Bike Flow Prediction with Multi-Graph Convolutional Networks

Di Chai*, Leye Wang*, Qiang Yang
Hong Kong University of Science and Technology, Hong Kong SAR, China
dchai@connect.ust.hk, wangleye@gmail.com, qyang@cse.ust.hk
(*Equal contribution, ordered alphabetically)

ABSTRACT

One fundamental issue in managing bike sharing systems is the bike flow prediction. Due to the hardness of predicting the flow for a single station, recent research works often predict the bike flow at cluster-level. While such studies gain satisfactory prediction accuracy, they cannot directly guide some fine-grained bike sharing 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 leveraging the breakthroughs of deep learning techniques. We propose a new multi-graph convolutional neural network model to predict the bike flow at station-level, where the key novelty is viewing the bike sharing system from the graph perspective. More specifically, we construct multiple inter-station graphs for a bike sharing system. In each graph, nodes are stations, and edges are a certain type of relations between stations. Then, multiple graphs are constructed to reflect heterogeneous relationships (e.g., distance, ride record correlation). Afterward, we fuse the multiple graphs and then apply the convolutional layers on the fused graph to predict station-level future bike flow. In addition to the estimated bike flow value, our model also gives the prediction confidence interval so as to help the bike sharing system managers make decisions. Using New York City and Chicago bike sharing data for experiments, our model can outperform state-of-the-art station-level prediction models by reducing 25.1% and 17.0% of prediction error in New York City and Chicago, respectively.

CCS CONCEPTS

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

KEYWORDS

Graph Convolutional Network, Bike Flow Prediction

ACM Reference format:
Di Chai*, Leye Wang*, Qiang Yang
Hong Kong University of Science and Technology, Hong Kong SAR, China
dchai@connect.ust.hk, wangleye@gmail.com, qyang@cse.ust.hk
(*Equal contribution, ordered alphabetically). 2010. Bike Flow Prediction with Multi-Graph Convolutional Networks. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 9 pages.
DOI: 0000001.0000001

Conference'17, Washington, DC, USA

DOI: 0000001.0000001

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, 14].

In reality, the bike flow of a single station in a city usually has a very complicated dynamic pattern, which makes it hard to predict with traditional statistical learning or machine learning methods [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, 14]. 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 as there is no ground truth, and (ii) the prediction result at cluster-level still cannot directly support the precise management on single stations.

In this paper, we revisit the single station-level bike flow prediction problem in bike sharing systems, which can provide fine-grained information for the system administrators' decision making process and avoid the hard-to-evaluate clustering problem. To achieve this goal, we make effort from two aspects:

(i) **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 [25]. 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 [9] is built to catch the temporal patterns. Hence, both spatial and temporal patterns are effectively captured for station-level bike flow prediction.

(ii) **Compute confidence interval for the prediction**: The demand of single stations sometimes fluctuates a lot in reality. At that time, only providing the estimation value to the bike sharing system managers may not be enough. To this end, our model is designed to further compute the confidence of prediction to help managers make better decisions. More specifically, by leveraging the dropout techniques in neural networks, we simulate various realistic factors affecting the uncertainty of prediction, such as *model uncertainty*, *model misspecification*, and *inherent noise* [34].

Based on these simulations, we can infer the confidence interval for our prediction accurately.

Briefly, this paper has the following contributions:

(i) To the best of our knowledge, this is the first work of leveraging graph convolutional neural networks in to predict station-level bike flow in a bike sharing system, as well as providing prediction confidence estimation.

(ii) We propose a novel *multi-graph convolutional neural network* model to utilize heterogeneous spatial correlations between stations for station-level bike flow prediction. More specifically, three different graphs are constructed for a bike sharing system, i.e., *distance*, *interaction*, and *correlation* graphs. A method to fuse multiple graphs is designed so that graph convolution can be applied on heterogeneous inter-station spatial correlations simultaneously. Then, an encoder-decoder structure with LSTM units is built to capture temporal patterns in historical bike flow records. By properly leveraging dropout techniques, our proposed model can not only calculate the bike flow prediction results, but also the confidence interval at the same time.

(iii) Evaluations on real bike flow dataset in New York City and Chicago shows the effectiveness of our mothod. Compared with the state-of-the-art station-level bike flow prediction models such as LSTM and ARIMA, our multi-graph convolutional neural network model can reduce up to 25.1% prediction error.

2 RELATED WORK

We describe the related work from two perspectives, *bike flow prediction* and *graph convolutional neural networks*.

2.1 Bike Flow Prediction

Flow prediction is a very important topic in bike sharing system. Current studies on bike flow prediction most fall into three categories which are *cluster-based*, *area-based*, and *station-based* flow prediction.

