these cycling records in the dataset are true, complete and reliable. Therefore, in order to simulate the data defects in real-time prediction, we process part of the data by adding random noise. The specific operation is introduced in Section 4.3.

4.2. Evaluation Metrics and Baselines

In order to evaluate our model, we use Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) to measure the difference between the predicted traffic flow $\hat{Y}_t = \{\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_N\}$ and the actual traffic flow $Y_t = \{y_1, y_2, \cdots, y_N\}$, where N is the number of stations in traffic networks. RMSE is calculated as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}. \tag{11}$$

And MAE is calculated as:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i|. \tag{12}$$

Both of them can describe the difference between the predicted traffic flow and the actual traffic flow. The smaller the value, the more accurate the prediction.

We compare our proposed dynamic graph-based spatio-temporal long-term traffic flow prediction model (Dynamic graph) with the following baseline method:

ARIMA [42]: The Auto-Regressive Integrated Moving Average model is a classic time series predictive analysis method. It uses the historical time data of the variable itself to predict its changes by difference method.

SVR [43]: Support Vector Regression method. Use the sliding window overlap to set the input and output data, and train the model to obtain the relationship of historical data input and output. The RBF kernel function is used in the experiment.

LSTM [44]: Consistent with LSTM layer in Section 3.5. However, only temporal features are extracted, and spatial features are not extracted.

DCRNN [45]: Diffusion Convolutional Recurrent Neural Network. A traffic flow prediction method that captures both spatial and temporal dependencies. It uses bidirectional random walk to model spatial dependencies, and uses GRU and encoder-decoder framework to model temporal dependencies.

STGCN [9]: Spatio-Temporal Graph Convolutional Networks. A traffic prediction method that only uses convolutional structure, which combines graph convolution and temporal convolution to capture spatio-temporal correlation.

T-GCN [8]: A deep learning network based on GCN and GRU to extract the spatio-temporal features in the traffic network. We change the modeling of the road network in the original model to the modeling of the stations suitable for our dataset, and generate the undirected graph with edge weights as the actual distance of the stations as the input of GCN.

Static graph: Use the approach in Section 3.3 to model the traffic network, and learn the spatio-temporal features through GCN and LSTM. The difference from the prediction network in Section 3.5 is that the input of the graph convolution layer uses only one static graph generated after counting long-term data sequences.

4.3. Experimental Settings

We randomly add noise ϵ to simulate the situation of data defects in practical applications, that is, $\tilde{X}^t = X^t + \epsilon$. The added random noise obeys Gaussian distribution, that is, $\epsilon \sim N(\mu, \sigma^2)$. We set $\mu = 0$ and $\sigma^2 = 1$. For all the baseline methods in Section 4.2 and our dynamic graph method, the training set accounts for 60% of the total data, the validation set accounts for 20%, and the test set accounts for 20%. In the training set, 50% of the data is completely correct. This part of the data is used for both spatio-temporal network training and dynamic graph generation. The remaining 10% of the data is added with

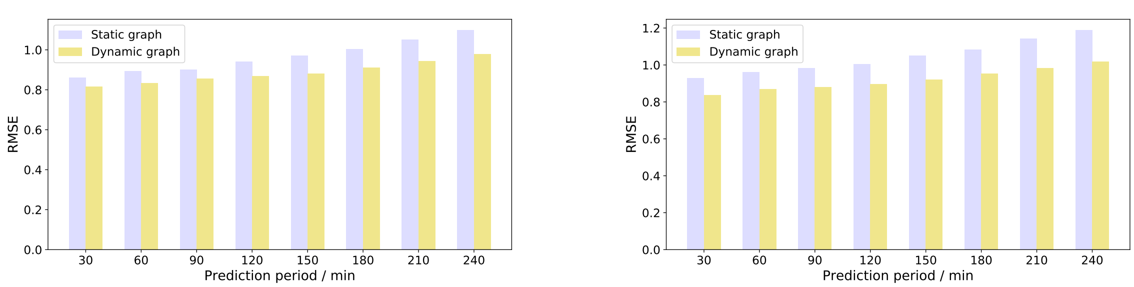
[1]NYC City Bike Data: http://www.citibikenyc.com/system-data

