STG2Seq: Spatial-Temporal Graph to Sequence Model for Multi-step Passenger Demand Forecasting

Lei Bai[1*] , **Lina Yao**[1] , **Salil S. Kanhere**[1] , **Xianzhi Wang**[2] and **Quan Z. Sheng**[3]

[1]University of New South Wales
[2]University of Technology Sydney
[3]Macquarie University

baisanshi@gmail.com, {lina.yao, salil.kanhere}@unsw.edu.au, xianzhi.wang@uts.edu.au,
michael.sheng@mq.edu.au

Abstract

Multi-step passenger demand forecasting is a crucial task in on-demand vehicle sharing services. However, predicting passenger demand over multiple time horizons is generally challenging due to the nonlinear and dynamic spatial-temporal dependencies. In this work, we propose to model multi-step citywide passenger demand prediction based on a graph and use a hierarchical graph convolutional structure to capture both spatial and temporal correlations simultaneously. Our model consists of three parts: 1) a long-term encoder to encode historical passenger demands; 2) a short-term encoder to derive the next-step prediction for generating multi-step prediction; 3) an attention-based output module to model the dynamic temporal and channel-wise information. Experiments on three real-world datasets show that our model consistently outperforms many baseline methods and state-of-the-art models.

1 Introduction

The widespread adoption of on-demand vehicle (cars and bikes) sharing services has revolutionized urban transportation. Passengers can now easily discover available vehicles in their surroundings using apps running on smartphones such as Uber, Didi, and Ola. However, these new ride-sharing platforms still suffer from the age-old problem of low accuracy in predicting passenger demand. On the one hand, drivers often have to drive a long way before they can find passengers due to low demand volumes in their proximity; on the other hand, passengers may experience long delays in obtaining rides due to high demands around their locations. This mismatch often leads to the excessive waiting time of passengers and a loss of income and waster energy resources for the drivers [Bai *et al.*, 2019]. In particular, accurate prediction of passenger demand over multiple time steps (i.e., multi-step prediction) in different regions of the city is crucial for effective vehicle dispatching to overcome the aforementioned mismatch problem.
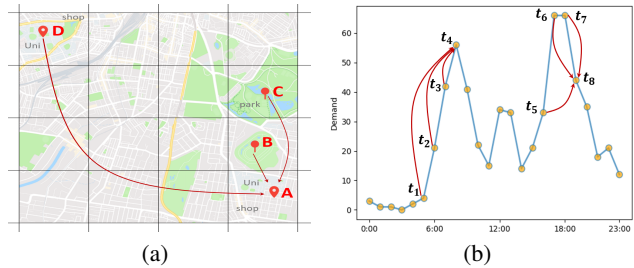
---

*Contact Author

Figure 1: (a): Example of non-Euclidean spatial correlation; (b): Illustration of dynamic temporal correlation.

Predicting passenger demand is a challenging task due to the complex, nonlinear and dynamic spatial-temporal dependencies: the future passenger demand of a target region is influenced by not only the historical demand of this region but also the demand of other regions in the city [Zhang *et al.*, 2017; Yao *et al.*, 2018]. Previous work have proposed to use time series models [Moreira-Matias *et al.*, 2013], cluster models [Li *et al.*, 2015], or hybrid methods [Zhang *et al.*, 2016b] to capture the correlations. Recent works focus on leveraging the representation capabilities of deep learning methods. These methods usually employ RNN and its variants such as Long-Short Term Memory (LSTM) [Yao *et al.*, 2018] networks to capture temporal correlation and CNN [LeCun *et al.*, 2015] to extract spatial relationships from the whole city [Zhang *et al.*, 2016a; Zhang *et al.*, 2017] or geographically nearest regions [Yao *et al.*, 2018]. Similarly, hybrid models that combine CNN and RNN (e.g., Convolutional LSTM (ConvLSTM) [Xingjian *et al.*, 2015; Ke *et al.*, 2017; Zhou *et al.*, 2018]) are proposed to extract both spatial and temporal correlations simultaneously.

However, these methods suffer from the following drawbacks:

- CNN-based methods (including ConvLSTM) assume that a city is partitioned into small grids (such as 1km × 1km area), which does not always hold [Chu *et al.*, 2018]. Moreover, these methods can only model the Euclidean relationships between near regions and remote regions but not the non-Euclidean correlation among remote regions with similar characteristics. Consider the example in Figure 1(a). Region A shares points of in-

terests with region D (university and shopping areas) rather than co-located regions B and C (which contain parks). Thus, the passenger demand in region A shows a stronger correlation with D rather than B and C.