Cluster-based flow prediction: The demand of bike-sharing system is influenced by many factors such as weather, holiday, special events and the influence between stations. To make the prediciton result more accurate, Li et al. group the stations into several clusters using distance and bike usage information. Then they predicted the aggregate demand over all the stations and the proportion for each cluster [14]. Chen et al. used a graph based clustering method to get high internal connectivity, and then predict the over-demand probability of each cluster [3]. Cluster based flow prediction is also used in [26]. Topics about building clusters in bike sharing system are also studied in [1, 5, 16, 21, 33]. While this steam of studies is very popular, the intrinsic difficulty for applying such techniques in real life is the cluster result may not be desired for bike sharing system administration. In most cases, providing station-level prediction is still more practical.

Area-based flow prediction: Unlike the cluster-based flow prediction, area-based methodology focuses on the bike flow of a specific area. One recent way is to conduct grid based map segmentation over the city, and then applies state-of-the-art deep models, such as CNN, ResNet, or ConvLSTM, to predict the flow of each area [30, 31]. But this methodology does not work in single-station prediction because it is hard to decide the size of the area. More than one station will appear in one area if the grid size is large or many grids will contain no station if it is small.

Station-based flow prediction: Compared with the first two types of flow prediction, station-based flow prediction is harder but can provide more fine-grained information in the system operation process. Jon Froehlich et al. compared four simple models in predicting available bikes' number which are *last value*, *historical mean*, *historical trend* and *Bayesian network* [6]. Some researchers adopted time series analysis to predict the future bike demand [22, 28]. Kaltenbrunner et al. used a statistical model to predict the availability of bikes in the future [10]. Compared to these works, we are the first to apply deep learning methods for station-based bike flow prediction, and our experiments show that the performance improvement is significant.

2.2 Graph Convolutional Neural Networks

The graph convolutional neural network was first introduced by Bruna et al. [2], which applies the convolutional layers on the graph data. It is later extended by Defferrard et al. [4] with fast localized convolutions to accelerate the computation process. Kipf et al. proposed an efficient variant of convolutional neural network which can be used directly on graphs, and the network achieved good performance on graph node classification task [12]. Seo et al. proposed a graph convolutional recurrent network which can deal with structured sequence data [17]. The implementation of graph convolutional network is also studied in image classification [27] and the segmentation of point cloud [24, 32]. Two most relevant papers to our work are [13, 29], both applying graph convolutional neural networks to predict traffic speed in road segments. Our work is distinct from them in two aspects. First, [13, 29] only use distance to create a graph for road segments; 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. Second, our model can output prediction confidence interval, which can thus provide more information for the decision making process of bike sharing system organizers.

3 DEFINITIONS AND PROBLEM

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

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 (e.g., the edge's weight can be the reciprocal of distance between two stations). How to construct the graph is one part of our method and will be elaborated in the next section.

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 at the time interval t 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.

4 MULTI-GRAPH CONVOLUTIONAL NEURAL NETWORK MODEL

To solve the above problem, we propose a novel multi-graph convolutional neural network model, which will be elaborated next.

4.1 Framework Overview

Figure 1 gives an overview of our model containing three parts:

(i) Graph Generation: As the grid-based map segmentation [31] does not work when we want to predict the single station's demand, we propose to build inter-station graphs to represent the bike-sharing system, where the links between stations reflect the spatial relationships. More specifically, the nodes in the graph are the bike stations, and the edges represent relationships between stations. We also encode weights on the edges as the relationship strength between stations can be different (e.g., smaller distance between stations may refer to a closer relationship). Moreover, since there may be various relationships between stations that can help our prediction, we construct multiple graphs, such as *distance*, *interaction*, and *correlation*, which will be elaborated later.

(ii) Multi-Graph Convolution: As we construct multiple inter-station graphs to represent one bike sharing system, we introduce a multi-graph convolution part to perform the convolution operation considering all these graphs. More specifically, we first develop a fusion way to incorporate multiple graph information into one fused inter-station graph. Then, we use the graph convolutional layers on the fused graph to encode the graph structure and features of nodes [12]. In our fused bike-sharing graph, graph convolution can extract features of various *spatial relationships* between stations.

(iii) Prediction Network: Based on the graph convolution result, the third step designs a network structure to *predict the bike flow and compute the confidence simultaneously*. More specifically, the first component of the prediction network is an encoder-decoder structure with LSTM (Long-Short Term Memory) units [9], which can learn the hidden representation for each station to catch the *temporal correlations* in the bike flow history. Then, by decomposing the prediction uncertainty into three parts: *model uncertainty*, *model misspecification*, and *inherent noise* [34] (details will be elaborated later), we further estimate the confidence interval of our station-level bike flow prediction, which can provide more information to the managers of bike sharing systems for decision making.

