A Data-Driven Spatial-Temporal Graph Neural Network for Docked Bike Prediction

Guanyao Li*, Xiaofeng Wang†, Gunarto Sindoro Njoo†, Shuhan Zhong*,
S.-H. Gary Chan*, Chih-Chieh Hung§, Wen-Chih Peng†
*Department of Computer Science and Engineering, The Hong Kong University of Science and Technology
†Department of Computer Science, National Yang Ming Chiao Tung University
§Department of Computer Science and Engineering, National Chung Hsing University
gliaw@cse.ust.hk, wangxiao@cs.nctu.edu.tw, gunarto.cs01g@g2.nctu.edu.tw, {szhongaj,gchan}@cse.ust.hk,
smalloshin@nchu.edu.tw, wcpeng@g2.nctu.edu.tw

***Abstract*—Docked bike systems have been widely deployed in many cities around the world. To the service provider, predicting the demand and supply of bikes at any station is crucial to offering the best service quality. The docked bike prediction problem is highly challenging because of the complicated joint spatial-temporal (ST) dependency as bikes are picked up and dropped off, the so-called "flows", between stations. Prior works often considered the spatial and temporal dependencies separately using sequential network models, and based on locality assumptions. Without sufficiently capturing the joint spatial and temporal features, these approaches are not optimal for attaining the best prediction accuracy.**

We propose STGNN-DJD, a novel data-driven Spatial-Temporal Graph Neural Network to solve the bike demand and supply prediction problem by unifiedly embedding the Dynamic and Joint ST Dependency in two novel ST graphs. Given station locations and historical rental data on bike flow over the past time slots 0 to $t-1$, we seek to predict online the bike demand and supply at any station at time t. To extract joint spatial-temporal dependency, STGNN-DJD employs a graph generator to construct, at the beginning of time t, two graphs which embed the flow relationships between stations at various time slots (flow-convoluted graph) and dynamic demand-supply pattern correlation between stations (pattern correlation graph), respectively. Given the two spatial-temporal graphs, STGNN-DJD subsequently employs a graph neural network with novel flow-based and attention-based aggregators to generate embedding of each station for docked bike prediction. We have conducted extensive experiments on two large bike-sharing datasets. Our results confirm the effectiveness of STGNN-DJD as compared with other state-of-the-art approaches, with significant improvement on RMSE and MAE (by $20\%-50\%$). We also provide a case study on dynamic dependencies between stations, and demonstrate that the locality assumption does not always hold for a docked bike system.

***Index Terms*—bike demand and supply prediction; spatial-temporal data prediction; spatial-temporal data management.**

I. Introduction

Dock-based bike sharing, widely deployed in many cities such as Chicago, Los Angeles, London and Shanghai, offers a cost-effective means for last-mile solutions, promotes a healthy lifestyle, and eases the growing traffic congestion and environmental concerns. The global bike-sharing service market is estimated to grow to over 13.7 billion U.S. dollars by 2026 from 3.3 billion U.S. dollars in 2020 [1].

A dock-based bike-sharing system consists of fixed stations distributed around the city. A user can pick up a bike at any station, and end the ride by returning the rented bike to any station in the city. To the service provider, bike shortage at a station means revenue loss. On the other hand, due to finite docking capacity at a station, bike over-supply at one station means that users have to return their bikes to some other stations, leading to inconvenience and reduction in service quality. It is hence in the provider's interest to predict the demand and supply of docked bikes at stations (so that bikes can be dispatched in advance to meet the demand and supply). Such docked bike prediction also has immense value in bike lane planning [2] and route recommendation [3].

In this work, we consider predicting docked bikes, i.e. their demand and supply, at any station from a service provider point of view. Time is slotted, with a certain slot duration of, say, 10 to 20 minutes. Historical customer rental data for bike pickup time (demand) and drop-off time (supply) between any pair of stations, the so-called "flow" between the stations, are available over the past t slots labelled as $0, 1, 2 \ldots, t-1$ (e.g., $t = 500$ or 1,000). With the data, the provider would like to make online predictions of docked bikes at any station for the next slot, i.e., at time t.