- Current methods strictly rely on RNN-based architecture (e.g., hybrid CNN-LSTM and ConvLSTM architectures) to capture temporal correlations. However, typical chain structured RNN architectures require execution of a number of iterative steps (equal to the window size of the input data) to process the demand data, and therefore lead to severe information oblivion in modeling the long-term temporal dependency. Moreover, utilizing RNN as a decoder for multi-step prediction is known to cause error accumulation in every step and may result in faster model deterioration [Yu *et al.*, 2018].

- The current research efforts do not capture the dynamics that may exist in temporal correlation accurately. Most of them only reflect the collective influence of historical passenger demands. However, each previous step may have different and time-varying influence on the target step. Figure 1 (b) illustrates a passenger demand time series for a particular region where the influence of t_1, t_2 and t_3 on t_4 varies significantly. Moreover, the importance of t_1, t_2, t_3 on t_4 is different from the importance of t_5, t_6, t_7 on t_8.

In this paper, we propose a sequence-to-sequence model for multi-step passenger demand forecasting that is based on Graph Convolutional Networks (GCN) to solve the issues above. Specifically, we formulate the passenger demand on a graph with each region in the city acting as a node. Multiple GCN layers are utilized to form a Gated Graph Convolutional Module (GGCM) to capture the spatial and temporal relationships at the same time. Based on the GGCM, two encoder modules named long-term encoder and short-term encoder are designed to encode historical passenger demand and integrate new predictions, separately. Compared to the chain structured RNN, the hierarchical GCN structure shortens the path to capture the long-range temporal dependency [Gehring *et al.*, 2017]. Besides, having two distinct encoders allows our model to utilize last step's prediction to generate the next step's prediction without requiring a RNN to act as a decoder, which reduces the associated issue of error accumulation. Finally, we also take into account the dynamic temporal correlations and design an attention-based output module, which can adaptively capture these dynamics. Overall, the contribution of this paper can be summarized as follows:

- We formulate the citywide passenger demand on a graph and present a GCN-based sequence-to-sequence model for citywide multi-step passenger demand forecasting. To the best of our knowledge, this is the first work that purely relies on graph convolution structure to extract spatial-temporal correlations for multi-step prediction.

- We propose an attention-based output module to capture the effect of the most influential historical time steps on the predicted demand and the dynamism that is inherent in these relationships.

- We conduct extensive experiments on three real-world

datasets and compare our method with three baselines and eight discriminative deep learning based state-of-the-art methods. The experimental results show that our model can consistently outperform all the comparison methods by a significant margin.

2 Notations and Problem Statement

Suppose a city is partitioned into N small regions, irrespective of whether grid [Zhang *et al.*, 2017] or road network [Chu *et al.*, 2018] based partitioning is employed. We represent the region sets as $\{r_1, r_2, ..., r_i, ...r_N\}$. At each time step t, a 2-D matrix $\boldsymbol{D_t} \in \mathbb{R}^{N \times d_{in}}$ represents the passenger demand of all regions in time step t. Another vector $\boldsymbol{E_t} \in \mathbb{R}^{d_e}$ represent the time features in time step t, which includes time of day, day of week, and information about holidays.

Given the citywide historical passenger demand sequence $\{\boldsymbol{D_0}, \boldsymbol{D_1}, ..., \boldsymbol{D_t}\}$ and time features $\{\boldsymbol{E_0}, \boldsymbol{E_1}, ..., \boldsymbol{E_{t+\tau}}\}$, the target is to learn a prediction function $\boldsymbol{\Gamma}(\cdot)$ that forecasts the citywide passenger demand sequence in the next τ time steps. Instead of using all the historical passenger demand, we only consider the most recent h time steps demand sequence $\{\boldsymbol{D_{t-h+1}}, \boldsymbol{D_{t-h+2}}, ..., \boldsymbol{D_t}\}$ as input, which is a common practice in time series data analysis. Our problem is thus formulated as:

$$\begin{aligned}(\boldsymbol{D_{t+1}}, \boldsymbol{D_{t+2}}, &..., \boldsymbol{D_T}, ..., \boldsymbol{D_{t+\tau}}) = \\ \boldsymbol{\Gamma}(\boldsymbol{D_{t-h+1}}, &\boldsymbol{D_{t-h+2}}, ..., \boldsymbol{D_t}; \boldsymbol{E_0}, \boldsymbol{E_1}, ..., \boldsymbol{E_{t+\tau}})\end{aligned} \quad (1)$$

3 Methodology

The architecture of STG2Seq (Figure 2) comprises three components: (i) the long-term encoder (ii) the short-term encoder and (iii) the attention-based output module. Both the long-term encoder and short-term encoder comprise several serial spatial-temporal Gated Graph Convolutional Modules (GGCM), which can extract the spatial-temporal correlations simultaneously through the use of GCN along the temporal axis. We will elaborate on each component as follows.

