Dynamic Transit Flow Graph Prediction in Spatial-Temporal Network

Liying Jiang[1], Yongxuan Lai[1(✉)], Quan Chen[2], Wenhua Zeng[1], Fan Yang[3], Fan Yi[4], and Qisheng Liao[5]

[1] School of Informatics/Shenzhen Research Institute, Xiamen University, Shenzhen, China
jiangliying@stu.xmu.edu.cn, {laiyx,whzeng}@xmu.edu.cn
[2] Department of Computer Science, Xiamen University Malaysia, Sepang, Malaysia
[3] Department of Automation, Xiamen University, Xiamen, China
yang@xmu.edu.cn
[4] School of Mathematics and Statistics, Key Laboratory of Complex Systems and Intelligent Computing, Qiannan Normal University for Nationalities, Dunyun, China
[5] New York University, New York, USA
q.liao@nyu.edu

Abstract. Traffic flow prediction is of great importance for traffic management. However, most existing researches only focus on region flow or road segment flow (vertex value) prediction, and the transit flow (edge weight) prediction is largely untouched. Compared to region flow and road segment flow prediction, transit flow prediction is more challenging in that 1) the transit flow between pairs of regions has complex spatial-temporal dependencies, and 2) it has larger changes over time due to the large number of region pairs. To address these issues, in this paper we define the transit flow as edges in directed graphs and formulate the transit flow prediction problem as a dynamic weighted link prediction problem. We propose a deep learning based method called Spatial-Temporal Network (STN) to make an accurate prediction of the transit flow. The STN model combines graph convolutional network (GCN) and long short-term memory (LSTM) to capture the dynamic spatial-temporal correlations. To capture the static topological structure, the neighborhood relation graph is adopted as an auxiliary graph to improve the prediction accuracy, and a two-stage-skip strategy is adopted to allow edge features reused which makes the STN focus more on the edge values compared to simple GCN modeling. We conduct the proposed STN model and verify its effectiveness in transit flow prediction on two real-world taxi datasets. Experiments demonstrate that our model reduces the prediction RMSE error by approximately 15.88%–52.48% on real-world datasets compared to state-of-the-art methods.

This work was supported in part by the Natural Science Foundation of Guangdong under Grant 2021A1515011578, Natural Science Foundation of China under Grant 61672441 and Grant 61673324, Natural Science Foundation of Fujian under Grant 2018J01097, Shenzhen Basic Research Program under Grant JCYJ20170818141325209 and Grant JCYJ20190809161603551.

W. Zhang et al. (Eds.): WISE 2021, LNCS 13080, pp. 603–618, 2021.
https://doi.org/10.1007/978-3-030-90888-1_46

Keywords: Transit flow prediction · Spatio-temporal network · LSTM · GCN

1 Introduction

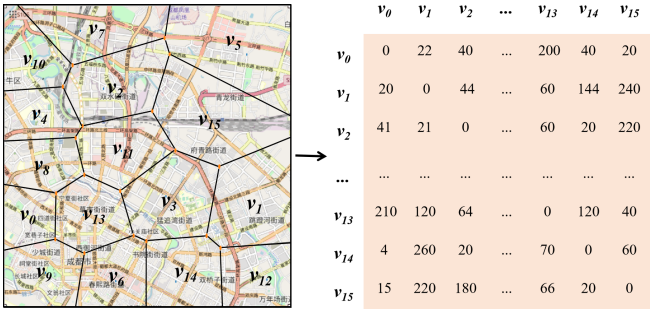


Fig. 1. Example of a transit flow graph in Chengdu City, China. The city map is divided into irregular regions, each region can be described as a node and the transit flow between a pair of regions can be described as an edge. The transit flow graph is represented by a square matrix, where each element of the matrix indicates the transit flow between a pair of regions.

Intelligent Transport System (ITS) plays an important role in improving our mode of traveling and enhancing the safety of transportation, among which traffic flow prediction is of great importance for traffic management. It is critical for traffic administration to analyze the traffic condition and propose some preventive measures in advance. Predicting traffic flow is also helpful for ride-sourcing systems to address supply-demand imbalance across space and time [12]. For example, if a system can predict the number of passengers who will leave region A for region B, it can reallocate idle vehicles to region A accordingly so that there are enough vehicles for the rides, which also shortens the waiting time.