Predicting bike demand and supply at a station is a challenging problem due to complicated joint spatial-temporal dependency between stations. This is because the supply of a station is related to the bike demand of another station with some temporal lag due to the travel time between stations. Moreover, stations with similar demand-supply patterns are significantly correlated. For example, close stations may have similar patterns because of the locality effect, and stations near two remote schools may also share similar patterns because the schools have similar operating hours. Such correlation between stations may be time-varying, which means that the correlation between stations could differ greatly based on different temporal signatures. To achieve high prediction accuracy, the model needs to appropriately capture the joint spatial-temporal dependency between stations regarding their flow relationship and demand-supply patterns.

Although much work has attempted to capture spatial and

temporal dependencies to predict docked bike demand and supply, most considers them separately with independent modules [4]–[7]. Regarding spatial dependency, prior formulations, no matters whether based on non-overlapping grids or spatial graphs to model station locations, often apply convolution approaches to address it. For temporal dependency, they use Recurrent Neural Networks (RNNs) and its variants such as Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). While commendable, these works consider the two dependencies sequentially in a decoupled manner, hence failing to capture the *joint* spatial-temporal nature of our current setting.

Moreover, existing approaches often solely assume a locality effect in their study (i.e., local dependency) by overlooking the dependency on distant stations (i.e., global dependency) [8]–[11]. While these approaches are applicable in some settings, they may not work well for the bike-sharing case where users are not likely to bike between two nearby stations, and two remote stations may have similar demand-supply patterns. Our empirical evaluations in Section VIII also present counter intuitive results regarding the locality effect. In reality, a better approach is to learn the relationship between distance and dependency in a data-driven manner instead of relying on prior assumptions.

To solve the aforementioned problems, we propose STGNN-DJD, a novel, effective, and data-driven **S**patial-**T**emporal **G**raph **N**eural **N**etwork to capture the **D**ynamic and **J**oint spatial-temporal **D**ependency between stations for demand and supply prediction at any docking station. Given historical data on flow between stations up to time $t-1$, STGNN-DJD predicts the docked bikes at any station at time t.

Our solution first uses flow convolution to extract spatial-temporal features for any individual station, and generate two novel and effective graphs to encode the dependency between stations in terms of direct flow between stations as well as the correlation of their flow time-series. After that, it employs two novel aggregators to capture the dynamic and joint spatial-temporal dependency in the flow-convoluted graph and the pattern correlation graph, without assuming any relationship between distance and traffic. As both spatial and temporal information is encoded by the unified graph structure, our solution captures the spatial and temporal dependency jointly rather than independently. To summarize, we make the following contributions in this paper:

- *A generator to construct time-dependent graphs encoding the dynamic and joint spatial-temporal dependency:* With a novel graph generator, STGNN-DJD first constructs two graphs which embed the spatial and temporal information between stations. These spatial-temporal graphs, with stations as nodes, are generated at the beginning of time slot t so as to make prediction at the slot. The first one is called the flow-convoluted graph (FCG), which captures the features of bike exchanges between stations. The second one is called the pattern correlation graph (PCG), which accounts for pattern correlation between stations by relating the features of demand and supply patterns at each station with the other (there may not be physical bike flows between the two stations). FCG and PCG co-exist to solve both the local and global dependency between stations. In both graphs, the node features and edges are time-varying, generated by the data of the past t slots, in order to encode the dynamic spatial-temporal dependency between stations.
- *A graph neural network with novel aggregators to process spatial-temporal graphs for prediction:* We propose a novel spatial-temporal graph neural network (GNN) to process FCG and PCG. In particular, we design a flow-based aggregator to learn the dependency in FCG, and propose a data-driven attention-based aggregator to learn inter-station dependency in PCG without assuming any relationship between distance and traffic. The learned representation of stations is then fed into a demand-supply neural network predictor for prediction.
- *Extensive experiments and a case study to validate the effectiveness of STGNN-DJD:* We evaluate the performance of STGNN-DJD using two large-scale bike-sharing datasets collected in Chicago and Los Angeles. Our results demonstrate the effectiveness of STGNN-DJD to capture jointly spatial-temporal dependency for accurate prediction on docked bikes. STGNN-DJD significantly outperforms the state-of-the-art models (by $20\% - 50\%$ on RMSE and MAE). We also conduct a case study on the relationship between inter-station dependency and distance. The study shows that close stations do not always mean higher dependency than distant ones, and vice versa, and distant stations may have high impact on bike demand and supply prediction.