3.1 Passenger Demand on Graph

We first introduce how to formulate the citywide passenger demand on a graph. Previous works assume that the passenger demand in a region is influenced by the demand in nearby regions. However, we argue that spatial correlation does not exclusively rely on geographic locations. Remote regions may also share similar passenger demand patterns if they have similar attributes, such as point of interests (POIs). Therefore, we treat the city as a graph $G = (\nu, \xi, A)$, where ν is the set of regions $\nu = \{r_i | i = 1, 2, ...N\}$, ξ denotes the set of edges and A the adjacency matrix. We define the connectivity of the graph according to the similarity of the passenger demand patterns among the regions.

$$A_{i,j} = \begin{cases} 1, & \text{if } Similarity_{r_i, r_j} > \epsilon \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

where ϵ is a threshold to which will determine the sparsity of matrix A. To quantify the similarity in passenger demand between different regions, we use the Pearson Correlation Coefficient. Let $D_{0 \sim t}(r_i)$ represent the historical passenger demand sequence for region r_i from time 0 to t (in the training
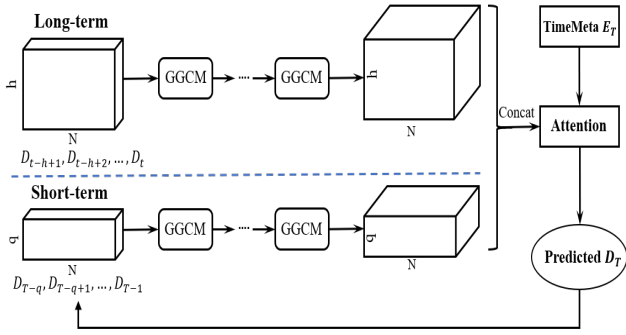
Figure 2: The architecture of STG2Seq.

Figure 3: An illustration of extracting spatial-temporal correlations with GCN.

data). Then the similarity of region r_i and r_j can be defined as:

$$Similarity_{r_i,r_j} = Pearson(D_{0 \sim t}(r_i), D_{0 \sim t}(r_j)) \quad (3)$$

3.2 Long-term and Short-term Encoders

Most prior works only consider next-step prediction, i.e., predicting passenger demand in the next time step. They optimize these models by reducing the error incurred in the prediction for the next time step in the training stage without considering the subsequent time steps. Hence, these methods are known to deteriorate rapidly for multi-step prediction. Only a few works have considered the problem of prediction for multiple time steps [Xingjian *et al.*, 2015; Li *et al.*, 2018]. These works adopt an encoder-decoder architecture based on RNN or its variants (i.e., ConvLSTM) as the encoder and decoder. These methods have two disadvantages: (1) The chain-structured RNNs employed in the encoder iterate over one input time step at a time. Thus, they require an equivalent number (i.e., h) of iterative RNN units when using the historical data over h time steps as the input. The long calculation distance between target demand and previous demands could cause severe information oblivion. (2) In the decoder part, to predict the demand for time step T, RNN takes as input the hidden state and the prediction of the previous time step $T - 1$. Thus, the errors from the previous time step are carried forward and directly influence the prediction, which results in error accumulation in each future step.

Different to all these previous works, we introduce an architecture that relies on the long-term and short-term encoder operating simultaneously to achieve multi-step prediction without the use of RNN. The long-term encoder takes the most recent h time steps citywide historical passenger demand sequence $\{D_{t-h+1}, D_{t-h+2}, ..., D_t\}$ as input to learn the historical spatial-temporal patterns. These h steps citywide demand are combined and organized into a 3-D matrix, shape $h \times N \times d_{in}$. The long term encoder comprises a number of GCCMs, wherein each GGCM captures spatial correlation among all N regions and temporal correlation among k (the patch size, a hyperparameter) time steps, which we will elaborate on in Section 3.3. Thus, only $\frac{h-1}{k-1}$ iterative steps are needed to capture the temporal correlation over the h historical steps. Compared to a RNN structure, our GGCM-based long-term encoder significantly decreases the iterative
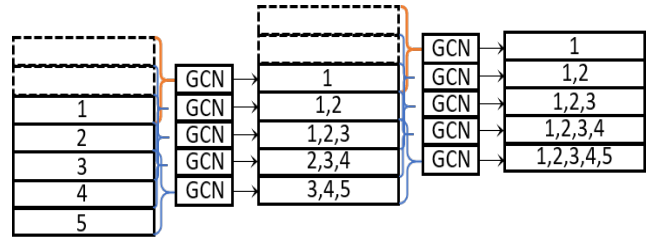
steps, which can further decrease the information loss. The output of the long-term encoder is another matrix Y_h shaped $h \times N \times d_{out}$, which is the encoded representation of the input