Most of the existing researches on traffic flow prediction only focus on the traffic change of flow of a region or a road segment (vertex value). But they do not address the problem of change of flow between pairs of regions or road segments (edge weight). As illustrated in Fig. 1, the city map of Chengdu City, China is divided into 16 non-overlap regions, and each region can be described as a vertex and the transit flow between a pair of regions can be described as an edge. The transit flow graph is represented by a square matrix, where each element of the matrix indicates the traffic flow between a pair of regions. For example, $(v_1, v_2) = 44$ indicates that there are 44 taxis (or passengers) leaving region 1 for region 2. So the prediction of the traffic flow between pairs of regions is denoted as the *transit flow prediction* problem, which is more difficult than the region flow prediction problem. The major challenges lie in three folds: 1)

the transit flow has more complex spatial-temporal dependencies in nature; 2) the transit flow between regions has larger changes over time than the region flow; 3) the transit flow data is usually very sparse. The transit flow between far-away regions is usually close to or equal to 0.

To tackle the aforementioned challenges, we formulate the transit flow prediction problem as a dynamic weighted link prediction problem [14], where the transit flow among all regions at a certain timeslot can be described as a directed graph. We aim to make an accurate prediction for the transit flow graph at the next timeslot when given the transit flow graphs of previous timeslots. In this paper, we propose a deep-learning approach called STN (Spatial-Temporal Network) to make an accurate prediction of the transit traffic flow. It integrates the GCN [13] and LSTM [10] to capture the dynamic spatial-temporal characteristics of the traffic flow in the city, and adopts the neighborhood relation graph as an auxiliary graph to enhance the model of transit flow prediction. The main contributions of this paper are summarized as follows:

1. We partition a city into irregular non-overlap regions based on the K-Means clustering algorithm and represent the transit flow as directed graphs, where nodes represent regions and edges represent transit flow of pairs of regions. Then we formulate the transit flow graph prediction problem as a dynamic weighted link prediction problem.
2. We propose a Spatial-Temporal Network (STN) model to predict the transit traffic flow in the graph. The model combines GCN and LSTM to model the dynamic spatial dependencies and temporal dependencies respectively. We use the neighborhood relation graph as an auxiliary graph to capture the static spatial dependencies and use a two-stage-skip strategy to allow edge feature reusing.
3. We evaluate our STN model on two real-world taxi datasets. The results show that our method reduces the prediction RMSE error by approximately 15.88%–52.48% on TaxiXM and TaxiCD[1] compared to state-of-the-art methods, which demonstrates the effectiveness of our model in the transit flow graph prediction.

The remainder of this paper is organized as follows. Section 2 describes the related works. Section 3 defines several key concepts and introduces our problem. Section 4 shows the details of our method. Section 5 introduces the datasets and experimental settings. Section 6 presents the performance of the proposed model and compares it with other methods in two real-world datasets. Section 7 concludes the paper and outlines some future works.

2 Related Work

Traffic flow prediction has received considerable attention in recent years. The existing traffic flow prediction methods can be divided into two categories: 1)

[1] https://gaia.didichuxing.com.

traditional methods; 2) deep learning based methods. Traditional methods consist of statistical based methods and machine learning methods. Statistical based methods, such as ARIMA (Autoregressive Integrated Moving Average model) [1], can only work on steady-state traffic conditions. Furthermore, such methods require comprehensive prior knowledge which is hard for most researchers. Compare with statistical based methods, machine learning methods, such as SVM (Support Vector Machine) [22], and Random Forest [7], have a stronger ability to model the complex traffic data. But these methods still cannot model the spatial correlations and temporal correlations at the same time.

In recent years, with the rapid development of deep learning, deep neural network models have been used to predict the traffic flow because of its strong ability to capture the dynamic characteristics of traffic data [9,23]. Deep learning based methods usually consist of several components to capture spatial dependencies and temporal dependencies. Researchers usually apply RNN and its variants (e.g., LSTM, GRU) [8,15], 1D CNN [17] to model the temporal correlations, use CNN to model the spatial correlations in euclidean space [18] and use GCN to capture the spatial correlations in non-euclidean space [23]. For example, Ma et al. [15] proposed an LSTM based model to predict traffic speed/flow in road segments and it showed more superior performance in capturing temporal dependencies than traditional methods. Zhang et al. [19] proposed a CNN based model DeepST to predict the region flow. It was the first time to partition a city into $I \times J$ regular regions so that CNN can be applied to model the spatial correlations. Zhao et al. [21] proposed the T-GCN model which combined GCN and GRU to capture the spatial correlations and temporal correlations of the traffic data on the road network simultaneously. The result showed that the combined model T-GCN had superiority in traffic forecasting than single GCN and single GRU.