random noise to simulate data defects. The random noise is also added to 10% of the data in the validation set and test set. It is worth noting that since the datasets provided to all methods are uniform, the comparison between different methods is fair. The methods using static graphs utilize the correct original data to generate static graphs, while the dynamic graph method will learn from the correct original data through GCPN to generate dynamic graphs. For the dynamic graph generation task, we employ the OpenAI Gym environment and adapt it to our traffic flow graph dataset. The target graph is set to the traffic flow transfer graph after 15 minutes from the current graph. We set the final reward β to 5, the reward factor λ to 0.01, and the immediate reward range is $[-5, 5]$. We use the single-layer GCPN model to train the policy network. The learning rate of the PPO algorithm is set to 0.001. The training process uses the Adam optimizer. For long-term traffic flow prediction tasks, since more GCN layers will make the embedding of each node more similar, resulting in over smoothing [46], which is not conducive to the loss function and back propagation, we use the single-layer GCN model in our Dynamic graph methods. The number of layers of LSTM is also set to 1. The input sequence of the LSTM unit is the spatial feature in the past 6 hours, and the number of hidden units is set to 128. The batch size is set to 64. The training process utilize the Adam optimizer and the learning rate is set to 0.001. All baseline methods using LSTM follow the above settings. The time length of the traffic flow probability graph in the Static graph method is consistent with that in our Dynamic graph method. Since we detect that there is a large difference in traffic flow between weekdays and weekends in the city bike dataset, the average traffic flow on weekdays is about 3-5 times that of weekends, so we distinguish between weekdays and weekends for training and prediction for all methods. The prediction results of weekdays and weekends are evaluated together using the unified evaluation metrics.

To make fair comparisons across different methods, we evaluate the experiments using the same multi-node GPU cluster, where each node consists of a 64-core Intel Xeon CPU E5-2680 v4@2.40GHz with 512GB RAM and four NVIDIA TeslaP100 GPUs. The cluster system runs on Ubuntu 20.04 LTS with Linux kernel v.5.4.0. Our experiments is implemented using PyTorch 1.5.0 and Python 3.6.10.

Table 1: Results of long-term traffic flow prediction without data defects.

Methods	1d15min		1d30min		1d1h		1d2h	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
ARIMA	1.613	1.196	1.625	1.207	1.662	1.241	1.695	1.284
SVR	1.443	0.816	1.458	0.826	1.479	0.827	1.487	0.836
LSTM	0.962	0.496	0.989	0.503	0.993	0.538	1.041	0.579
DCRNN	0.824	0.403	0.869	0.421	0.915	0.452	0.942	0.478
STGCN	0.832	0.410	0.863	0.434	0.935	0.462	0.968	0.472
T-GCN	0.829	0.417	0.872	0.439	0.923	0.459	0.954	0.487
Static Graph	0.831	0.404	0.862	0.428	0.893	0.454	0.942	0.468
Dynamic Graph	0.801	0.398	0.815	0.401	0.834	0.418	0.868	0.436



(a) The RMSE of traffic flow prediction after 1 day.

(b) The RMSE of traffic flow prediction after 1 week.

Figure 6: Long-term traffic flow prediction RMSE of static graph and dynamic graph method under different prediction periods length without data defects.

4.4. Main Results

In this section, we evaluate the performance of our model on the New York City bike-sharing data. In order to test our model's ability to make long-term prediction, we test the prediction results of the baseline methods and our method (Dynamic graph) with and without data defects.