The short-term encoder is used to integrate already predicted demand for multi-step prediction. It uses a sliding window sized q to capture the recent spatial-temporal correlations. When predicting the passenger demand at time step T ($T \in [t + 1, t + \tau]$), it takes the citywide passenger demand of the most recent q time steps, i.e., $\{D_{T-q}, D_{T-q+1}, ..., D_{T-1}\}$ as input. Except for the length of time steps (h and q in long-term and short-term, respectively), the operation of the short-term encoder is the same as the long-term encoder. The short-term encoder generates a matrix Y_q^T shaped $q \times N \times d_{out}$ as the near trend representation. In contrast to a RNN-based decoder, the prediction of the last time step is fed back exclusively to the short-term encoder. Thus, the prediction error will be attenuated by the long-term encoder, which eases the severe error accumulation problem that is inherent in the RNN-based decoder model [Yu *et al.*, 2018].

3.3 Gated Graph Convolutional Module

The Gated Graph Convolutional Module is the core of both long-term encoder and short-term encoder. Each GGCM consists of several GCN layers, which are parallelized along the temporal axis. To capture the spatial-temporal correlations, each GCN layer operates on a limited historical window (k time steps) of the citywide demand data. It can extract the spatial correlation among all regions within these k time steps. By stacking multiple serial GGCMs, our model forms a hierarchical structure and can capture the spatial-temporal correlations from the entire input (h for long-term encoder and q for short-term encoder, $h > q \geq k$). Figure 3 illustrates the exclusive use of GCN for extracting spatial-temporal correlations, where we omit the channel (dimension) axis for simplicity. In literature, there is one similar work [Yu *et al.*, 2018] to our GGCM module. Their work first employs CNN to capture temporal correlation and then uses GCN to capture spatial correlation. Our approach is significantly simplified compared to theirs, as we can extract the spatial-temporal correlations simultaneously.

The detailed design of the GGCM module is shown in Figure 4. The input of the l_{th} GGCM is a matrix shaped $h \times N \times C^l$ (or $q \times N \times C^l$ for short-term encoder, we only use h in the following for simplicity), where C^l is the input dimension. In the first GGCM module, C^l is d_{in} (as noted in Section 2). The output shape of l_{th} GGCM is
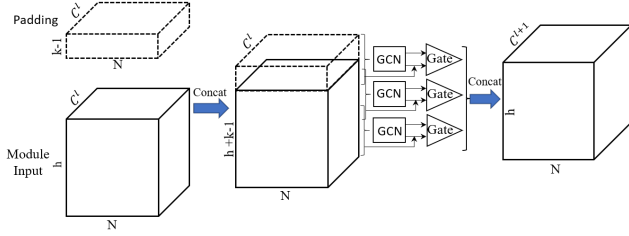
Figure 4: The spatial-termporal Gated Graph Convolutional Module

$h \times N \times C^{l+1}$. We first concatenate a zero padding matrix shaped $(k-1) \times N \times C^l$ and form the new input $(h+k-1) \times N \times C^l$ to ensure the transformation do not decrease the length of the sequence. Next, each GCN in the GGCM takes k time steps data shaped $k \times N \times C^l$ as input to extract spatial-temporal correlations, which is further reshaped as a 2-D matrix $N \times (k \cdot C^l)$ for GCN calculation. According to [Kipf and Welling, 2017], the calculation of a GCN layer can be formulated as:

$$X^{l+1} = (\widetilde{P}^{-\frac{1}{2}} \widetilde{A} \widetilde{P}^{-\frac{1}{2}}) X^l W \quad (4)$$

where $\widetilde{A} = A + I_n$ (A is the adjacency matrix of the graph defined in Section 3.1, I_n is the identity matrix), $\widetilde{P}_{ii} = \sum_j \widetilde{A}_{ij}$, $X \in \mathbb{R}^{N \times (k \cdot C^l)}$ denotes the reshaped k time steps demand, $W \in \mathbb{R}^{(k \cdot C^l) \times C^{l+1}}$ represents the learned parameters, $X^{l+1} \in \mathbb{R}^{N \times C^{l+1}}$ is the output of GCN.