The researches mentioned above mainly focus on region or road segment flow (vertex value) prediction. They do not address the more challenging transit flow (edge weight) prediction problem. Region flow and road segment flow prediction address the changes of node features (i.e., the total in-flow and out-flow of a region or a road segment) and don't concern about the changes between nodes (i.e., a flow leave a region for another region). To tackle this problem, Zhang et al. [20] first divided the city into regular grids then proposed a fully connected and CNN based method to predict region flow (edge vertex) and in/out flow (edge weight) of regions simultaneously. It used fully convolutional and external factors to capture temporal correlations and use CNN to capture spatial correlations. Same as it [20], in this paper, we focus on in flow predictions (our model could be applied to predictions of out flow). Different from it, first we divided the city into non-regular grids based on K-Means. Then we used LSTM and GCN to capture temporal-temporal correlations. Compared to CNN, GCN are more suitable for non-euclidean spatial correlations modeling.

For traffic flow prediction researches, the existing GCN based methods [9,13] usually assumed the relations among regions to be static and usually used fixed matrices to represent them. In our problem, the transit flow between regions is

dynamic and we aim to predict it in future timeslots. In region flow prediction problem, the input of model is dynamic node feature and several static adjacency matrix. But in transit flow prediction problem, the input of model is dynamic transit matrix and dynamic node feature. The transit flow graph prediction problem can be formulated as a dynamic weighted link prediction problem. But different from the researches on conventional link prediction, which only focus on predicting the existence of links, in this paper we also predict the weights of links. Furthermore, conventional link prediction methods usually focus on one relation between nodes in the modeling phase and predicting phase. But in our approach, we adopt the neighborhood relation graph as an auxiliary graph for the transit flow prediction, which takes advantage of static and dynamic relationships among nodes.

3 Preliminaries

Table 1. Description of notations

Symbol	Description
v_i	i-th node, i.e., i-th region
t	t-th timeslot
$e_t(v_i, v_j)$	The relation of node v_i and v_j at t-th timeslot,i.e.,the total transit flow from region v_i to region v_j during t-th time interval
V	Node set
E_t	The relations among all nodes at t-th timeslot
$G_t = (V, E_t)$	Transit flow graph at t-th timeslot
t_j	Timestamp of j-th trajectory point
p_{j,t_j}	j-th trajectory point
$x_{i,t}$	i-th node feature at t-th timeslot, i.e., the total region flow of region v_i at t-th timeslot
X_t	Node features of all nodes at t-th timeslot
A	The adjacency matrix
$G_{t':t}$	Transit flow graph from t'-th timeslot to t-th timeslot
$X_{t':t}$	Node features of all nodes from t'-th timeslot to t-th timeslot

Some traffic flow prediction researches divided cities into various regular regions [18,19]. These partitions cannot work well under the complex administrative and functional properties of cities. There are also a few researches divided cities into irregular regions according to the road networks [11]. However, it is hard to divide cities well due to the complexity of road networks. In this paper, we divide cities into irregular regions based on K-Means [16], a simple clustering algorithm. First,

we mine the origin-destination pairs from taxi trajectories. Second, we use K-Means to get the cluster centroids from origin-destination pairs. Finally, we use Voronoi tessellation to define the Voronoi cells based on the K-Means centroids. As Fig. 1 shows, Voronoi tessellation divides the city into non-overlap irregular regions. Compared to regular partition and road network based partition, K-Means based method has two advantages; 1) origin-destination pairs reflect the characteristics of the resident trip so it can partition a city well; 2) we can obtain origin-destination data from taxi datasets and no additional data is needed.

In the rest of this section, we first define several key concepts then formulate the research problem. Table 1 lists the notations used in this paper.

3.1 Transit Flow Graph

We divide a day into several uniform intervals (e.g., 1 h). Then we represent the transit flow at t-th timeslot as a weighted graph $G_t = \{V, E_t\}$, whose nodes are regions and edges are transit flow among regions. $v_i \in V$ denotes the i-th region, and $e_t(v_i, v_j)$ denotes a traffic flow from region v_i to region v_j during t-th time interval. The example of a transit flow graph is shown in Fig. 1. The time-order transit graphs can be described as:

$$G = \{G_1, G_2, ..., G_t\} \tag{1}$$

where G_t is the transit flow graph of t-th timeslot.

3.2 Node Flow

We define trajectory point as a historical GPS point. Each GPS point p_j contains: the region number v_i and timestamp t_k. A region can be described as a node, for a node v_i, the node flow during the time interval t is defined as

$$x_{i,t} = \{p_{j,t_k} \in v_i \wedge t_k \in t\} \tag{2}$$

where $p_{j,t_k} \in v_i$ means the trajectory point p_{j,t_k} lies within the region v_i and $t_k \in t$ means the timestamp t_k is in the time interval t. We use X_t to denote the node flow of all nodes at t-th timeslot. Then we use X_t as the dynamic node features during the GCN stage.