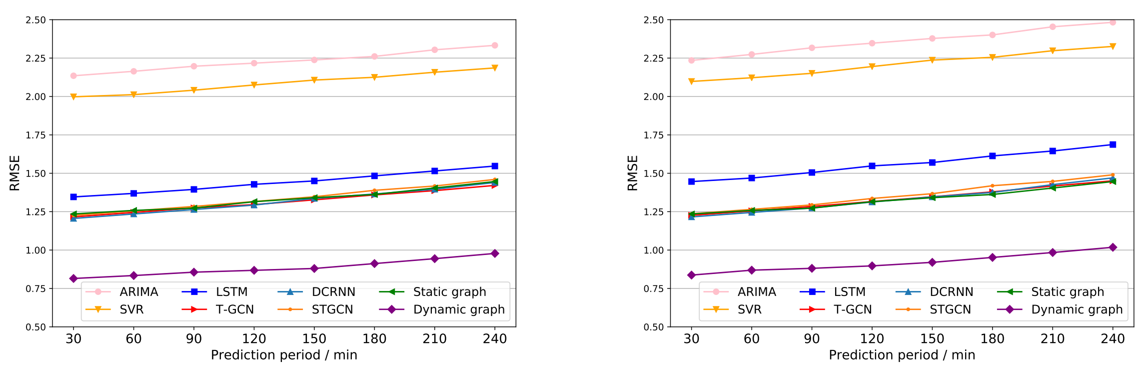
4.4.1. Comparison of other baseline methods without data defects

We first evaluate the performance of our method without data defects. Table 1 shows the result of long-term traffic flow prediction without data defects. We set the sampling interval (the length of a time step) to 15min, and give the prediction results when the prediction period is 15min, 30min, 1h, and 2h after 1 day.

The results in Table 1 shows that classic machine learning models such as ARIMA and SVR for time series prediction problems are difficult to deal with the instability of time feature sequences when facing more complex spatio-temporal prediction problems. Although the performance of LSTM is acceptable, considering only temporal features and ignoring spatial features is obviously not as comprehensive as extracting both temporal and spatial features. The DCRNN, STGCN, T-GCN comprehensively consider the spatio-temporal features, but its graph modeling only considers the geographical distance information of the stations, and does not consider the dynamic changes of the transportation network in the long-term prediction. In addition, when there is no defect in the data, the result of the Static Graph method is not better than other methods, which indicates that the transfer relationship in traffic graph modeling is not better than the distance relationship. However, the graph based on distance modeling is always static, while the transfer of traffic changes over time. Therefore, the Dynamic Graph method can capture more effective temporal and spatial dependence.

Table 2: Results of long-term traffic flow prediction with data defects.

Methods	1d15min		1d30min		1d1h		1d2h	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
ARIMA	2.121	1.448	2.135	1.454	2.164	1.467	2.217	1.491
SVR	1.986	1.218	1.998	1.225	2.012	1.239	2.075	1.258
LSTM	1.325	0.878	1.346	0.894	1.369	0.914	1.428	0.927
DCRNN	1.174	0.723	1.206	0.741	1.238	0.774	1.293	0.785
STGCN	1.213	0.762	1.226	0.787	1.254	0.806	1.318	0.834
T-GCN	1.202	0.756	1.216	0.770	1.235	0.781	1.296	0.795
Static Graph	1.217	0.775	1.235	0.784	1.258	0.797	1.315	0.813
Dynamic Graph	0.809	0.403	0.823	0.413	0.852	0.428	0.875	0.447



(a) Comparison of traffic flow prediction after 1 day.

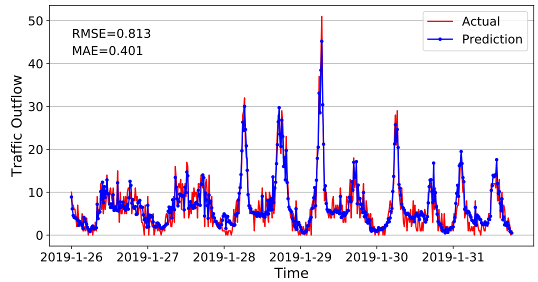
(b) Comparison of traffic flow prediction after 1 week.