Furthermore, we adopt the gating mechanism [Dauphin *et al.*, 2017] to model the complex non-linearity in passenger demand forecasting. And Eq. (4) is re-formulated as:

$$X^{l+1} = ((\widetilde{P}^{-\frac{1}{2}} \widetilde{A} \widetilde{P}^{-\frac{1}{2}}) X^l W_1 + X^l) \otimes \sigma((\widetilde{P}^{-\frac{1}{2}} \widetilde{A} \widetilde{P}^{-\frac{1}{2}}) X^l W_2) \quad (5)$$

where $\otimes$ is the element-wise product operation, σ denotes the sigmoid function. Thus, the output is a linear transformation $(\widetilde{P}^{-\frac{1}{2}} \widetilde{A} \widetilde{P}^{-\frac{1}{2}}) X^l W_1 + X^l$ modulated by a non-linear gate $\sigma((\widetilde{P}^{-\frac{1}{2}} \widetilde{A} \widetilde{P}^{-\frac{1}{2}}) X^l W_2)$. The non-linear gate controls which part of the linear transformation can pass through the gate and contribute to the prediction. Besides, residual connection [He *et al.*, 2016] is utilized to avoid the network degradation as shown in Eq. (5).

Finally, the h outputs from the gating mechanism are combined along the temporal axis to generate the output of one GGCM module shaped $h \times N \times C^{l+1}$.

3.4 Attention-based Output Module

As notated in Section 3.2, the long-term spatial-temporal dependency and recent time spatial-temporal dependency are captured and represented as two matrices Y_h and Y_q^T for the target time step T. We concatenate them together to form a joint representation $Y_{h+q} \in \mathbb{R}^{(h+q) \times N \times d_{out}}$, which will be decoded by the attention-based output module to obtain the prediction (we omit T here and in the following for simplicity). The three axes of Y_{h+q} are time, space (i.e., region) and channel (i.e., dimension), respectively.

We first introduce the temporal attention mechanism for decoding Y_{h+q}. The passenger demand is a typical time series

data as previous historical demands have an influence on the future passenger demand. However, the importance of each previous step to target demand is different, and this influence changes with time. We design a temporal attention mechanism to add an importance score for each historical time step to measure the influence. The score is generated by aligning the joint representation $Y_{h+q} = [y_1, y_2, ..., y_{h+q}](y_i \in \mathbb{R}^{(N \times d_{out})})$ with the target time step's time features E_T, which can adaptively learn the dynamic temporal influence of each previous time step changing with time. We define the calculation of temporal attention as:

$$\boldsymbol{\alpha} = softmax(tanh(Y_{h+q} W_3^Y + E_T W_4^E + b_1)) \quad (6)$$

where $W_3^Y \in \mathbb{R}^{(h+q) \times (N \times d_{out}) \times 1}$, $W_4^E \in \mathbb{R}^{d_e \times (h+q)}$ and $b_1 \in \mathbb{R}^{(h+q)}$ are transformation matrices to be learned, $\boldsymbol{\alpha} \in \mathbb{R}^{(h+q)}$ is the temporal importance score which is normalized by the softmax function $softmax(\cdot)$. The joint representation Y_{h+q} is then transformed by the importance score $\boldsymbol{\alpha}$:

$$Y_\alpha = \sum_{i=1}^{h+q} \alpha^i y_i \qquad \in \mathbb{R}^{N \times d_{out}} \quad (7)$$

Inspired by [Chen *et al.*, 2017] which showed that the importance of each channel is also different, we further add a channel attention module after the temporal attention module to find the most important frames in $Y_\alpha = [\mathscr{Y}_1, \mathscr{Y}_2, ..., \mathscr{Y}_{d_{out}}]$ with $\mathscr{Y}_i \in \mathbb{R}^N$. The calculation of channel attention is similar to temporal attention:

$$\boldsymbol{\beta} = softmax(tanh(Y_\alpha W_5^Y + E_T W_6^E + b_2)) \quad (8)$$

$$Y_\beta = \sum_{i=1}^{d_{out}} \beta^i \mathscr{Y}_i \qquad \in \mathbb{R}^N \quad (9)$$

where $W_5^Y \in \mathbb{R}^{d_{out} \times N \times 1}$, $W_6^E \in \mathbb{R}^{d_e \times d_{out}}$, b_2 are transformation matrices; $\boldsymbol{\beta} \in \mathbb{R}^{d_{out}}$ is the importance score for each channel. In the case that target demand dimension is 1, Y_β is our predicted passenger demand $\boldsymbol{D'_T}$. When the target demand dimension is 2 (predict both start-demand and end-demand [Yao *et al.*, 2019]), we can simply conduct channel attention for each dimension and concatenate them together to form the predicted passenger demand $\boldsymbol{D'_T}$.