The remainder of this paper is organized as follows. We present related works in Section II, followed by our problem formulation and system overview in Section III. We present the spatial-temporal graph generation in Section IV. Then we introduce the spatial-temporal dependency learning and demand-supply predictor in Sections V and VI, respectively. We present experimental settings and results in Section VII, and a case study in Section VIII. We conclude in Section IX.

II. Related Works

Bike-sharing plays an important role in public transportation systems, and predicting demand and supply for bike-sharing systems has hence attracted much attention in recent years. Depending on the spatial granularity of the prediction task, bike demand and supply prediction can be separated into three levels: cluster-based, grid-based, and station-based.

Cluster-based works group neighboring stations into a few clusters based on geographical locations, historical transition or usage patterns, and then predict the demand and supply for each cluster [12], [13]. In particular, a multi-similarity-based inference model with a Gradient Boosting Regression Tree (GBRT) is used for prediction in the work [12], while Monte Carlo simulation is adopted in WCN-MC [13]. As for grid-based works, they divide a city map into equal-sized and

non-overlapping grids, and the traffic (e.g., bike demand) of each grid is predicted [8], [9], [14]–[16]. A city map with girds is seen as an image with pixels, and convolution neural networks are used in these works to capture the correlations among grids for prediction. While impressive, both cluster-based and grid-based works can hardly be extended to the scenario of demand and supply prediction for docked bike stations because the docking stations are not uniformly distributed in the grids of a city.

In contrast to the cluster-based and grid-based approaches, station-based approaches predict the demand and supply for individual stations. Earlier works employ traditional time series analysis approaches, such as ARIMA and its variants to predict bike demand [17], [18]. However, these approaches solely consider the temporal dependency on historical data, but ignore the importance of dependency among stations, which we believe is an essential aspect of bike-sharing systems. Some works employ machine learning algorithms to predict the demand and supply based on manually defined features. They first extract features from historical flow data and other external data such as POIs and weather, and then predict the demand and supply using machine learning algorithms, such as linear regression [19], K-Nearest-Neighbor regression [20], Support Vector Machine [21], and Bayesian model [22]. Nevertheless, these works highly rely on manually defined features, and they cannot capture the joint spatial-temporal dependency among stations.

In recent years, deep learning has seen rapid development [23] and it has been applied to capture the spatial and temporal dependencies between stations to predict the demand and supply of bike stations. Most of them consider the spatial and temporal dependencies separately. In terms of spatial dependency, some recent works started to use graph structures to embed the spatial dependency between stations and learn latent representation for stations using graph neural networks, such as graph convolution networks [24] and graph attention networks [25]. Most of these works assume that spatially closer stations have similar demand-supply patterns and have stronger dependency than distant stations. They construct the graphs based on the distance of stations or road networks [5]–[7], [10], [11], [26]–[29]. However, these works are inclined to capture influence from nearby stations but overlook those distant stations. Other works use Transformer with self-attention [30] to capture the dependency among stations [31]–[33]. Nevertheless, they do not consider the flow relationships between stations, which we believe is a significant indicator of inter-station dependency but is not well considered in these works. Compared with existing works on spatial dependency modelling, STGNN-DJD considers dependency between stations regarding flow relationships and demand-supply correlations. It generates a flow-convoluted graph to embed the flow features between stations and a demand-supply correlation graph to embed the demand-supply pattern correlations between stations. In particular, STGNN-DJD employs a data-driven attention approach to learn the dependency between stations, without assuming any relationship between distance and traffic.

Another important aspect for docked bike prediction is the temporal dependency. Flow between stations highly depends on the time-of-the-day and the-day-of-the-week due to human periodic mobility behavior [34], [35]. To this end, most existing works employ RNNs and their variants such as LSTM and GRU to consider the temporal dependency [4], [14], [26], [36]–[39]. However, RNN-based approaches are difficult to capture the dependency between distant positions in a sequence due to the vanishing or exploding gradient problem. To address these issues, we propose a flow convolution approach based on the 1×1 convolution kernel. The 1×1 convolution kernel has been used for cross channel pooling in Network-in-Network [40] and dimension reduction in GoogLeNet [41]. Compared with RNN-based models, our flow convolution approach is effective for capturing long-term dependency.