Figure 7: The performance of different models in long-term predictions at different length of prediction periods in the case of data defects.

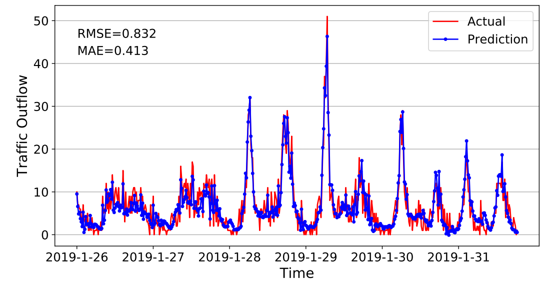
In order to more intuitively reflect that our Dynamic Graph method still has slight advantages without data defects, we use Static Graph model and Dynamic Graph model to predict the traffic flow in the future 30 minutes to 4 hours after 1 day and 1 week respectively. The results of RMSE are shown in Figure 6. It can be found that although the prediction error of both methods will inevitably increase with the prediction time, the growth rate of the Dynamic Graph method is lower than that of the Static Graph method. This phenomenon becomes more obvious when the prediction period becomes longer. The prediction result after one week has a larger error than the prediction one day later.

4.4.2. Comparison of other baseline methods with data defects

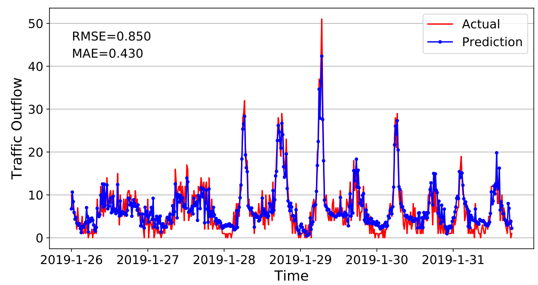
We then evaluate the performance of our method with data defects. Table 2 shows the result of long-term traffic flow prediction without data defects. We set the sampling interval (the length of a time step) to 15min, and give the prediction



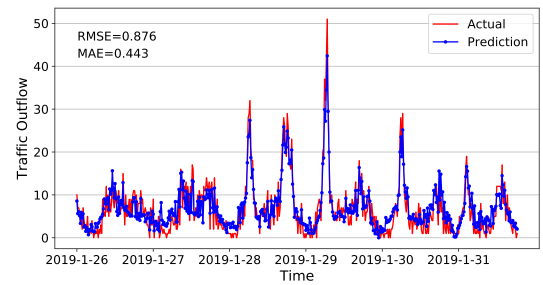
(a) Visualization of traffic outflow for prediction period of 15 minutes.



(b) Visualization of traffic outflow for prediction period of 30 minutes.



(c) Visualization of traffic outflow for prediction period of 1 hour.



(d) Visualization of traffic outflow for prediction period of 2 hours.

Figure 8: Visualization of actual and predicted traffic outflow for some high-traffic flow stations in New York City bikes systems at different prediction periods length.

results when the prediction period is 15min, 30min, 1h, and 2h after 1 day. The results in Table 2 indicates that when the data has defects, the traffic flow series input by each method has deviations, so only extracting the temporal dependence cause the error to be further amplified. In the deep learning methods that extract spatio-temporal dependencies, since the graph of DCRNN, STGCN, and T-GCN are distance-based, they do not change with the absence of traffic data, so they are less affected by spatial relationships, but still affected by data defects. Since the Static Graph method is based on traffic transfer, it is more affected by data defects. Our Dynamic Graph method can generate dynamic graphs through reinforcement learning to compensate for the deviation of the input sequence itself, so it is less affected by data defects. In order to more intuitively reflect the ability of our model to overcome data defects in long-term prediction tasks, we set the sampling interval to 15 minutes on the defective data, using baseline methods and our model to predict the traffic flow in future 30 minutes to 4 hours after 1 day and 1 week respectively. Figure 7 shows the results of long-term prediction in the case of data defects. It can be seen that the errors of the baseline method are much higher than our model. Moreover, the RMSE of other baseline methods increases significantly after the prediction period changes from one day to one week. However, the RMSE of our proposed method only slightly increases since it can automatically compensate for the impact of data defects.

