Bike Flow Prediction with Multi-Graph Convolutional Networks

Di Chai
Clustar, Beijing, China
Hong Kong University of Science and Technology, HK SAR, China
dchai@connect.ust.hk

Leye Wang*
Peking University, Beijing, China
Hong Kong University of Science and Technology, HK SAR, China
wangleye@pku.edu.cn

Qiang Yang
Hong Kong University of Science and Technology, HK SAR, China
qyang@cse.ust.hk

ABSTRACT

One fundamental issue in managing bike sharing systems is bike flow prediction. Due to the hardness of predicting flow for a single station, recent research often predicts flow at cluster-level. However, they cannot directly guide fine-grained system management issues at station-level. In this paper, we revisit the problem of the station-level bike flow prediction, aiming to boost the prediction accuracy using the breakthroughs of deep learning techniques. We propose a multi-graph convolutional neural network model to predict flow at station-level, where the key novelty is viewing the bike sharing system from the graph perspective. More specifically, we construct multiple graphs for a bike sharing system to reflect heterogeneous inter-station relationships. Afterward, we fuse multiple graphs and apply the convolutional layers to predict station-level future bike flow. The results on realistic bike flow datasets verify that our multi-graph model can outperform state-of-the-art prediction models by reducing up to 25.1% prediction error.

CCS CONCEPTS

• **Computing methodologies** → *Neural networks*;

KEYWORDS

Graph Convolutional Network, Bike Flow Prediction

ACM Reference Format:
Di Chai, Leye Wang, and Qiang Yang. 2018. Bike Flow Prediction with Multi-Graph Convolutional Networks. In *26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '18), November 6–9, 2018, Seattle, WA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3274895.3274896

1 INTRODUCTION

Bike sharing systems are gaining increasing popularity in city transportation as a way to provide flexible transport mode and reduce the production of greenhouse gas. In a bike sharing system, users can check out at nearby stations and return the bike to the stations near the destination. Bike flow prediction is one of the key research and practical issues in bike sharing systems, which plays an important role in various tasks such as bike rebalancing [3, 8].

*Di Chai and Leye Wang contribute equally to this research, ordered alphabetically.

SIGSPATIAL '18, November 6–9, 2018, Seattle, WA, USA

ACM ISBN 978-1-4503-5889-7/18/11.
https://doi.org/10.1145/3274895.3274896

In reality, the bike flow of a single station usually has a complicated dynamic pattern, which makes it hard to predict [3]. As a result, most recent researchers try to address the bike flow prediction in a cluster-level. That is, they first group up the stations, and then predict the bike flow for each cluster [3, 8]. Although the cluster-level prediction accuracy is more satisfied, there are two limitations: (i) whether the output clusters are appropriate or not is hard to evaluate, and (ii) the prediction result at cluster-level cannot directly support the management on single stations.

In this paper, we revisit the station-level bike flow prediction problem, which can provide fine-grained information for the bike sharing administrators' decision making process and avoid the hard-to-evaluate clustering problem. To achieve this goal, we propose a novel multi-graph convolutional neural network model to catch heterogeneous inter-station spatial correlations. Traditional single station-level prediction usually pays more focus on the station's historical data, such as ARIMA [10]. However, in addition to this temporal correlations, the inter-station spatial correlations may also play an important role in bike flow prediction. In this work, we propose a new multi-graph convolutional neural network to capture heterogeneous spatial relationships between stations, such as distance and historical usage correlations. After the multi-graph convolutional layers, an encoder-decoder structure including LSTM (Long-Short Term Memory) units [5] is built to catch the temporal patterns. Hence, both spatial and temporal patterns are effectively captured for station-level bike flow prediction.

To the best of our knowledge, this is the first work of leveraging multi-graph convolutional neural networks in to predict station-level bike flow in a bike sharing system. Evaluations on real bike flow dataset in New York City and Chicago shows the effectiveness of our method. Compared with the state-of-the-art station-level bike flow prediction models, our multi-graph convolutional neural network model can reduce up to 25.1% prediction error.