While commendable, the existing works separately considered spatial and temporal dependency in a decoupled manner, and they failed to capture the joint spatial-temporal dependency among stations. On the other hand, considering the spatial and temporal dependencies in a joint manner has attracted much attention in recent years. Similar to STSGCN [42], our proposed STGNN-DJD generates spatial-temporal graphs to embed both spatial and temporal information in the unified structure so that it can capture the joint spatial-temporal dependency between stations for prediction. Nevertheless, STSGCN focuses on capturing the locality spatial-temporal correlation while STGNN-DJD has the capacity to consider both locality and global spatial-temporal dependency.

III. Problem Formulation and System Overview

A. Problem Formulation

In a bike-sharing system, a trip record is denoted as $\{r_{id}, s_o, s_d, t_s, t_e\}$, which contains the following information: (1) a trip ID r_{id}, (2) an origin station s_o, (3) a destination station s_d, (4) a start time t_s, and (5) an end time t_e.

Let $S = \{s_1, s_2, \ldots, s_n\}$ be the set of bike stations in a city, where $s_i = (lon_i, lat_i)$ is a station whose longitude and latitude are (lon_i, lat_i). Users can borrow or return bikes at any station in the city.

To describe the traffic between stations at a time slot t, let $I^t \in \mathbb{R}^{n \times n}$ and $O^t \in \mathbb{R}^{n \times n}$ be the inflow and outflow matrices, where n is the number of stations. $I^t_{i,j} \in I^t$ is the number of bikes borrowed from station s_j and returned to s_i at time t, where t is the returning time. $O^t_{i,j} \in O^t$ refers to the number of bike checked out from station s_i at t and returned to station s_j, where t is the checkout time. Correspondingly, each row in the inflow and outflow matrices represents a station's inflow from and outflow to other stations at a time slot t, denoted as $I^t_i \in \mathbb{R}^{1 \times n}$ and $O^t_i \in \mathbb{R}^{1 \times n}$, respectively. Consider that a user borrows a bike from a station s_i at t_i and returns it to station s_j at t_j. Then $O^{t_i}_{i,j}$ and $I^{t_j}_{j,i}$ will be both increased by 1. Note that because t_i and t_j may not refer to the same time slot, $I^{t^T} = O^t$ does not hold.

Based on the above definitions, the docked bike demand and supply prediction problem is formulated as below:

Definition 1: (Docked bike demand and supply prediction problem) Given a set of bike stations S, and the historical inflow and outflow matrices until time $t-1$, $\{I^0, ..., I^{t-1}\}$ and $\{O^0, ..., O^{t-1}\}$, we would like to predict the bike demand x_i^t and supply y_i^t for any individual station s_i at the time slot t, where $x_i^t = \sum_{j=0}^{n} O_{i,j}^t$ and $y_i^t = \sum_{j=0}^{n} I_{i,j}^t$.

B. STGNN-DJD Overview

Figure 1 overviews the process of STGNN-DJD, which consists of the following three essential components:

- Spatial-temporal graph generation: Given the historical flow data between stations until $t-1$, STGNN-DJD generates two spatial-temporal graphs, namely the flow-convoluted graph and the pattern correlation graph, to represent the spatial-temporal dependency between stations in terms of their flow dependency and demand-supply pattern correlation respectively. To this end, STGNN-DJD first uses a flow convolution to learn time-dependent spatial-temporal features for stations from their historical flow data. Based on the station features, it generates a *flow-convoluted graph*, in which edges are generated according to the flow convolution result. However, stations with similar patterns may not have physical flow between them. To address the issue, STGNN-DJD employs an attention mechanism to generate a *pattern correlation graph*, which accounts for pattern correlation between stations by relating the features of demand and supply patterns at each station with the other.
- Spatial-temporal dependency learning: Based on the generated spatial-temporal graphs, STGNN-DJD uses a graph neural network to learn spatial-temporal embedding for each node (i.e., station) by aggregating information from other nodes in the graphs. The spatial-temporal dependencies between stations are captured via the aggregation process of GNNs. Because flow between stations explictly indicates their dependency level, STGNN-DJD employs a flow-based aggregator to measure the dependency in terms of flow in the flow-convoluted graph. For dependency learning in the pattern correlation graph, STGNN-DJD uses a data-driven attention-based aggregator to automatically learn the dependency, without any assumptions of the relationship between distance and dependency. Moreover, a multi-head attention mechanism is used in the pattern correlation graph to improve the generalization of the model and capture various dependencies.
- Demand-supply predictor: Given the spatial-temporal embedding of stations, STGNN-DJD simultaneously predicts bike demand and supply of any individual station at time t using fully connected neural networks.