In next a few sections, we will elaborate each part of our multi-graph neural network model in more details.

4.2 Detailed Solution

(i) 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.

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

$$A = \begin{pmatrix} 0 & \frac{1}{dist_{0,1}} & \frac{1}{dist_{0,2}} & \cdots & \frac{1}{dist_{0,N-1}} \\ \frac{1}{dist_{1,0}} & 0 & \cdots & \cdots & \frac{1}{dist_{1,N-1}} \\ \frac{1}{dist_{2,0}} & \frac{1}{dist_{2,1}} & 0 & \cdots & \frac{1}{dist_{2,N-1}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{dist_{N-1,0}} & \cdots & \cdots & \cdots & 0 \end{pmatrix}$$

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:

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

$$A = \begin{pmatrix} d_{0,0} & d_{0,1} & d_{0,2} & \cdots & d_{0,N-1} \\ d_{1,0} & d_{1,1} & d_{1,2} & \cdots & d_{1,N-1} \\ d_{2,0} & d_{2,1} & d_{2,2} & \cdots & d_{2,N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{N-1,0} & \cdots & \cdots & \cdots & d_{N-1,N-1} \end{pmatrix}$$

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:

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

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}$$

$$A = \begin{pmatrix} 0 & r_{0,1} & r_{0,2} & \cdots & r_{0,N-1} \\ r_{1,0} & 0 & r_{1,2} & \cdots & r_{1,N-1} \\ r_{2,0} & r_{2,1} & 0 & \cdots & r_{2,N-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{N-1,0} & \cdots & \cdots & \cdots & 0 \end{pmatrix}$$

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

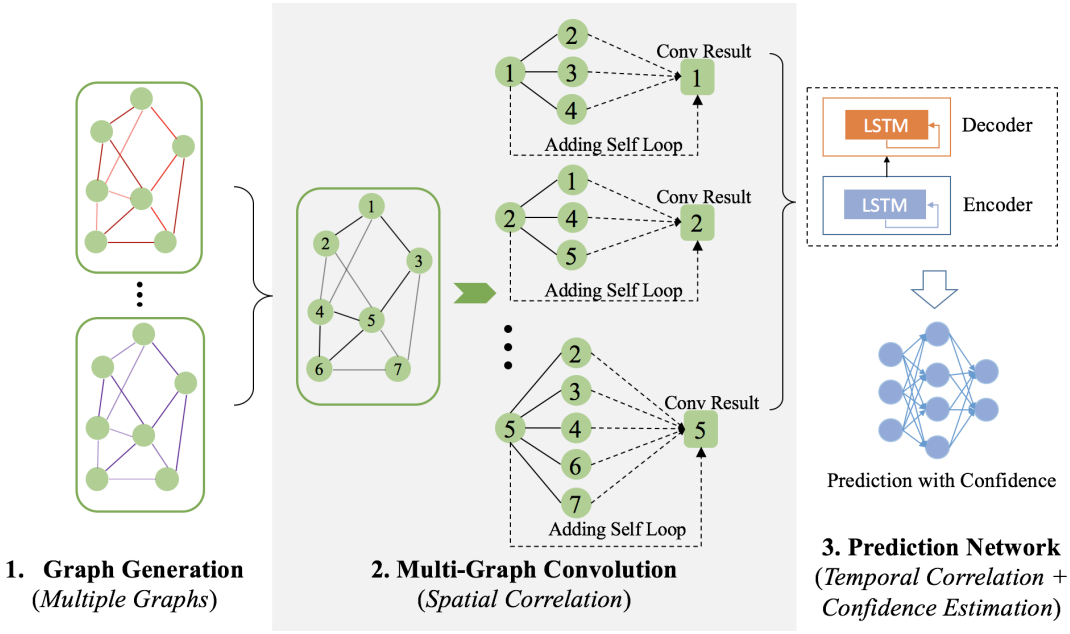

Figure 1: Overview of the Multi-Graph Convolutional Neural Network Model for Bike Flow Prediction

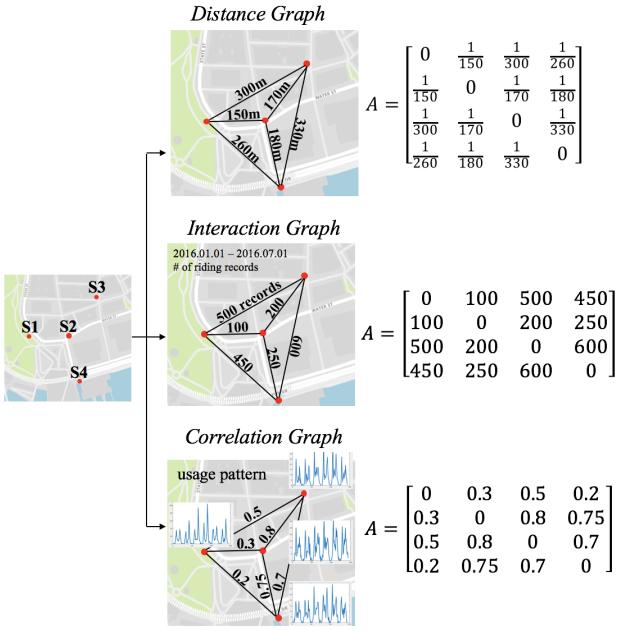

Figure 2: Example of different inter-station graphs.