3.5 Optimization

The outputs of all the τ time steps constitute the predicted passenger demand sequence $(\boldsymbol{D'_{t+1}}, \boldsymbol{D'_{t+2}}, ..., \boldsymbol{D'_{t+\tau}})$. In the training process, our objective is to minimize the error between the predicted and actual passenger demand sequences. We define the loss function as sum of the mean squared error between the predicted and actual passenger demand for τ time steps, written as:

$$\mathcal{L}(W_\theta) = \sum_{t+1}^{t+\tau} \|\boldsymbol{D_T} - \boldsymbol{D'_T}\|_2^2 \quad (10)$$

where W_θ represents all the learnable parameters in the network. These can be obtained via back-propagation and Adam optimizer. Further, we use the teacher forcing strategy in the training stage to achieve high efficiency. Specifically, we always use the true value for short-term encoder instead of the predicted value when training the model.

Index	Method	DidiSY			BikeNYC			TaxiBJ		
		RMSE	MAE	MAPE	RMSE	MAE	MAPE	RMSE	MAE	MAPE
1	HA	4.112	2.646	0.426	8.541	3.695	0.437	40.439	20.696	0.268
2	OLR	3.713	2.528	0.379	8.502	4.652	0.391	23.921	14.937	0.276
3	XGBoost	3.612	2.394	0.402	6.914	3.423	0.367	22.927	13.687	0.212
4	DeepST	3.362	2.221	0.337	6.603	2.549	0.242	18.305	11.264	0.157
5	ResST-Net	3.449	2.331	0.318	6.159	2.432	0.228	17.649	10.599	0.141
6	DMVST-Net	3.440	2.232	0.373	4.766	2.318	0.224	18.206	11.085	0.153
7	ConvLSTM	3.414	2.222	0.379	4.745	2.435	0.226	18.788	11.461	0.163
8	FCL-Net	3.364	2.172	0.381	4.959	2.362	0.275	18.176	10.756	0.169
9	FlowFLexDP	3.292	2.143	0.336	6.003	2.801	0.271	19.538	11.945	0.160
10	DCRNN	3.465	2.281	0.371	5.215	2.776	0.241	20.569	12.517	0.177
11	STGCN	3.397	2.236	0.372	4.759	2.438	0.220	19.101	11.573	0.167
12	**STG2Seq**	**3.206**	**2.134**	**0.306**	**4.513**	**2.257**	**0.210**	**17.241**	**10.219**	**0.138**

Table 1: Evaluation on next-step prediction over three datasets of different scales (best performance displayed in bold).

4 Experiments

4.1 Experimental Setup

We use three real-world datasets in our comparisons:

- **DiDiSY**: This is a self-collected dataset that consists of 1) share car demand data from Didi, the biggest online ride-sharing company in China; 2) Time meta, including time of day, day of week, and holidays; This dataset was collected from Dec 5th, 2016 to Feb 4th, 2017 in Shenyang, a large city in China. Each time step is one hour. We use the data from the last six days for testing while the rest for training.

- **BikeNYC** [Zhang *et al.*, 2017]: The public BikeNYC dataset consists of the bike demand and the time meta. The bike demand covers the shared bike hire and returns data of CityBike in New York from 1 Apr 2014 to 30th Sept 2014. Each time step is one hour. To be consistent with previous works that used this dataset [Zhang *et al.*, 2016a][Zhang *et al.*, 2017], the last ten days' data are used for testing.

- **TaxiBJ** [Zhang *et al.*, 2017]: The public TaxiBJ dataset contains taxi demand in Beijing from 1 Mar 2015 to 30th Jun 2015. Similar to the DiDiSY dataset, TaxiBJ contains passenger demand, time meta, and meteorological data. Each time step is 30 minutes. The data of the last ten days is used for testing to keep consistent with previous works.

We implemented our model in Python with TensorFlow 1.8. Historical passenger demand length h is set to 12, sliding window size q and patch size k are both set to 3. At each time step, we use three evaluation metrics to evaluate the model: Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE).

4.2 Experimental Results

Next-step Prediction Comparison

We first compare our method with three representative traditional baselines: 1) Historical Average (HA); 2) Ordinary Linear Regression (OLR); 3) XGBoost [Chen and Guestrin, 2016]; and eight discriminative state-of-the-art methods: 4) DeepST [Zhang *et al.*, 2016a]; 5) ResST-Net [Zhang *et al.*, 2017]; 6) DMVST-Net [Yao *et al.*, 2018]; 7) ConvLSTM [Xingjian *et al.*, 2015]; 8) FCL-Net [Ke *et al.*, 2017]; 9) FlowFlexDP [Chu *et al.*, 2018]; 10)DCRNN [Li *et al.*, 2018]; 11) STGCN [Yu *et al.*, 2018].