3.3 Neighborhood Relation Graph

Transit flow between adjacent regions is likely to be larger than that between non-adjacent regions. So we use the neighborhood relation graph to enhance the relations of adjacent regions. We define an adjacent matrix to indicate whether two regions are adjacent.

$$A = \begin{cases} 1 & \text{region } v_i \text{ and region } v_j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

For example, as Fig. 1 shows, $A_{v_0,v_8} = 1$, $A_{v_0,v_4} = 0$.

3.4 Problem Statement

Transit flow graph prediction problem aims to learn a function f that is able to forecast the next timeslot transit flow graph while given t' historical transit flow graphs, node features and a neighborhood relation graph A, it can be formulated as:

$$[G_{t-t'+1:t}, X_{t-t'+1:t}, A] \xrightarrow{f} [G_{t+1}] \tag{4}$$

The transit flow between regions changes over time but the neighborhood relation between regions is always constant. Therefore, we use neighborhood relation graph A as an auxiliary graph to capture the static spatial dependencies of regions.

4 Methodology

4.1 Background Technologies

Graph Convolutional Networks (GCN). The traditional convolutional neural network (CNN) can obtain local spatial features, but it can only be used in euclidean space and is not applicable for general graphs. To address this issue, Bruna et al. [4] proposed graph convolutional networks (GCN) which redefine convolution operators to capture the spatial features for non-euclidean data. GCN is defined over a graph $G = (V, A)$, where V is the set of vertices and $A \in R^{|V|\times|V|}$ is the adjacency matrix whose entries represent the relation between vertices. A 1-layer GCN operation is defined as:

$$f(X, A) = \sigma(\widehat{A}XW) \tag{5}$$

where X represents the node feature, $\widehat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ denotes the graph Laplacian matrix, I is an identity matrix, $\tilde{A} = A + I_N$ is a matrix with self-connection structure, D is the degree matrix, W represents the learnable weights, $\sigma(\cdot)$ represents the activation function.

Long Short-Term Memory (LSTM). LSTM is a variant of recurrent neural network (RNN), which is RNN with learned gating mechanisms. LSTM has 3 more gates (Input, Forget and Output) than simple RNN, which mitigates the vanishing gradient problem and allow the model to learn longer-term dependencies. For the input x_t, LSTM follows these manners:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{6}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{7}$$

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{8}$$

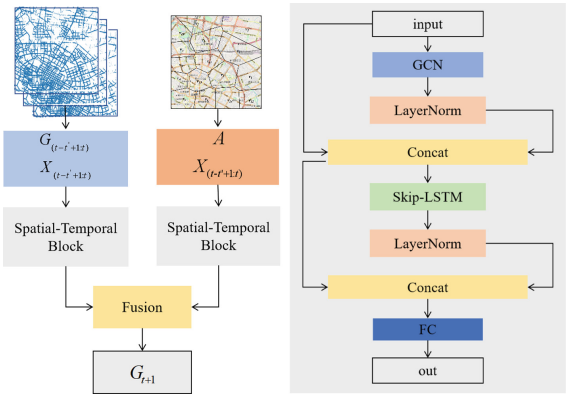
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{9}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{10}$$

$$h_t = o_t * tanh(C_t) \tag{11}$$

4.2 Spatial-Temporal Network

Figure 2(a) illustrates the framework of our proposed Spatial-Temporal Network (STN), which has two Spatial-Temporal Blocks (ST Block) and a fusion block. In ST block, we use a two-stage-skip strategy to improve the prediction accuracy. The two ST blocks have the same structure, the left one is used to get dynamic spatial dependencies from the transit flow graph (we called dynamic ST block) and the right one is used to get the static spatial dependencies from the neighborhood relation graph (we called static ST block). In dynamic ST block, we first convert the trajectories data along time into transit flow graphs $G = \{G_1, G_2, ..., G_t\}$, which is a time-ordered sequence of graphs. And then the transit flow graphs G and node features X are fed into the ST block. In static ST block, we first get the adjacent matrix A from the regions' neighborhood relation. Then the adjacent matrix A and node features X are fed into the ST block. The static ST block is an auxiliary part to predict the transit flow graph. Finally, we use a matrix fusion block to fuse the representations get from these two ST blocks to get the future transit flow graph G_{t+1}. In the rest of this section, we will elaborate these two components of STN in detail.