For readers' understanding, Figure 2 gives a toy example of the above three graph construction methods on four stations.

(ii) 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 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 vary a lot (see Figure 2 for examples), we first normalize the adjacency matrix A for each graph.

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

where D is :

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

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:

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

$$A_i' = D_i^{-1}A_i + I \quad (1 \le i \le N)$$

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

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

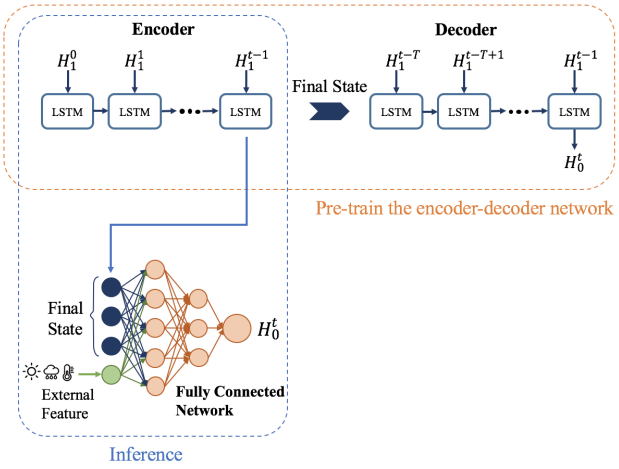

Figure 3: Structure of Prediction Network

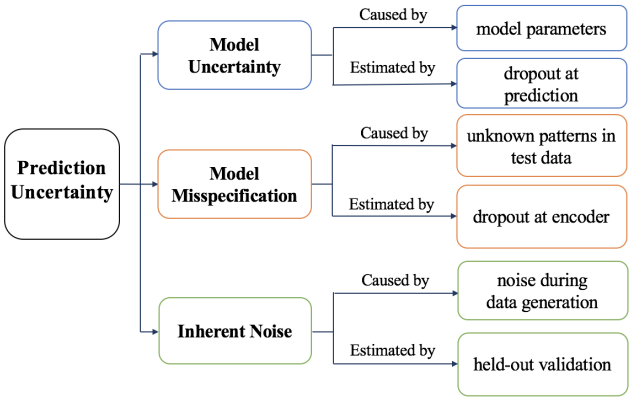

Figure 4: Component of prediction uncertainty

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' (inflow $I^{t'}$ and outflow $O^{t'}$). We take $H_1^{t'}$ as the convolution result, and then use $H_1^{t'}$ as the input of the next prediction network.[2]

The graph convolution operation is performed with the filter matrix W_c over the whole bike flow matrix $H_0^{t'}$ at time t'. It is worth noting that, although the size of the filter W_c is equal to the size of $H_0^{t'}$, it is still (roughly) a local convolution at the corresponding station due to the existence of the inter-station graph F. The reason is that the graph is not fully connected if we build the interaction graph (i.e., two stations are too far from each other to have interacted rides), or most weights of the edge is near zero if we build the distance graph. In a word, many entries in F will be very close to zero. Then, the difficulty of tuning the weight matrix W_c is reduced in the network parameter training process because the initial values of part of W_c will be zero or near zero after multiplied with F.

(iii) Prediction Network

The structure of our prediction network is shown in Figure 3. First, we want to highlight that while multi-graph convolutional layers are able to capture diverse spatial correlations between stations, the temporal patterns in a station's historical ride records have not been exploited yet. Hence, we include the LSTM layers in our neural network model to catch the temporal patterns after the multi-graph convolution. Moreover, we leverage an *encoder-decoder structure* along with LSTM layers for two reasons. First, encoder-decoder structure has been verified very effective in spatio-temporal prediction tasks, and now is one of the most widely used neural network prediction structures [18]. Second, with the encoder-decoder structure, in fact we can also infer the prediction uncertainty, or *confidence interval*, which we will elaborate later.

As shown in Figure 3, the details of the 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 [30]) to a fully connected network (lower part of Figure 3) for predicting the bike flow in the next time H_0^t.