Preliminary: *Graph Convolutional Neural Networks* were first introduced by Bruna et al. [1], which applies the convolutional layers on the graph data. It is later extended by Defferrard et al. [4] with fast localized convolutions. Two relevant papers to our work are [7, 11], both applying graph convolutional neural networks to predict traffic speed in road segments. Note that [7, 11] only use distance to create a graph; however, as one graph may not be able to describe inter-station relationships comprehensively, we propose new ways (in addition to distance) to construct inter-station graphs and further design a multi-graph convolutional network structure.

2 DEFINITIONS AND PROBLEM

In this section, we first define several key concepts, and then formulate the problem.

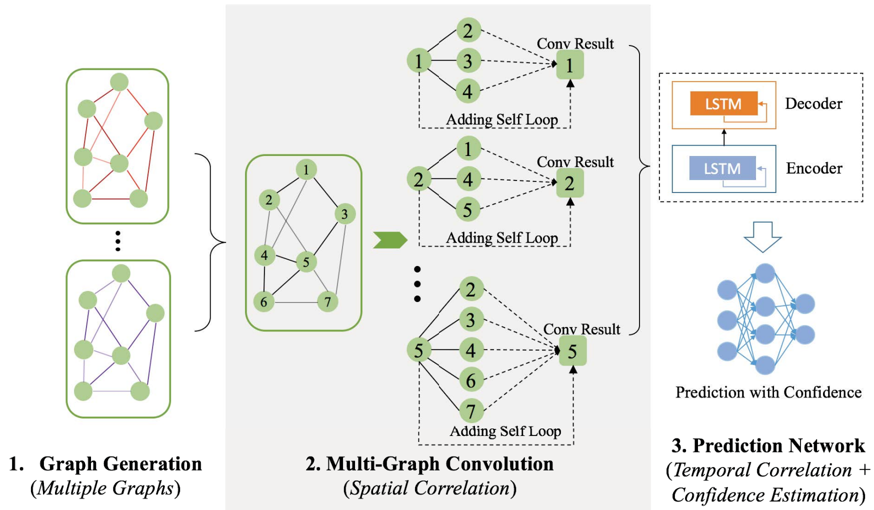

Figure 1: Overview of the Multi-Graph Convolutional Neural Network Model for Bike Flow Prediction. Note that our model can also output confidence interval of the prediction. Interested readers can refer to our full technical report for details [2].

Definition 1. *Bike-Sharing System Graph*: The bike-sharing system is represented as a weighted graph, whose nodes are stations and edges are inter-station relationships. The weights of edges represent the relation strength between stations. Usually, the larger weights mean that the two stations have higher correlations. How to construct the graph is the key part of our method.

Definition 2. *Bike Flow*: There are two types of bike flow: inflow and outflow. Suppose we have N bike stations, inflow at the time interval t (e.g., one-hour) can be denoted as $I^t = [ci_1^t, ci_2^t, ..., ci_N^t]$, outflow can be denoted as $O^t = [co_1^t, co_2^t, ..., co_N^t]$.

Problem: Suppose the current time is $t-1$, and we have the history data $[(I^0, O^0), (I^1, O^1), ..., (I^{t-1}, O^{t-1})]$. The problem is to predict the bike flow at the next time $(\hat{I}^t, \hat{O}^t)$, aiming to:

$$\min ||\hat{I}^t - I^t||_2^2, \quad \min ||\hat{O}^t - O^t||_2^2$$

where (I^t, O^t) is the ground truth bike flow of the next time t.

3 MULTI-GRAPH CONVOLUTIONAL NEURAL NETWORK MODEL

We propose a novel multi-graph convolutional neural network model (Figure 1) including three steps of *graph generation*, *multi-graph convolution*, and *prediction network*.

3.1 Graph Generation

Graph generation is the key to the success of graph convolutional model. If the constructed graph cannot encode the effective relationships between stations, it will not help the network parameter learning while even degrading the prediction performance. In general, we want to assign large weights to the edges between stations with similar dynamic flow patterns. Based on this idea, we propose three alternative ways for building inter-station graphs: *distance graph*, *interaction graph* and *correlation graph*.