The details of each component will be elaborated in Sections IV, V, and VI, respectively.

IV. Spatial-temporal Graph Generation

Graph generation is fundamental to the success of a GNN-based model. If the generated graph cannot effectively capture the relationships between nodes (i.e., stations in our work), it may degrade the prediction performance [7], [36].

In this work, we consider the spatial-temporal dependencies between stations in terms of their flow dependency and demand-supply pattern correlation. To this end, we propose a graph generator to construct a flow-convoluted graph (FCG) and a pattern correlation graph (PCG) encoding the dynamic and joint spatial-temporal dependency in this section. First, we propose a flow convolution approach to learn spatial-temporal features for each node from its historical flow data (Section IV-A). Subsequently, we introduce the edge generation for FCG and PCG in Section IV-B.

A. Node Feature Learning

Historical flow data reveal the spatial-temporal dependency over time. Intuitively, bike demand and supply are affected by the most recent past time slots' flow, termed as short-term dependency. Meanwhile, the demand and supply have a significant daily periodic dependency (long-term dependency), as usually seen in the time series data with periodic movement.

Inspired by the success of 1×1 convolution in fusing information from different channels in computer vision tasks [40], we propose a flow convolution approach with 1×1 convolution kernels to capture dependency from different time slots. As shown in Figure 2, stations' inflow from or outflow to other stations at different time slots is represented as tensors with multiple channels. Then we use 1×1 convolution kernels to capture short-term and long-term dependency from different channels.

We use the inflow and outflow matrices in the past k time slots to capture the short-term dependency, namely $\{I^{t-k}, \dots I^{t-2}, I^{t-1}\}$ and $\{O^{t-k}, \dots O^{t-2}, O^{t-1}\}$. We first concatenate them along the temporal dimension respectively, to obtain two tensors $\mathbb{I}^S \in \mathbb{R}^{k \times n \times n}$ and $\mathbb{O}^S \in \mathbb{R}^{k \times n \times n}$. For each station, its inflow/outflow over time can be seen as a $1 \times n$ matrix with k channels, where n is the number of stations. We apply 1×1 convolution kernels on their inflow/outflow matrices to integrate the flow information at different time slots, and obtain their short-term temporal embedding:

$$\hat{\mathbb{I}}^S = \sigma_1(W_1 * \mathbb{I}^S + b_1), \tag{1}$$

$$\hat{\mathbb{O}}^S = \sigma_1(W_2 * \mathbb{O}^S + b_2), \tag{2}$$

where $W_1 \in \mathbb{R}^{1 \times k}$, $W_2 \in \mathbb{R}^{1 \times k}$, $b_1 \in \mathbb{R}^{n \times n}$ and $b_2 \in \mathbb{R}^{n \times n}$ are learnable parameters, $*$ represents the convolution operator, and $\sigma_1(\cdot)$ is the ReLU activation function. $\hat{\mathbb{I}}^S \in \mathbb{R}^{n \times n}$ and $\hat{\mathbb{O}}^S \in \mathbb{R}^{n \times n}$ are the inflow and outflow short-term embedding respectively.