(a) STN (b) ST Block

Fig. 2. The architecture of proposed model Spatial-Temporal Network (STN) and Spatial-Temporal Block (ST Block)

Spatial-Temporal Block (ST Block). The structure of ST block is illustrated in Fig. 2(b). This component mainly consists of two parts. The first part is GCN, and the second part is Skip-LSTM (Fig. 3) [3]. Assuming that the input transit flow graph is G and the node feature is X. First, we utilize GCN to explore the topology characteristics of each graph and use LayerNorm [2] to normalize the representation of each node. Then we get a new representation G'. We concatenate G and G', called first-stage-skip, and pass it to a Skip-LSTM network to model the dynamic evolution of graphs along time. Figure 3 illustrates

the structure of Skip-LSTM. Different from the conventional LSTM network, we concatenate the input x_t and hidden layer's output h_t of an LSTM cell together as the final hidden layer's output, then pass it to the next LSTM cell. We also use LayerNorm to normalize the output of Skip-LSTM. Assuming that the output of Skip-LSTM is G'_T, we concatenate G' and G'_T, called second-stage-skip, and pass it to a fully connected layer to create a final representation G'_{ST}.

The propagation rule of the Spatial-Temporal block can be summarized as follows:

$$G' = LayerNorm(GCN(G, X)) \tag{12}$$

$$G'_T = LayerNorm(Skip - LSTM(G||G')) \tag{13}$$

$$G'_{ST} = FC(G'||G'_T) \tag{14}$$

where $||$ represents the concatenation operation.

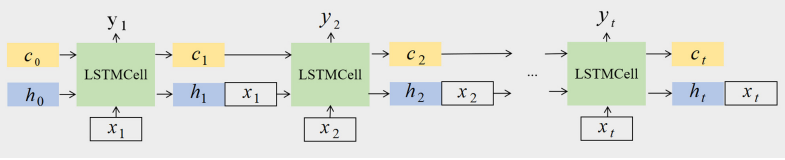


Fig. 3. The architecture of Skip-LSTM

Fusion Block. As Sect. 3 mentioned, the transit flow between adjacent regions is likely to be larger than that in non-adjacent regions. The spatial properties of transit flow are affected by the dynamic topological structure and static topological structure by varying degrees. Inspired by this, we use a parametric-matrix-based fusion method to fuse the two ST blocks. Assuming that the output of the dynamic ST block is G^d_{t+1} and the output of the static ST block is G^s_{t+1}, the fusion method can be defined as:

$$G_{t+1} = W_d \circ G^d_{t+1} + W_s \circ G^s_{t+1} \tag{15}$$

where $\circ$ is Hadamard product, W_d and W_s are learnable parameters that adjust the degrees affected by the dynamic and static spatial correlations respectively.

5 Experimental Settings

5.1 Datasets

To evaluate the effectiveness of our model, we carried out comparative experiments on two real-world taxi datasets as shown in Table 2, detailed as follows:

TaxiXM. The trajectory data is taxi GPS data of Xiamen city, China from 1st Jul. 2014 to 31st Jul. 2014. We select Xiamen island as the study area, with an area of approximately 132.5 km^2.

TaxiCD. The trajectory data is taxi GPS data of Chengdu City, China from 1st Oct. 2016 to 30th Nov. 2016. We select an area that lies between 30.65 °E to 30.72 °E latitude and 104.04 °N to 104.12 °N longitude as the study area, with an area of approximately 65 km^2.

In the experiments, we partitioned the Xiamen city into 32 regions and Chengdu City into 16 regions, each region with an average area of approximately 4 km^2, and divided a day into 72 timeslots, each slot is 20 min. We used Min-Max normalization to normalize the input data. We used 80% of the data for training, 20% for testing.

Table 2. Details of datasets

Dataset	TaxiXM	TaxiCD
Data type	Taxi GPS	Taxi GPS
Location	Xiamen	Chengdu
Area	132.5 km^2	65 km^2
Time Span	7/1/2014–7/31/2014	10/1/2016–11/30/2016
Time interval	20 min	20 min
Total time slot	2232	4392
Region number	32	16

5.2 Hyperparameters

Recalling that the task is to learn a function $f : [G_{(t-t'+1:t)}, X_{(t-t'+1:t)}, A] \rightarrow [G_{t+1}]$, we aim at forecasting the next timeslot transit flow graph given previous t' transit flow graphs. In our experiment, we set $t' = 6$ (we set $t' = 3$ to 12 then select the best result, in Sect. 6, we show the impact of input sequence length). We implemented the STN model based on PyTorch 1.5.1, a widely used Deep Learning Python library. In our model, we set graph convolution kernels as 64 and LSTM units as 256. During the training phase, we set batch size as 64 and the learning rate as 0.001.