Confidence Estimation: Next, we elaborate how to infer the uncertainty of our prediction, i.e., confidence interval based on the encoder-decoder prediction network. According to literature, the uncertainty of the prediction result can be divided into three parts: *model uncertainty*, *model misspecification*, and *inherent noise* [34] (Figure 4).

(1) *Model uncertainty*, also called epistemic uncertainty, is the variance caused by the uncertainty of the trained model parameters. This uncertainty can often be reduced with the increase of training dataset.
(2) *Model misspecification* appears when the test dataset contains some different patterns from the training dataset. Such uncertainty is more common when the test data is sampled from a different distribution of the training data. In bike flow prediction, this is non-ignorable as in the future, there may happen some special events that have never happened before, thus leading to the model misspecification issue.
(3) *Inherent noise* emerges when the data is originally generated, which is irreducible in practice.

[2]We can stack several graph convolutional layers in our neural network model. In this work, for brevity, we just use one graph convolutional layer, and our experiment shows that even with only one layer, our method can already outperform traditional methods significantly in prediction accuracy.

To estimate the model uncertainty and misspecification, previous works have pointed out that the precise calculation is hard due to the non-linear property of neural networks, while dropout techniques can provide a useful approximation [7, 34]. That is, given an input sequence of the bike flow matrix, during the inference, we perform dropouts in the encoder network (i.e. LSTM units) to get various embeddings for the input (i.e., final state of LSTM units). The variance in such embeddings can represent the model misspecification. At the same time, we also conduct dropouts in the fully-connected prediction network to approximate the model uncertainty. In a word, by performing dropouts in both encoder and fully connected networks for an input, we can estimate the model uncertainty and misspecification by calculating the variance of output prediction results under different dropout trials. Note that the variational dropout is used for LSTM as it has been verified more effective for recurrent neural networks [8]. For the inherent noise, an easy way to estimate it is using a held-out validation dataset to calculate the prediction error variance, which can be proved to be an asymptotically unbiased estimation on the inherent noise level [34].

Finally, after obtaining the prediction variance σ_1 incurred by the model uncertainty and model misspecification with dropout, and variance σ_2 incurred by inherent noise with a held-out validation dataset, we can infer the confidence interval as:

$$[y^* - Z_{\frac{\alpha}{2}} * \sqrt{\sigma_1^2 + \sigma_2^2}, y^* + Z_{\frac{\alpha}{2}} * \sqrt{\sigma_1^2 + \sigma_2^2}]$$

where y^* is the prediction result for a certain station at a time slot, and $Z_{\frac{\alpha}{2}}$ is the $(1-\alpha)$ confidence interval of standard normal distribution.

5 EVALUATION

In this section, we evaluate our multi-graph convolutional network method with real bike sharing datasets. We will first introduce the dataset, experiment settings, and then illustrate our experiments results comprehensively.

5.1 Experiment Setting

Datasets: We used bike flow dataset collected from New York City and Chicago[3]. The datasets cover a four year time period. All the data are in the forms of riding record containing start station, start time, stop station, stop time and so on. We summarize the dataset statistics in table 1. Weather data comes from the NCEI website[4] (National Centers for Environmenttal Information).

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.

Network Implementation and Parameters: The encoder and decoder implemented 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% [11].

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

Table 1: Dataset statistics

	New York City	Chicago
Time span	2013.07-2017.09	2013.06-2017.12
#Riding records	49,669,470	13,826,196
#Stations	827	586

Table 2: 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*
ARIMA	11.329	9.545	8.049
SARIMA	8.677	7.363	6.521
LSTM	6.802	5.981	5.345
Multi-graph	**4.745**	**4.473**	**4.003**

Chicago

	Top 5 stations	*Top 10 stations*	*Average*
ARIMA	9.853	8.535	6.163
SARIMA	6.797	6.175	4.608
LSTM	6.231	5.853	4.405
Multi-graph	**5.177**	**4.930**	**3.658**

In the encoder-decoder structure of the prediction network, we set $T = 3$ in the decoder (refer to Figure 3). We use the past 6-hour history data to predict the bike flow in the next one hour. For the confidence computation, we compute the 95% confidence interval for the prediction result. For brevity, we only report the prediction results of bike inflow in this section, while the bike outflow results are very similar. Note that all of the above network implementation and parameter settings are chosen as they can perform well on the 40-day validation data, while the reported results in the next subsections are based on the 80-day test data.

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

- *ARIMA* [25]: Auto-Regressive Integrated Moving Average is a widely used time series prediction model.
- *SARIMA* [25]: The seasonal version of ARIMA.
- *LSTM* [19]: Recent studies in traffic flow prediction, such as [19], adopted the long short-term memory (LSTM) recurrent neural network model and verified its effectiveness.