Distance Graph: Tobler's first law of geography has pointed out that '*everything is related to everything else, but near things are more related than distant things*'.[1] In bike sharing systems, for two stations near each other (e.g., around a metro station), they may share similar usage patterns. Following this idea, we use the distance to construct the inter-station graphs. More specifically, we use the reciprocal of the distance to mark the weight between two stations so that closer stations will be linked with higher weights.

$$G_d(V, E) \qquad weight = Distance^{-1}$$

Interaction Graph: The historical ride records can also provide plenty of information to construct the inter-station graphs. For example, if there exist many ride records between station i and station j. Then the two stations i and j tend to affect each other regarding the dynamic bike flow patterns. With this idea in mind, we construct an *interaction graph* to indicate whether two stations are interacted with each other frequently according to the historical ride records. Denote $d_{i,j}$ as the number of ride records between i and j, we build the interaction graph as:

$$G_i(V, E) \qquad weight = \#RidingRecordNumber$$

Correlation Graph: With ride records, we also try another way to build the inter-station graph with the correlation of stations' historical usages. That is, we calculate the historical usages (inflow or outflow) of each station in each time slot (e.g., one hour), and then compute the correlations between every two stations as the inter-station link weights in the graph. In this work, we use the popular *Pearson coefficient* to calculate the correlation. Denote $r_{i,j}$ as the Pearson correlation between station i and station j, we can represent the correlation graph as follows:

$$G_c(V, E) \qquad weight = Correlation$$

3.2 Multi-graph Convolution

To fully exploit different inter-station graphs that contain heterogeneous useful spatial correlation information, we propose a novel multi-graph convolutional layer in our neural network model. It is

[1] https://en.wikipedia.org/wiki/Tobler%27s_first_law_of_geography

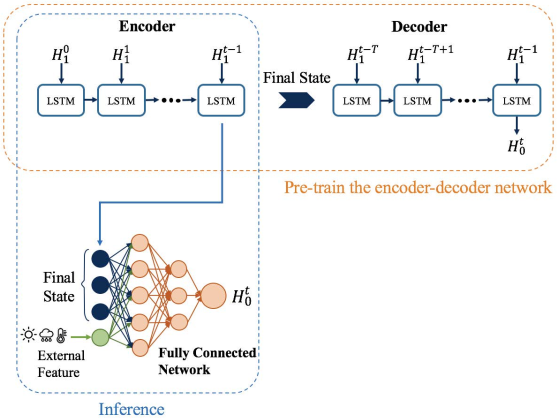

Figure 2: Structure of Prediction Network

able to conduct graph convolution on different kinds of graphs by merging them together first. There are two major steps of multi-graph convolution part: *graph fusion* and *graph convolution*.

Graph fusion: The graph fusion step merges different graphs into one fused graph. We combine different graphs by the weighted summing their adjacency matrices at the element level. Since the adjacency matrices' value of different graphs may fall in different ranges, we first normalize the adjacency matrix A for each graph.

$$A' = D^{-1}A + I$$

where D is :

$$D = \begin{pmatrix} \sum_{j=0}^{N-1} A_{0,j} & 0 & \cdots & 0 \\ 0 & \sum_{j=0}^{N-1} A_{1,j} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sum_{j=0}^{N-1} A_{N-1,j} \end{pmatrix}$$

The resultant A' is the normalized adjacency matrix with *self loop*. Self-loop can maintain the information of the target station itself in the convolution part, which is a required design strategy in graph convolutional neural networks.

To keep the fusion result normalized after the weighted sum operation, we further add a softmax operation to the weight matrix. Suppose we have N graphs to blend together, we can denote the graph fusion process as:

$$W_1', W_2', ..., W_N' = Softmax(W_1, W_2, ..., W_N)$$

$$F = \sum_{i=1}^{N} W_i' \circ A_i'$$

where $\circ$ is the element-wise product, F is the graph fusion result which will be used in the graph convolution part.