5.3 Metrics

We evaluate the performance of our model by three widely used metrics, i.e., Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE). They are defined as follows:

$$RMSE = \sqrt{\frac{1}{n} * \sum_{i=1}^{n} (G_{t+1}^{i} - \tilde{G}_{t+1}^{i})/(|V| * |V|)} \tag{16}$$

$$MAE = \frac{1}{n} * |\sum_{i=1}^{n} (G_{t+1}^{i} - \tilde{G}_{t+1}^{i})/(|V| * |V|)| \tag{17}$$

$$MAPE = \frac{100\%}{n} * |\sum_{i=1}^{n} \frac{(G_{t+1}^{i} - \tilde{G}_{t+1}^{i})}{G_{t+1}^{i}} / (|V| * |V|)| \tag{18}$$

where G_{t+1} is the actual transit flow graph, $\tilde{G}_{t+1}^{i}$ is the predicted transit flow graph and $|V|$ is the total region number.

5.4 Compared Algorithms

We compare our model with the following 4 methods:

HA. Historical Average model uses the average of historical values in corresponding timeslots.

XGBoost [6]**.** Xtreme Gradient Boosting is a powerful ensemble method. In our experiment, we predicted the value of each edge in the transit flow graph separately. For every edge, we predict it according to its previous t' values.

E-lstm-d [5]**.** E-lstm-d is an LSTM based encoder-decoder model for conventional dynamic link prediction. Our research problem needs to predict the weights of links, so we replace the activation function Sigmoid in the model with ReLU which is more suitable for the regression problem.

GCN-GAN [14]**.** GCN-GAN is a non-linear model for weighted dynamic link prediction. GCN-GAN model combines GCN, LSTM and GAN and shows good performance in weighted temporal link prediction task.

6 Experimental Results

6.1 Performance Comparison

Table 3. Performance comparison of different approaches for transit flow graph prediction on TaxiXM and TaxiCD

Method	TaxiXM			TaxiCD		
	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)
HA	10.93	5.57	0.36	13.32	6.34	0.39
XGBoost	7.54	4.21	0.40	8.86	4.66	0.41
E-lstm-d	10.72	6.05	0.63	11.37	5.83	0.55
GCN-GAN	6.99	4.03	0.37	7.56	4.21	0.35
STN(ours)	**5.88**	**3.35**	**0.27**	**6.33**	**3.52**	**0.25**

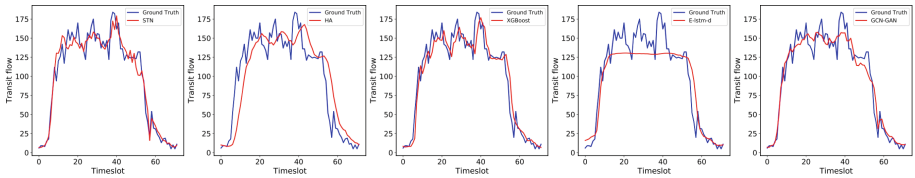
Table 3 shows the STN model and other baseline methods for transit flow graph prediction 20 min ahead on TaxiXM and TaxiCD. It can be seen that the STN model obtains the best prediction performance under all metrics. From Fig. 4, we can see that statistical based methods HA and machine learning method XGBoost have time lags and LSTM-based method E-lstm-d performs badly in

some timeslots. Therefore, they perform worse than GCN-based methods GCN-GAN and STN. On TaxiXM, the RMSE error is reduced approximately 15.88%–46.20%, the MAE error is reduced approximately 16.87%–44.63% and the MAPE error is reduced approximately 25.00%–57.14% compared to other methods. On TaxiCD datasets, the RMSE error is reduced approximately 16.27%–52.48%, the MAE error is reduced approximately 16.39%–44.48% and the MAPE error is reduced approximately 28.57%–54.54% compared to other methods. The results prove the effectiveness of the STN model in transit flow graph prediction.

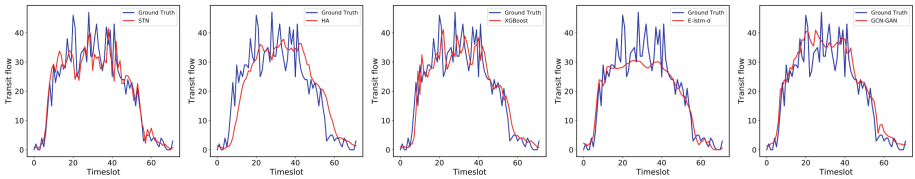
To explore the performance of these methods in adjacent regions and non-adjacent regions, we select a pair of adjacent regions and a pair of non-adjacent regions on Chengdu as a study case. Figure 4(a) is transit flow prediction between adjacent regions and Fig. 4(b) is transit flow prediction between non-adjacent regions. From these pictures, we observe that: 1) Statistical based method HA and machine learning method XGBoost have time lags in both situations, which leads to worse prediction precision than deep learning based methods GCN-GAN and STN. 2) LSTM based method E-lstm-d performs badly in modeling the time series with frequent changes and no fixed patterns (e.g., timeslot 10 to 50). Thus, in our experiment, E-lstm-d shows worse than XGBoost. 3) GCN based methods GCN-GAN and STN perform well in both adjacent regions and non-adjacent regions. And STN performs better during timeslot 10 to 50 while transit flow changes frequently. Table 4 shows the predicted result of GCN-GAN and STN in all adjacent regions and non-adjacent regions. Take GCN-GAN as a baseline, STN has a higher improvement in adjacent regions than in non-adjacent regions. For example, the MAPE error is reduced 10.52% in adjacent regions and reduced 3.44% in non-adjacent regions on TaxiCD. The reason may be that the neighborhood relation graph enables the model to learn the spatial dependencies from both dynamic topological and static topological and strengthen the relations of adjacent regions.