5.2 Experiment Results

Prediction error: We use RMSE (root mean square error) to measure the prediction error. Table 2 shows our evaluation results. In general, among various baselines, LSTM can perform the best. However, our multi-graph convolutional method can still beat LSTM significantly by reducing the average prediction error by 25.1% and 17.0% 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

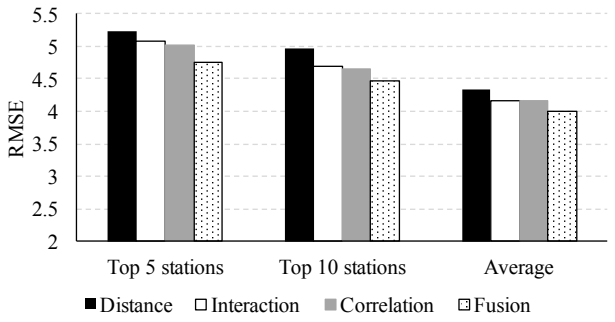

Figure 5: Comparison of multi-graph and single-graph convolutional models (New York City).

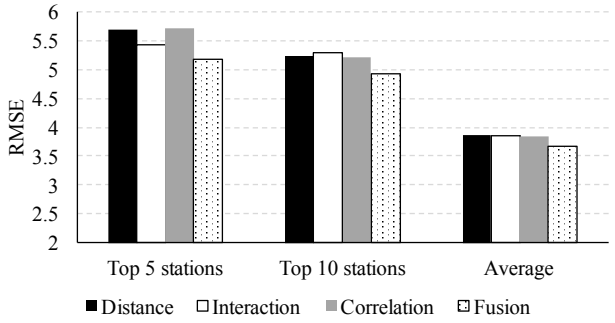

Figure 6: Comparison of multi-graph and single-graph convolutional models (Chicago).

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 5 and 6 show the results when we only use a single graph (*distance*, *interaction* or *correlation* graph) for prediction in New York City and Chicago. 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%.

Among the single-graph methods, we can observe that which graph model performs better is dependent on the test stations. For example, for the top 5 busy stations in Chicago, the best single-graph convolutional model is 'interaction graph'. However, when we evaluate on the top 10 busy stations, it performs worse than the other two single graphs, 'distance graph' and 'correlation graph'. So, if we use the single-graph method, how to choose the graph would be very hard and tricky. In comparison, with our proposed multi-graph method, we do not need to bother selecting which single graph representation of the bike sharing system. Our proposed fusion method can automatically and adaptively extract useful information from all of the single graphs and then achieve better prediction performance.

Table 3: The ratio of actual bike flow values falling into the estimated confidence interval (Chicago).

Method	*Confidence*
dropout (model uncertainty + misspecification)	0.486
validation variance (inherent noise)	0.869
our method	0.933

Confidence interval estimation: To evaluate whether our confidence interval estimation is accurate, we calculate the ratio that the real bike flow values fall into our estimated confidence interval. If the ratio is close to 95%, then it means that the estimation is accurate. Table 3 shows the result of the confidence interval computation. In addition to our method which considers three components of uncertainty (*model uncertainty*, *model misspecification*, and *inherent noise*), we also test the confidence estimation result if we only use dropout or validation variance. We find that our method can achieve the closest value toward 95%, verifying that both dropout and validation variance are effective in the uncertainty estimation.

In reality, the confidence interval estimation can provide richer information than only a value estimation. For example, if a station usually has a larger confidence interval in flow prediction, it implies that the station may be intrinsically hard to predict. Then, by studying the stations with larger confidence intervals, we may be able to identify key factors impacting the station predictability, which might further guide us to design a more effective prediction model. Take the two stations in Figure 7 as an example, the two stations have similar daily inflow but perform quite different in the estimated confidence intervals. More specifically, the station near school has a very complicated usage pattern, leading to a large estimated confidence interval; in comparison, the station near residential area performs more regularly, even though it has a peak usage pattern around 8:00 a.m. From this result, we may infer that there is much more space for the station near school to improve its prediction accuracy. Then, the bike sharing system manager can devote more efforts to such hard-to-predict stations and explore whether more factors can be incorporated into the prediction model to increase these stations' prediction performance.

Tuning the training data length: In our method, the training data has two major roles: (i) building the inter-station graphs such as interaction and correlation graphs; (ii) training the whole multi-graph neural network model parameters. Hence, if the training data length is too short, the prediction results may not be satisfactory.