To consider the long-term dependency, the inflow and outflow matrices of the same time slot in the past d days are used: $\mathbb{I}^L = \{I^{t-d \times day}, \dots I^{t-1 \times day}\}$ and $\mathbb{O}^L = \{O^{t-d \times day}, \dots O^{t-1 \times day}\}$, where $\mathbb{I}^L \in \mathbb{R}^{d \times n \times n}$ and $\mathbb{O}^L \in \mathbb{R}^{d \times n \times n}$ denote the inflow and outflow matrices in the past d days. The long-term temporal dependency is then captured using the 1×1 convolution kernel:

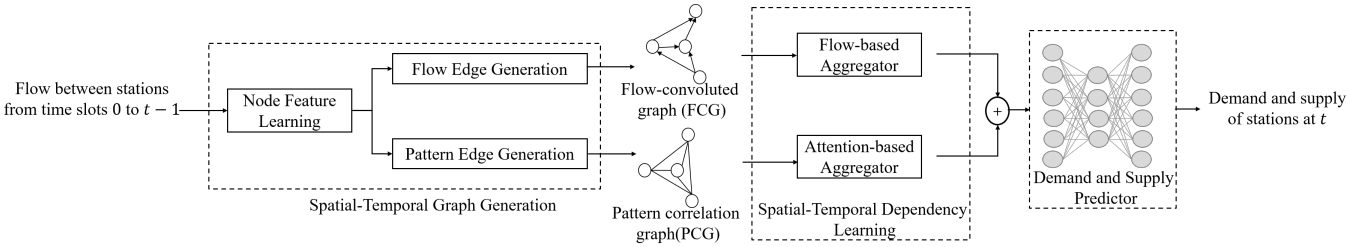


Fig. 1. Overview of STGNN-DJD.

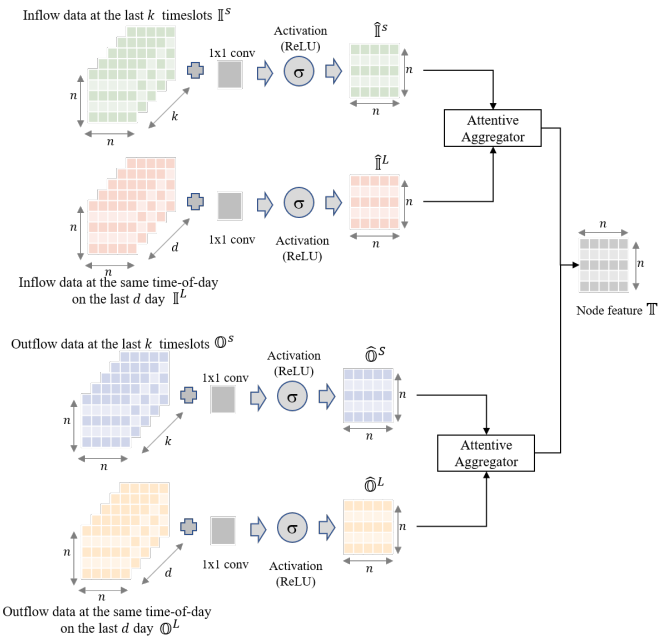


Fig. 2. Illustration of flow convolution. We apply 1×1 convolution kernel on the historical inflow/outflow data, and fuse the short-term and long-term dependency using an attentive aggregator. We consider the last k time slots (same day) for short-term dependency, and the last d days (same time-of-day) for long-term dependency.

$$\hat{\mathbb{I}}^L = \sigma_1(W_3 * \mathbb{I}^L + b_3), \tag{3}$$

$$\hat{\mathbb{O}}^L = \sigma_1(W_4 * \mathbb{O}^L + b_4), \tag{4}$$

where $W_3 \in \mathbb{R}^{1\times k}$, $W_4 \in \mathbb{R}^{1\times k}$, $b_3 \in \mathbb{R}^{n\times n}$ and $b_4 \in \mathbb{R}^{n\times n}$ are learnable parameters, and $\sigma_1(\cdot)$ is the ReLU activation function. $\hat{\mathbb{I}}^L \in \mathbb{R}^{n\times n}$ and $\hat{\mathbb{O}}^L \in \mathbb{R}^{n\times n}$ are the inflow and outflow long-term temporal embedding respectively.

The short-term and long-term dependency are not always equal for stations. Thus, we propose an attentive aggregation approach to fuse the short-term and long-term dependency, respectively.