Through the above analysis, we can consider that our model can effectively extract the spatio-temporal features in the traffic network, especially in terms of spatial features. Compared with considering only temporal features, our proposed modeling method of traffic flow probability graph can effectively capture spatial features and reduce errors in long-term traffic flow prediction. Secondly, our proposed method based on reinforcement learning for dynamic graph generation can effectively overcome the problems of data delay and missing in practical applications. In the case of data defects, the anti-interference ability of the dynamic graph generation method is stronger. Finally, our model has better long-term prediction capabilities. Regardless of whether the data is defective, our model can effectively reduce the error in long-term prediction.

4.5. Model Analysis

In order to further analyze our model and experimental results, we compare the actual traffic flow with the traffic flow predicted by our dynamic graph spatio-temporal network model through visualization. We select some stations (10 in total) with a large traffic in the New York City bike sharing system, and predict the data at the end of January, 2019 when the prediction period length is 15min, 30min, 1h, and 2h after 1 day, respectively. The sampling interval for the traffic flow is 15 minutes. For better visualization, we only visualize the traffic outflow. The visualization results are shown in Figure 8. The error (RMSE and MAE) between predicted traffic flow and actual traffic flow is also shown in the figure. It should be noted

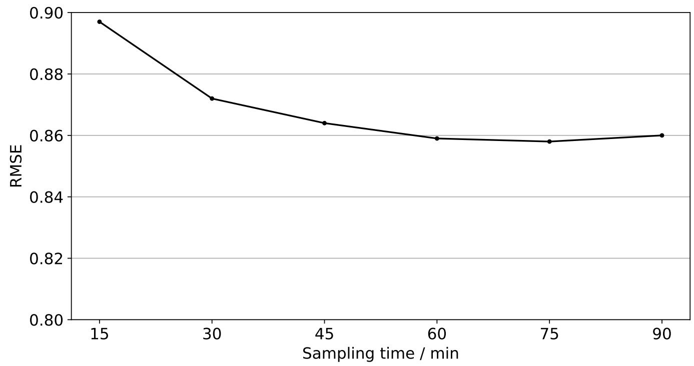


Figure 9: The relationship between traffic flow prediction results and the length of data sampling time required to generate the graph.

that the actual traffic flow is obtained from the statistics of the original data, and the predicted traffic flow is obtained from the data after adding noise.

From the figure we can get the following information: Firstly, although the traffic flow has a certain periodicity (for example, there are usually two peaks in a weekday), but its changes are not very regular. It can be found from Figure 8 that the features of traffic flow on weekends are significantly different from those on weekdays, and the peak traffic flow on each day of the weekdays is also quite different. Therefore, we can see the advantages of our dynamic graph method. Since our dynamic graph method captures the different spatial features of the traffic network at different times, it can more accurately predict the abnormal peak of traffic flow. The static graph method is difficult to capture these spatial features. Even in the absence of data defects, the dynamic graph method is better than the static graph at these peak points. This is why the RMSE of the dynamic graph method in Figure 6 is smaller. In the case of data defects, the advantages of the dynamic graph method are more obvious. Secondly, as the length of the prediction period increases exponentially, the error (RMSE and MAE) between the predicted traffic flow and the actual traffic flow gradually increases. However, there is no large deviation in the overall trend of traffic flow, and the prediction of the moments when the peaks and valleys appear is accurate. Therefore, our model can be considered to have the ability to make long-term predictions. Finally, it is not hard to find that the main error between the predicted traffic flow and the actual traffic flow occurs when the actual flow fluctuates rapidly in a short time, especially when the traffic flow is zero. The reason may be that when the traffic flow is small, a large number of zero elements appear in the adjacency matrix of the traffic flow probability graph, which affects the generation of dynamic graphs and the extraction of spatial features. However, in general, our spatio-temporal prediction network model based on dynamic graph generation is capable of long-term prediction of traffic flow when the data has defects.