Table 4. Performance comparison of GCN-GAN and STN in adjacent regions and non-adjacent regions on TaxiXM and TaxiCD

		TaxiXM			TaxiCD		
		RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE
Adjacent regions	GCN-GAN	13.40	9.42	0.18	13.04	9.01	0.19
	STN	11.88	8.34	0.16	12.03	8.38	0.17
		↓11.34%	↓11.46%	↓11.11%	↓7.74%	↓6.99%	↓10.52%
Non-adjacent regions	GCN-GAN	4.81	2.74	0.31	4.24	2.32	0.29
	STN	4.32	2.53	0.30	4.00	2.23	0.28
		↓10.18%	↓7.66%	↓3.22%	↓5.66%	↓3.87%	↓3.44%



(a) transit flow from region 1 to region 12 (between adjacent regions)



(b) transit flow from region 5 to region 12 (between non-adjacent regions)

Fig. 4. The results of transit flow prediction 20 min ahead on Chengdu City.

6.2 Effect of Input Sequence Length

In our method, we hope to mine the pattern and development trend of the transit flow from the input sequence. Hence, if the input sequence length is too short, the prediction results may not be satisfactory, but if the input sequence length is too long, the training data will contain some noise information which will mislead experiment results.

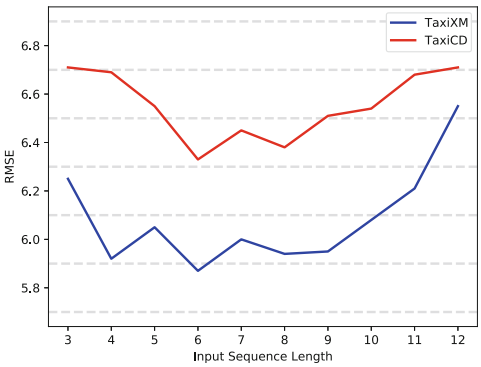


Fig. 5. Results of different lengths of input sequence on TaxiXM and TaxiCD.

To test how long input sequence length is needed to achieve a good performance, we vary the length of input sequence and see how prediction error changes. Figure 5 shows the results in TaxiXM and TaxiCD. In TaxiCD dataset, we can find that if the length of input sequence is shorter than 4, our model does not perform very well. By increasing the length of input sequence beyond

10, the model also has bad performance. With these results, we suggest that for robust prediction accuracy, the training data is desired to be last for least one and a half hours and no longer than three hours.

6.3 Effect of Each Component

To investigate the effect of each component in our model, we evaluate two variants by removing the auxiliary graph and the skip connection strategy separately, named STN-NA and STN-NS. Table 5 demonstrates the effect of different components on the final experiment performance. We find that the skip connection strategy affects the prediction results significantly. The RMSE error is reduced by approximately 1.85%–5.52% on TaxiXM and TaxiCD. This may because the skip connection strategy allows feature reusing and ensures the spatial features learned by the GCN layer are not fewer than the original graph. Therefore, GCN with skip connection strategy focuses more on edge values than simple GCN. The auxiliary graph also improves the prediction result. The RMSE error is reduced approximately 0.34%–1.40%, the MAPE error is reduced approximately 3.57%–7.40% and the MAE error is reduced approximately 1.18%–2.49%. These results confirm the effectiveness of the skip connection strategy and auxiliary graph.