To test how long training data is needed to achieve a good performance, we vary the length of training data and see how the prediction error changes. Figure 8 shows the results in Chicago. We can find that if the training data length is shorter than 6 months, our model does not perform very well. By increasing the length of

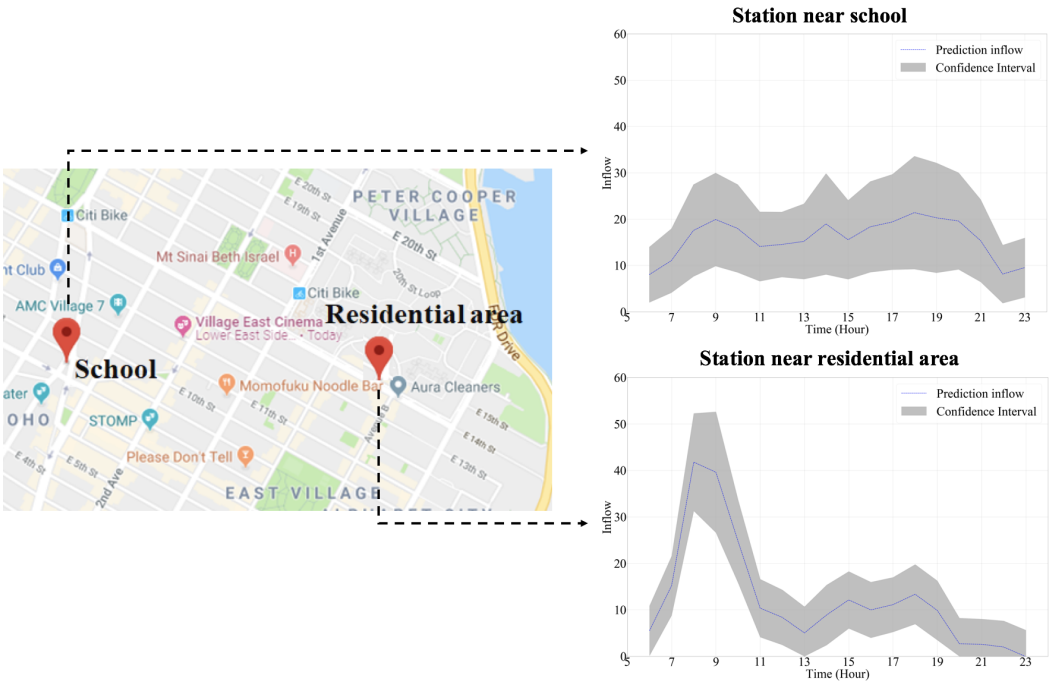

Figure 7: Case study of two bike stations with different estimated confidence intervals (New York City).

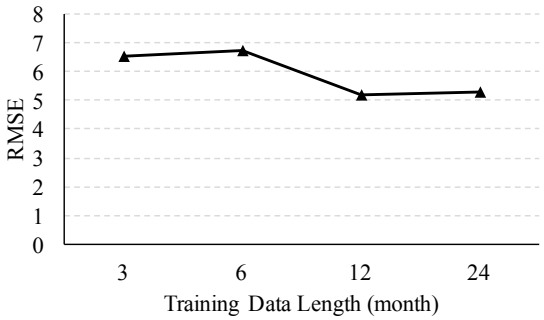

Figure 8: Tuning the length of training data (Chicago).

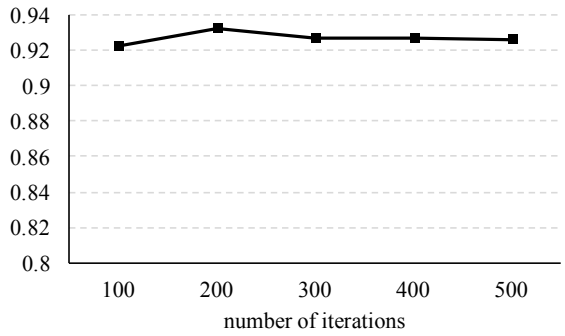

Figure 9: Actual coverage ratio of the estimated confidence interval by varying the number of dropout iterations (Chicago).

training data beyond 12 months, the prediction error reduces significantly. With these results, we suggest that for a robust prediction accuracy, the training data is desired to be last for at least one year. While some cities may not have enough historical records if they just start the bike sharing systems, we plan to study how to address the cold-starting problem in the future work, e.g., with cross-city knowledge transfer learning methods [23].

Tuning the number of dropout iterations: When calculating the confidence interval, we need to simulate several iterations of dropouts so as to estimate the model uncertainty and model misspecification. Here, we test which number of iterations is needed for obtaining a robust confidence interval estimation. As shown in Figure 9, we can find that the coverage ratio of actual bike flow values by our estimated confidence does not change significantly when we conduct 100 to 500 iterations, especially after 300 iterations. Hence, we think that several hundred of iterations should be enough for the confidence interval estimation of the station-level bike flow prediction.

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.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a new multi-graph convolutional neural network model to predict station-level bike flow in a bike sharing system. There are two novel aspects of our model. The first 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. The second aspect is the uncertainty computation part that is able to infer the confidence interval for our prediction. The confidence interval estimation employs the dropout technique to obtain a robust estimation interval.