In addition, obtaining a traffic flow probability graph by collecting data requires a certain data length. In order to verify that our proposed model can be applied to practical prediction tasks, we test the long-term prediction error (RMSE) of traffic flow at different data sampling time lengths. The sampling interval is 15 minutes and the prediction period is 1 hour. The results are shown in Figure 9. Since the weights of the edges in the traffic flow probability graph are all traffic flow transition probability values between $[0, 1]$, as long as the graph contains sufficient transition probability information. When the data sampling time exceeds 1 hour, the error value is basically stable. The result demonstrates that the dynamic graph model we use does not require long training data length when facing intact data. However, for dynamic graph generation tasks on defective data, the model still needs sufficient historical data as the data support for the policy network.

5. Conclusion

In this paper, we propose a method that can make long-term predictions of traffic flow even if the data is defective. For the traffic network in a specified area, we model the topology information in the traffic network through the graph structure, and establish a dynamic traffic flow probability graph model to represent the spatial features. In order to overcome the data defects that may occur in practical applications, such as the data provided by relevant institutions and organizations are not real-time and accurate, and the graph structure cannot be obtained in a short time, we propose a method of dynamic graph generation to perform reinforcement learning on the complete traffic flow transfer graph environment through GCPN. For the completed dynamic graph sequence, we extract spatial features through GCN, and then extract temporal features through LSTM, and train deep learning models to achieve long-term prediction of traffic flow.

We test our method on a bike sharing system in New York City by simulating data defects through randomly generating noise. The experimental results demonstrate that our model can outperform mainstream machine learning and deep learning methods in traffic flow prediction tasks, and has the ability to predict long-term traffic flow. However, the model still has room for improvement. In terms of dynamic graph generation, the technology for generating graphs using reinforcement learning is still immature, and the training period for policy networks for larger traffic networks will be very long. In terms of spatial feature extraction, we can further consider the addition of heterogeneous graph models [47, 48] and attention mechanisms [49, 50]. In the future, we will consider more efficient methods to combat data defects in traffic flow prediction tasks, and optimize the structure of the spatio-temporal prediction network to reduce errors in long-term prediction tasks.

Acknowledgement

This work is supported by NSFC through grants U20B2053, 62002007, and 62073012, S&T Program of Hebei through grant 20310101D. Xu Zhang was supported by Technological Innovation 2030 (2020AAA0108405, 2020AAA0108400) and NSFC U1636123. Lifang He was supported by NSF ONR N00014-18-1-2009 and Lehigh's accelerator grant S00010293.

References

[1] S. Wang, J. Cao, P. Yu, Deep learning for spatio-temporal data mining: A survey, IEEE Transactions on Knowledge and Data Engineering (2020).

[2] Z. He, C.-Y. Chow, J.-D. Zhang, Stcnn: A spatio-temporal convolutional neural network for long-term traffic prediction, in: 2019 20th IEEE International Conference on Mobile Data Management (MDM), IEEE, 2019, pp. 226–233.

[3] Y. Liu, H. Zheng, X. Feng, Z. Chen, Short-term traffic flow prediction with conv-lstm, in: 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP), IEEE, 2017, pp. 1–6.

[4] B. Du, H. Peng, S. Wang, M. Z. A. Bhuiyan, L. Wang, Q. Gong, L. Liu, J. Li, Deep irregular convolutional residual lstm for urban traffic passenger flows prediction, IEEE Transactions on Intelligent Transportation Systems (2019).