Graph convolution: Based on the graph fusion result F, we perform the graph convolution as:

$$H_0^{t'} = (I^{t'}, O^{t'}),\ t' \in [0, t-1]$$

$$H_1^{t'} = F * W_c * H_0^{t'}$$

where W_c is the convolution weight matrix, $H_0^{t'}$ is the bike flow at time t'. We take $H_1^{t'}$ as the convolution result, and then use $H_1^{t'}$ as the input of the next prediction network.

3.3 Prediction Network

As shown in Figure 2, the details of the prediction network with encoder-decoder structure is as follows. The encoder network takes the multi-graph convolutional result sequence $[H_1^0, H_1^1, ..., H_1^{t-1}]$ as input and encodes the temporal pattern into the final state of LSTM cell after the rolling process. The decoder network takes the encoder's final state as the initial state and the multi-graph convolutional result sequence $[H_1^{t-T}, H_1^{t-T+1}, ..., H_1^{t-1}]$ as input. The output of the decoder is H_0^t which is our prediction target. We can set T to a small value (e.g., half of t) which means that the decoder can predict the future bike flow based on a short period of history data and the encoder's final state. This implies that the encoder's final state provides important information for the predicting process. After training the encoder-decoder structure, we input the encoder network's final state, combined with some external context features (e.g., temperature, wind speed, weekday/weekend [12]) to a fully connected network (lower part of Figure 2) for predicting the bike flow in the next time H_0^t.

4 EVALUATION

4.1 Experiment Setting

Datasets: We used bike flow dataset collected from New York City and Chicago[2]. The datasets cover a four year time period (2013-2017). All the data are in the forms of riding record containing start station, start time, stop station, stop time and so on. Weather data comes from the NCEI website[3].

To set the training-validation-test data split, we choose the last 80 days in each city as test data, the 40 days before the test data are validation data, and all of the data before validate data are training data. The prediction granularity is set to one hour. Readers can find the code from https://github.com/Di-Chai/GraphCNN-Bike.

Network Implementation and Parameters: The encoder and decoder in the experiment contain one layer of LSTM and 64 hidden units. The fully connected prediction network contains 4 layers including the input and output layer. We choose the optimization algorithm as ADAM and the learning rate is set to 0.001% [6]. In the encoder-decoder structure of the prediction network, we set $T = 3$ in the decoder (refer to Figure 2). We use the past 6-hour history data to predict the bike flow in the next one hour.

Baselines: We compare our multi-graph convolutional network model with the following baselines:

- *HM* [8]: Historical Mean predicts the bike flow at a certain time slot according to the historical mean value of the same weekday and the same hour.
- *ARIMA* [10]: Auto-Regressive Integrated Moving Average is a widely used time series prediction model.
- *SARIMA* [10]: The seasonal version of ARIMA.
- *GBRT* [8]: Gradient Boosting Regression Tree is also widely used in bike flow prediction in literature.
- *LSTM* [9]: Recent studies in traffic flow prediction, such as [9], adopted the long short-term memory (LSTM) recurrent neural network model and verified its effectiveness.

[2]NYC bike sharing data: https://www.citibikenyc.com/system-data, Chicago bike sharing data: https://www.divvybikes.com/system-data
[3]https://www.ncdc.noaa.gov/data-access

Table 1: Prediction error in New York City and Chicago. Top stations are ranked by each station's total sum of bike flows in the historical ride records.

New York City

	Top 5 stations	*Top 10 stations*	*Average*
HM	7.572	7.055	5.997
ARIMA	11.329	9.545	8.049
SARIMA	8.677	7.363	6.521
GBRT	7.159	6.653	5.832
LSTM	6.802	5.981	5.345
Multi-graph	**4.745**	**4.473**	**4.003**

Chicago

	Top 5 stations	*Top 10 stations*	*Average*
HM	7.078	6.179	4.347
ARIMA	9.853	8.535	6.163
SARIMA	6.797	6.175	4.608
GBRT	5.945	5.738	4.410
LSTM	6.231	5.853	4.405
Multi-graph	**5.177**	**4.930**	**3.658**