Table 5. Effect of different components on TaxiXM and TaxiCD

	TaxiXM			TaxiCD		
	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)
STN-NA	5.90	3.39	0.28	6.42	3.61	0.27
STN-NS	6.18	3.47	0.28	6.70	3.70	0.26
STN	**5.88**	**3.35**	**0.27**	**6.33**	**3.52**	**0.25**

Table 6. Effect of city partition methods on TaxiXM and TaxiCD

	TaxiXM			TaxiCD		
	RMSE	MAE	MAPE(%)	RMSE	MAE	MAPE(%)
Regular	5.97	3.41	0.27	6.82	3.93	0.26
K-Means	5.88	3.35	0.27	6.33	3.52	0.25

6.4 Effect of City Partition Methods

To explore the effect of different city partition methods in transit flow predicting. We also divided the city into non-overlap regular regions, the number of which is equals to irregular regions. Then we predict the transit flow between these regular regions. Table 6 show the result of the effect on different city partition

methods. We can see that the model performs better in K-Means based partition regions both in TaxiXM and TaxiCD datasets. The result improves that origin-destination pairs reflect the characteristics of the resident trip, so it can partition a city well.

7 Conclusion

In this paper, we formulate the transit flow prediction problem as a dynamic weighted link prediction problem and propose a spatial-temporal network called STN to predict the transit traffic flow. Our model combines the GCN and Skip-LSTM, where GCN is to model the spatial correlations and Skip-LSTM to model the temporal correlations. The neighborhood relation graph is also adopted as an auxiliary graph to highlight the correlations between adjacent regions and a two-stage-skip strategy is used to reuse the edge features. The experiment results show the effectiveness of these two components. Compared with the HA, XGBoost, E-lstm-d and GCN-GAN models, our STN model achieves the best prediction results under different metrics on two real-world datasets.

Actually, the traffic flow is affected by many external factors, there are still many issues to investigate in future work. For example, we would like to involve more knowledge or data as auxiliary graphs, e.g. functional similarity graph and interaction graph of regions that could be extracted from historical taxi transition records, and embed them into the proposed framework to improve the prediction accuracy.

References

1. Ahmed, M.S., Cook, A.R.: Analysis of freeway traffic time-series data by using Box-Jenkins techniques, vol. 722 (1979)
2. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
3. Bonner, S., et al.: Temporal neighbourhood aggregation: Predicting future links in temporal graphs via recurrent variational graph convolutions. In: 2019 IEEE International Conference on Big Data (Big Data), pp. 5336–5345. IEEE (2019)
4. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203 (2013)
5. Chen, J., et al.: E-LSTM-D: a deep learning framework for dynamic network link prediction. IEEE Trans. Syst. Man Cybern. Syst. **51**(6), 3699–3712 (2019)
6. Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016)
7. Chen, Z., Ling, X., Feng, X., Zheng, H., Xu, Y.: Short-term traffic state prediction approach based on FCM and random forest. J. Electron. Inf. Technol. **40**(8), 1879–1886 (2018)
8. Fu, R., Zhang, Z., Li, L.: Using LSTM and GRU neural network methods for traffic flow prediction. In: 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 324–328. IEEE (2016)

9. Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929 (2019)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
11. Hu, J., Guo, C., Yang, B., Jensen, C.S., Chen, L.: Recurrent multi-graph neural networks for travel cost prediction. arXiv preprint arXiv:1811.05157 (2018)
12. Ke, J., Qin, X., Yang, H., Zheng, Z., Zhu, Z., Ye, J.: Predicting origin-destination ride-sourcing demand with a spatio-temporal encoder-decoder residual multi-graph convolutional network. arXiv preprint arXiv:1910.09103 (2019)
13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
14. Lei, K., Qin, M., Bai, B., Zhang, G., Yang, M.: GCN-GAN: a non-linear temporal link prediction model for weighted dynamic networks. In: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, pp. 388–396. IEEE (2019)
15. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp. Res. Part C Emerg. Technol. **54**, 187–197 (2015)
16. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297, Oakland, CA, USA (1967)
17. Wu, Y., Tan, H.: Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework. arXiv preprint arXiv:1612.01022 (2016)
18. Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. arXiv preprint arXiv:1610.00081 (2016)
19. Zhang, J., Zheng, Y., Qi, D., Li, R., Yi, X.: DNN-based prediction model for spatio-temporal data. In: Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 1–4 (2016)
20. Zhang, J., Zheng, Y., Sun, J., Qi, D.: Flow prediction in spatio-temporal networks based on multitask deep learning. IEEE Comput. Archit. Lett. **32**(03), 468–478 (2020)
21. Zhao, L., et al.: T-GCN: a temporal graph convolutional network for traffic prediction (2018)
22. Zhao-sheng, Y., Yuan, W., Qing, G.: Short-term traffic flow prediction method based on SVM. J. Jilin Univ. (Eng. Technol. Ed.) **6**, 009 (2006)
23. Zheng, C., Fan, X., Wang, C., Qi, J.: GMAN: a graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1234–1241 (2020)