[5] J. Li, H. Peng, L. Liu, G. Xiong, B. Du, H. Ma, L. Wang, M. Z. A. Bhuiyan, Graph cnns for urban traffic passenger flows prediction, in: 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), IEEE, 2018, pp. 29–36.

[6] Z. Zhene, P. Hao, L. Lin, X. Guixi, B. Du, M. Z. A. Bhuiyan, Y. Long, D. Li, Deep convolutional mesh rnn for urban traffic passenger flows prediction, in: 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), IEEE, 2018, pp. 1305–1310.

[7] Z. Cui, K. Henrickson, R. Ke, Y. Wang, Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting, IEEE Transactions on Intelligent Transportation Systems (2019).

[8] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-gcn: A temporal graph convolutional network for traffic prediction, IEEE Transactions on Intelligent Transportation Systems (2019).

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

[10] H. Peng, H. Wang, B. Du, M. Z. A. Bhuiyan, H. Ma, J. Liu, L. Wang, Z. Yang, L. Du, S. Wang, et al., Spatial temporal incidence dynamic graph neural networks for traffic flow forecasting, Information Sciences 521 (2020) 277–290.

[11] L. Zhao, Y. Song, M. Deng, H. Li, Temporal graph convolutional network for urban traffic flow prediction method, arXiv preprint arXiv:1811.05320 (2018).

[12] M. Uzunova, N. Linkov, R. Velichkova, K. Dimitrov, Bond graph based modeling approach with application for traffic flow, in: 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), IEEE, 2019, pp. 1705–1710.

[13] X. Zhang, A. Haghani, S. Yang, Is dynamic traffic sensor network profitable for network-level real-time information prediction?, Transportation Research Part C: Emerging Technologies 102 (2019) 32–59.

[14] C. Chen, K. Li, S. G. Teo, X. Zou, K. Wang, J. Wang, Z. Zeng, Gated residual recurrent graph neural networks for traffic prediction, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 485–492.

[15] K. S. Oberoi, G. Del Mondo, Y. Dupuis, P. Vasseur, Modeling Road Traffic Takes Time, in: Proceedings of 10th International Conference on Geographic Information Science (GIScience 2018), Melbourne, Australia, 2018, pp. 52:1–52:7.

[16] C. Chen, X. Liu, T. Qiu, A. K. Sangaiah, A short-term traffic prediction model in the vehicular cyber–physical systems, Future Generation Computer Systems (2017).

[17] J. Jiang, N. Dellaert, T. Van Woensel, L. Wu, Modelling traffic flows and estimating road travel times in transportation network under dynamic disturbances, Transportation (2019) 1–30.

[18] J. You, B. Liu, Z. Ying, V. Pande, J. Leskovec, Graph convolutional policy network for goal-directed molecular graph generation, in: Advances in neural information processing systems, 2018, pp. 6410–6421.

[19] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.

[20] G. Atluri, A. Karpatne, V. Kumar, Spatio-temporal data mining: A survey of problems and methods, ACM Computing Surveys 51 (2017).

[21] J. Zhang, Y. Zheng, J. Sun, D. Qi, Flow prediction in spatio-temporal networks based on multitask deep learning, IEEE Transactions on Knowledge and Data Engineering (2019).

[22] J. Zhang, Y. Zheng, D. Qi, Deep spatio-temporal residual networks for citywide crowd flows prediction, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017, p. 1655–1661.

[23] G. Guo, T. Zhang, A residual spatio-temporal architecture for travel demand forecasting, Transportation Research Part C: Emerging Technologies 115 (2020) 102639.

[24] Z. He, L. Zheng, P. Chen, W. Guan, Mapping to cells: a simple method to extract traffic dynamics from probe vehicle data, Computer-Aided Civil and Infrastructure Engineering 32 (2017) 252–267.

[25] S. Guo, Y. Lin, S. Li, Z. Chen, H. Wan, Deep spatial-temporal 3d convolutional neural networks for traffic data forecasting, IEEE Transactions on Intelligent Transportation Systems 20 (2019) 3913–3926.