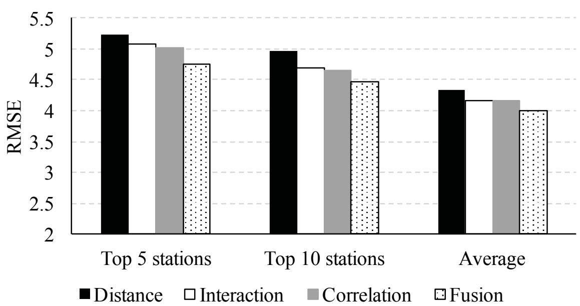

Figure 3: Comparison of multi-graph and single-graph convolutional models (New York City).

4.2 Experiment Results

Prediction error: We use RMSE (root mean square error) to measure the prediction error. Table 1 shows the results. In general, our multi-graph convolutional method can still beat the best baseline significantly by reducing the average prediction error by 25.1% and 15.8% in New York City and Chicago, respectively.

In addition to the average station-level prediction error, we investigate the prediction results for those stations with the highest usages in both cities. The prediction accuracy of these busy stations may be more important since most of the over-demand issues ('no bikes to use' or 'no docks to return a bike') could happen in those stations [3]. We thus select the top 5 and 10 busy stations to study the results in both cities. We observe that our multi-graph method can also consistently get better results for the busy stations. For example, for the top 5 and 10 busy stations in New York City, our method can outperform LSTM by 30.2% and 25.2%, respectively. This further verifies the practicability of our proposed method in real-life bike sharing system management.

Effectiveness of multi-graph fusion: Now we verify that our multi-graph fusion strategy can actually bring benefits to the prediction model. Figure 3 shows the results when we only use a single graph (*distance*, *interaction* or *correlation* graph) for prediction in New York City. Compared to the single-graph convolutional methods, our multi-graph convolutional method can perform consistently better. For example, for the top 5 busy stations in New York City, the multi-graph model can outperform the single-graph models by reducing error 5.6–9.2%. This improvement is also verified to be statistically significant (p-value < 0.05).

Computation efficiency: Our experiment runs in a Windows server with *CPU: Intel Xeon E5-2690, Memory: 56 GB, GPU: Nvidia Telsa K80*. The training time needs about 2 to 3 hours, while the inference just takes a few seconds. Since the training process is an offline process, this running efficiency is enough for real-life bike flow prediction systems.

5 CONCLUSION

In this paper, we propose a new multi-graph convolutional neural network model to predict station-level bike flow in a bike sharing system. The novel aspect is the multi-graph convolution part which utilizes the graph information in flow prediction. More specifically, we design three heterogeneous inter-station graphs to represent a bike sharing system, namely *distance*, *interaction*, and *correlation* graphs; a fusion method is then proposed to conduct the graph convolution operation on the three graphs simultaneously. In the future, we plan to extend the multi-graph model to more scenarios such as subway station crowd flow prediction.

REFERENCES

[1] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.
[2] Di Chai, Leye Wang, and Qiang Yang. Bike flow prediction with multi-graph convolutional networks. *arXiv preprint arXiv:1807.10934*, 2018.
[3] Longbiao Chen, Daqing Zhang, Leye Wang, Dingqi Yang, Xiaojuan Ma, Shijian Li, Zhaohui Wu, Gang Pan, Thi-Mai-Trang Nguyen, and Jérémie Jakubowicz. Dynamic cluster-based over-demand prediction in bike sharing systems. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 841–852. ACM, 2016.
[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3844–3852. Curran Associates, Inc., 2016.
[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
[7] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *international conference on learning representations*, 2018.
[8] 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.
[9] Yongxue Tian and Li Pan. Predicting short-term traffic flow by long short-term memory recurrent neural network. In *Smart City/SocialCom/SustainCom (SmartCity), 2015 IEEE International Conference on*, pages 153–158. IEEE, 2015.
[10] Billy M Williams and Lester A Hoel. Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. *Journal of transportation engineering*, 129(6):664–672, 2003.
[11] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *international joint conference on artificial intelligence*, 2018.
[12] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, pages 1655–1661, 2017.