Because most state-of-the-art methods can only achieve next-step prediction, we use the first time step result of STG2Seq in this comparison. Table 1 presents the comprehensive comparison results. We observe the following: (1) deep learning methods always outperform non-deep learning methods such as HA, OLSR, and XGBoost, which shows the superiority of deep learning methods in capturing non-linear spatial-temporal correlations. (2) Our model consistently achieves the best performance and outperforms other methods with a significant margin in all three datasets. More specifically, STG2Seq gains 2.6%, 4.9% and 2.3% relative improvements in RMSE; 0.4%, 2.6% and 3.6% relative improvements in MAE; 3.8%, 4.5% and 2.1% relative improvements in MAPE over the best state-of-the-art methods for the three datasets, respectively. The results indicate that our model can capture more accurate spatial-temporal correlations than the state-of-the-art.

Multi-step Prediction Comparison

Next, we compare our STG2Seq model with HA, ConvL-STM, and DCRNN, which are also capable of conducting multi-step prediction. Each method predicts the passenger demand in the following 3 time steps. Figure 5 presents the experimental results about RMSE and MAE in three datasets. We can observe that the prediction error for HA is large but consistent for all 3 time steps. ConvLSTM and DCRNN can achieve good prediction in the first time step. However, they deteriorate fast with time, especially with the DiDiSY dataset. STG2Seq can achieve good prediction result for all time steps and deteriorates slower than ConvLSTM and DCRNN, which demonstrates that our method is effective and eases the error accumulation problem that is inherent in RNN-based decoder for multi-step prediction.

Index	Removed Component	RMSE			MAE			MAPE		
		step1	step2	step3	step1	step2	step3	step1	step2	step3
1	Short-term Encoder	4.509	\	\	2.304	\	\	0.211	\	\
2	Temporal Attention	4.540	5.259	5.730	2.350	2.674	2.811	0.211	0.229	0.246
3	Channel Attention	4.618	5.310	5.861	2.334	2.552	2.717	0.213	0.239	0.258
4	Gate Mechanism	4.576	5.222	5.658	2.314	2.507	2.664	0.213	0.232	0.246
5	Teacher Forcing	4.574	5.245	5.873	2.324	2.491	2.834	0.212	0.232	0.250
6	STG2Seq	4.513	5.209	5.497	2.257	2.452	2.555	0.210	0.228	0.240

Table 2: Evaluation of different variants on Bike_NYC dataset.

Component Analysis

To evaluate the effect of each component on our model, we compare five variants obtained by (1) removing the short-term encoder, (2) replacing the temporal attention module by a 2-D CNN layer (for reducing the dimension), (3) replacing the channel attention module by a 2-D CNN layer, (4) replacing the gate mechanism in GGCM by Relu activation function, and (5) removing the teacher forcing strategy when training the model. The experimental results on Bike_NYC dataset are shown in Table 2. We gain three observations from this table. First, without the short-term encoder, the model degenerates to a single-step prediction model. Second, the temporal and channel attention modules not only improve the prediction accuracy but also slow down the rate of deterioration in multi-step prediction, which demonstrates the importance of both parts and the effectiveness of our design. Third, the gating mechanism is better at modeling non-linearities when compared with Relu. The results also exemplify the progressive

Index	Method	RMSE	MAE
1	HA	4.231	2.714
2	ARIMA	4.001	2.681
3	SARIMA	3.937	2.619
4	OLR	3.719	2.496
5	MLP	3.699	2.436
6	XGBoost	3.533	2.341
7	FlowFLexDP	3.518	2.322
8	DCRNN	3.526	2.332
9	STGCN	3.433	2.281
10	STG2Seq	3.244	2.136

Table 3: Prediction results on irregular regions

nature of our network design.

Prediction for Irregular Regions

Previous results rely on the assumption that the target regions are partitioned as regular grids. In this experiment, we investigate the feasibility of our method when the city is partitioned into irregular regions. The DidiSY dataset contains the precise GPS location of each service request. This allows us to re-partition the city (Shenyang) into sub-regions based on the road network, which results in irregularly sized partitions. Under this setting, our method is flexible and can be applied to the re-organized dataset without modification. However, most state-of-the-arts introduced in the next-step prediction comparison part cannot be used as they are only suitable to extract Euclidean correlations when the city is partitioned to equally sized grids. To benchmark our method, we include three better suited comparative methods, namely: Auto-Regressive Integrated Moving Average model (ARIMA), Seasonal Auto-Regressive Integrated Moving model (SARIMA) and Multiple Layer Perceptron (MLP). The results in Table 3 show that STG2Seq significantly outperforms the baselines, thus demonstrating its generality.