As the pioneering effort to employ the graph convolutional networks on bike flow prediction, there are still many issues to investigate in the future.

Extending usage scenarios. There are many other urban traffic systems similar to bike sharing, such as subway. We are now working on extending our bike flow prediction model to a more general urban traffic prediction methodology.

Anomaly detection. With confidence interval estimation, another important usage is the anomaly detection. That is, if we detect an irregular large uncertainty for a station at some time slots (compared to average), then it is probably that some abnormal events happen around the station. We will test how such anomaly detection works in real-life bike sharing systems.

Addressing cold-start problems. As shown in our experiments, our current model needs more than one-year historical bike flow records to obtain a good prediction accuracy. One of the important future issues is to reduce the length of required training data length, so as to address the cold-start problem of the bike flow prediction.

Improving network structure. In this work, we use LSTM as the basic units to capture temporal patterns in bike flow. Very recent studies [15, 20] have indicated that attention units may be potentially more effective. Hence, we will study whether replacing LSTM with attention can further boost the prediction performance.

REFERENCES

[1] Martin Zaltz Austwick, Oliver OfiBrien, Emanuele Strano, and Matheus Viana. The structure of spatial networks and communities in bicycle sharing systems. *PloS one*, 8(9):e74685, 2013.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[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] Côme Etienne and Oukhellou Latifa. Model-based count series clustering for bike sharing system usage mining: a case study with the vélibfisystem of paris. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):39, 2014.

[6] Jon Froehlich, Joachim Neumann, Nuria Oliver, et al. Sensing and predicting the pulse of the city through shared bicycling. In *IJCAI*, volume 9, pages 1420–1426, 2009.

[7] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[8] Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*, pages 1019–1027, 2016.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[10] Andreas Kaltenbrunner, Rodrigo Meza, Jens Grivolla, Joan Codina, and Rafael Banchs. Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system. *Pervasive and Mobile Computing*, 6(4):455–466, 2010.

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[13] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *international conference on learning representations*, 2018.

[14] 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.

[15] Yuxuan Liang, Songyu Ke, Junbo Zhang, Xiuwen Yi, and Yu Zheng. Geoman: Multi-level attention networks for geo-sensory time series prediction. In *IJCAI*, 2018.

[16] Jasper Schuijbroek, Robert C Hampshire, and W-J Van Hoeve. Inventory rebalancing and vehicle routing in bike sharing systems. *European Journal of Operational Research*, 257(3):992–1004, 2017.

[17] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks. *arXiv preprint arXiv:1612.07659*, 2016.

[18] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[19] 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.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[21] Patrick Vogel, Torsten Greiser, and Dirk Christian Mattfeld. Understanding bike-sharing systems using data mining: Exploring activity patterns. *Procedia-Social and Behavioral Sciences*, 20:514–523, 2011.

[22] Patrick Vogel and Dirk C Mattfeld. Strategic and operational planning of bike-sharing systems by data mining–a case study. In *International Conference on Computational Logistics*, pages 127–141. Springer, 2011.

[23] Leye Wang, Xu Geng, Xiaojuan Ma, Feng Liu, and Qiang Yang. Crowd flow prediction by deep spatio-temporal transfer learning. *arXiv preprint arXiv:1802.00386*, 2018.

[24] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

[25] 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.

[26] Haitao Xu, Jing Ying, Hao Wu, and Fei Lin. Public bicycle traffic flow prediction based on a hybrid model. *Applied Mathematics & Information Sciences*, 7(2):667, 2013.

[27] Li Yi, Hao Su, Xingwen Guo, and Leonidas Guibas. Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] Ji Won Yoon, Fabio Pinelli, and Francesco Calabrese. Cityride: a predictive bike sharing journey advisor. In *Mobile Data Management (MDM), 2012 IEEE 13th International Conference on*, pages 306–311. IEEE, 2012.

[29] 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.

[30] Junbo Zhang, Yu Zheng, and Dekang Qi. Deep spatio-temporal residual networks for citywide crowd flows prediction. In *AAAI*, pages 1655–1661, 2017.

[31] 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.

[32] Feng Zhou and Yuanqing Lin. Fine-grained image classification by exploring bipartite-graph labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1124–1133, 2016.

[33] Xiaolu Zhou. Understanding spatiotemporal patterns of biking behavior by analyzing massive bike sharing data in chicago. *PloS one*, 10(10):e0137922, 2015.

[34] Lingxue Zhu and Nikolay Laptev. Deep and confident prediction for time series at uber. In *IEEE ICDM Workshops (ICDMW)*, pages 103–110. IEEE, 2017.