[26] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: International Conference on Learning Representations, 2018.

[27] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial-temporal graph convolutional networks for traffic flow forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 922–929.

[28] D. Chai, L. Wang, Q. Yang, Bike flow prediction with multi-graph convolutional networks, in: Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2018, pp. 397–400.

[29] T. Zhang, G. Guo, Graph attention lstm: A spatio-temperal approach for traffic flow forecasting, IEEE Intelligent Transportation Systems Magazine (2020).

[30] Y. Wang, H. Yin, H. Chen, T. Wo, J. Xu, K. Zheng, Origin-destination matrix prediction via graph convolution: a new perspective of passenger demand modeling, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 1227–1235.

[31] C. Chen, K. Li, S. G. Teo, X. Zou, K. Li, Z. Zeng, Citywide traffic flow prediction based on multiple gated spatio-temporal convolutional neural networks, ACM Transactions on Knowledge Discovery from Data (TKDD) 14 (2020) 1–23.

[32] L. Schultz, V. Sokolov, Deep reinforcement learning for dynamic urban transportation problems, arXiv preprint arXiv:1806.05310 (2018).

[33] X.-Y. Liu, Z. Ding, S. Borst, A. Walid, Deep reinforcement learning for intelligent transportation systems, arXiv preprint arXiv:1812.00979 (2018).

[34] R. Trivedi, M. Farajtabar, P. Biswal, H. Zha, Dyrep: Learning representations over dynamic graphs, in: International Conference on Learning Representations, 2019.

[35] A. Sankar, Y. Wu, L. Gou, W. Zhang, H. Yang, Dynamic graph representation learning via self-attention networks, 2019.

[36] P. Goyal, N. Kamra, X. He, Y. Liu, Dyngem: Deep embedding method for dynamic graphs, arXiv preprint arXiv:1805.11273 (2018).

[37] A. Khan, E. Tolstaya, A. Ribeiro, V. Kumar, Graph policy gradients for large scale robot control, arXiv preprint arXiv:1907.03822 (2019).

[38] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proceedings of the 27th International Conference on Neural Information Processing Systems-Volume 2, 2014, pp. 2672–2680.

[39] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: International Conference on Learning Representations, 2017.

[40] L. Yu, W. Zhang, J. Wang, Y. Yu, Seqgan: Sequence generative adversarial nets with policy gradient, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[41] S. Fujimoto, H. Hoof, D. Meger, Addressing function approximation error in actor-critic methods, in: International Conference on Machine Learning, PMLR, 2018, pp. 1587–1596.

[42] B. M. Williams, L. A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results, Journal of transportation engineering 129 (2003) 664–672.

[43] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (2004) 199–222.

[44] Y. Tian, L. Pan, Predicting short-term traffic flow by long short-term memory recurrent neural network, in: 2015 IEEE international conference on smart city/SocialCom/SustainCom (SmartCity), IEEE, 2015, pp. 153–158.

[45] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: International Conference on Learning Representations, 2018.

[46] F. Wu, T. Zhang, A. H. d. Souza Jr, C. Fifty, T. Yu, K. Q. Weinberger, Simplifying graph convolutional networks, Proceedings of Machine Learning Research (2019).

[47] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, Y. Liu, Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, 2019, pp. 3656–3663.

[48] P. He, G. Jiang, S.-K. Lam, Y. Sun, Learning heterogeneous traffic patterns for travel time prediction of bus journeys, Information Sciences 512 (2020) 1394–1406.

[49] W. Cheng, Y. Shen, Y. Zhu, L. Huang, A neural attention model for urban air quality inference: Learning the weights of monitoring stations, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[50] Y. Zhou, J. Li, H. Chen, Y. Wu, J. Wu, L. Chen, A spatiotemporal hierarchical attention mechanism-based model for multi-step station-level crowd flow prediction, Information Sciences 544 (2021) 308–324.