5 Conclusion

In this paper, we propose a novel deep learning framework for multi-step citywide passenger demand forecasting. We formulate the citywide passenger demand on a graph and employ the hierarchical graph convolution architecture to extract spatial and temporal correlations simultaneously. The long-term encoder and short-term encoder are introduced to achieve multi-step prediction without relying on RNN. Moreover, our model considers the dynamic attribute in temporal correlation by using the attention mechanism. Experimental results on three real-world datasets show that our model outperforms other state-of-the-art methods by a large margin.
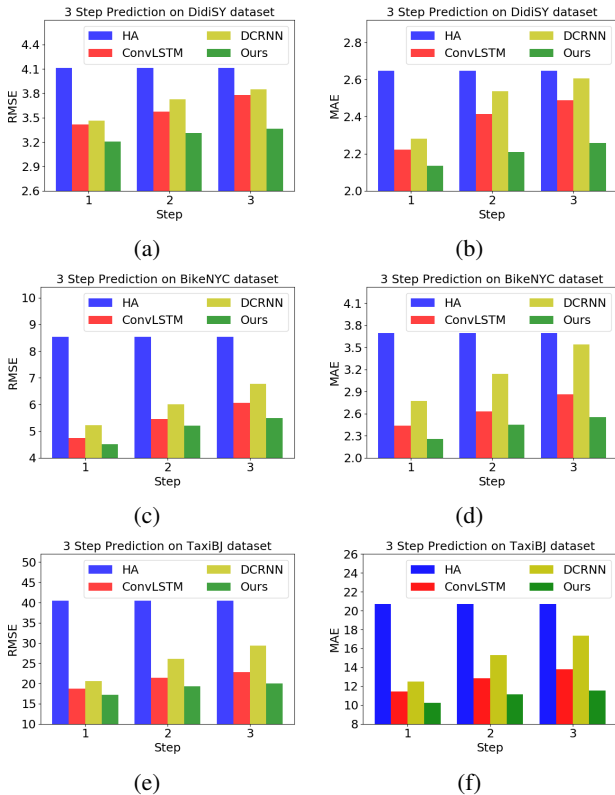

Figure 5: Evaluation on multi-step prediction.

References

[Bai *et al.*, 2019] Lei Bai, Lina Yao, Salil S Kanhere, Zheng Yang, Jing Chu, and Xianzhi Wang. Passenger demand forecasting with multi-task convolutional recurrent neural networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 29–42. Springer, 2019.

[Chen and Guestrin, 2016] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[Chen *et al.*, 2017] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6298–6306. IEEE, 2017.

[Chu *et al.*, 2018] Jing Chu, Kun Qian, Xu Wang, Lina Yao, Fu Xiao, Jianbo Li, Xin Miao, and Zheng Yang. Passenger demand prediction with cellular footprints. In *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pages 1–9. IEEE, 2018.

[Dauphin *et al.*, 2017] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International Conference on Machine Learning*, pages 933–941, 2017.

[Gehring *et al.*, 2017] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252, 2017.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Ke *et al.*, 2017] Jintao Ke, Hongyu Zheng, Hai Yang, and Xiqun Michael Chen. Short-term forecasting of passenger demand under on-demand ride services: A spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies*, 85:591–608, 2017.

[Kipf and Welling, 2017] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations (ICLR)*, Apr 2017.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[Li *et al.*, 2015] Yexin Li, Yu Zheng, Huichu Zhang, and Lei Chen. Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 33. ACM, 2015.

[Li *et al.*, 2018] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *ICLR*, Sep 2018.

[Moreira-Matias *et al.*, 2013] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. Predicting taxi–passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1393–1402, 2013.

[Xingjian *et al.*, 2015] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[Yao *et al.*, 2018] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. Deep multi-view spatial-temporal network for taxi demand prediction. In *2018 AAAI Conference on Artificial Intelligence (AAAI'18)*, 2018.

[Yao *et al.*, 2019] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, and Zhenhui Li. Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction. In *2019 AAAI Conference on Artificial Intelligence (AAAI'19)*, 2019.

[Yu *et al.*, 2018] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *IJCAI*, 2018.

[Zhang *et al.*, 2016a] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 92. ACM, 2016.

[Zhang *et al.*, 2016b] Kai Zhang, Zhiyong Feng, Shizhan Chen, Keman Huang, and Guiling Wang. A framework for passengers demand prediction and recommendation. In *Services Computing (SCC), 2016 IEEE International Conference on*, pages 340–347. IEEE, 2016.

[Zhang *et al.*, 2017] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *2017 AAAI Conference on Artificial Intelligence (AAAI'17)*, pages 1655–1661, 2017.

[Zhou *et al.*, 2018] Xian Zhou, Yanyan Shen, Yanmin Zhu, and Linpeng Huang. Predicting multi-step citywide passenger demands using attention-based neural networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 736–744. ACM, 2018.