We define the temporal inflow matrix:

$$\hat{\mathbb{I}} = \beta_I^S \cdot \hat{\mathbb{I}}^S + \beta_I^L \hat{\mathbb{I}}^L. \tag{5}$$

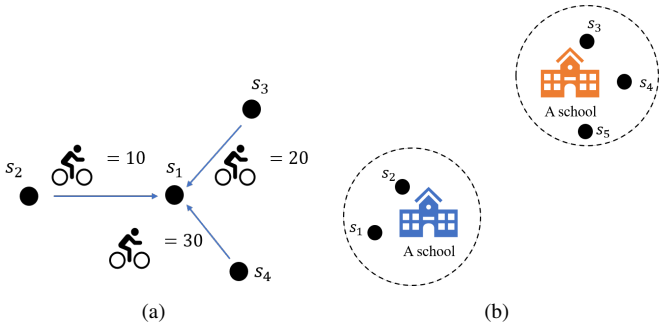


Fig. 3. Illustration of dependencies between stations: (a) Flow between stations indicates their dependency. In this example, s_1 depends more on s_4 than s_2 and s_3 because there exists more flow from s_4 to s_1 than from s_2 and s_3. (b) Stations with similar patterns are likely to be significantly correlated. For example, stations near two schools (no matter near or distant) may have similar demand and supply patterns because the schools have similar operating hours.

β_I^S and β_I^L are computed by

$$\beta_I^S = \frac{\exp(W_5 \cdot \hat{\mathbb{I}}^S)}{\exp(W_5 \cdot \hat{\mathbb{I}}^S) + \exp(W_5 \cdot \hat{\mathbb{I}}^L)}, \tag{6}$$

and

$$\beta_I^L = \frac{\exp(W_5 \cdot \hat{\mathbb{I}}^L)}{\exp(W_5 \cdot \hat{\mathbb{I}}^S) + \exp(W_5 \cdot \hat{\mathbb{I}}^L)}, \tag{7}$$

where $W_5 \in \mathbb{R}^{n\times n}$ are learnable parameters.

Similarly, we have the temporal outflow matrix as follows:

$$\hat{\mathbb{O}} = \beta_O^S \cdot \hat{\mathbb{O}}^S + \beta_O^L \hat{\mathbb{O}}^L, \tag{8}$$

where β_O^S and β_O^L are computed similar to Equations 6 and 7, with learnable parameters $W_6 \in \mathbb{R}^{n\times n}$.

To fuse the various temporal dependencies and jointly consider the inflow and outflow information, we concatenate the above temporal embedding of inflow and outflow as follows:

$$\mathbb{T} = (\hat{\mathbb{I}} || \hat{\mathbb{O}}) \cdot W_7 \tag{9}$$

where $W_7 \in \mathbb{R}^{2n\times n}$ are learnable parameters, and $||$ denotes the concatenation operation. The learned embedding $\mathbb{T} \in \mathbb{R}^{n\times n}$ is used as node features for stations, where $\mathbb{T}_i \in \mathbb{R}^{1\times n}$ is the feature of station s_i. Note that $\mathbb{T}$ is dynamic over time.

B. Flow and Pattern Edge Generations

Edges in a graph reflect the relationships between stations, and the edge weight can be used to indicate their dependency. We consider the flow dependency between stations in FCG, and apply an attention mechanism to emphasize the dependency between stations in terms of demand-supply pattern correlation.

1) Flow edge generation: The flow between stations provides plenty of information to generate the graph. If the flow between two stations is large, the two stations would highly depend on each other regarding their dynamic flow. We present an example in Figure 3(a). The flow from s_4 to s_1 ($O_{4,1} = 30$) is larger than that from s_2 ($O_{2,1} = 10$) and s_3 ($O_{3,1} = 20$), indicating that s_1 depends more on s_4 than s_2 and s_3. Based on this intuition, we generate the FCG to embed the dependency between stations.

We generate a flow-convoluted graph based on $\hat{\mathbb{I}}$ (Equation 5) and $\hat{\mathbb{O}}$ (Equation 8). We generate a edge from s_j to s_i if $\hat{\mathbb{I}}_{i,j} > 0$ or $\hat{\mathbb{O}}_{j,i} > 0$.

A formal definition of the flow-convoluted graph is as follows:

Definition 2: (*Flow-convoluted graph (FCG)*) A flow-convoluted graph at time t is represented as $\mathbb{G}_t^f = (\mathbb{N}^t, \mathbb{E}_f^t)$, where a node $\mathbb{N}_i^t = (s_i, \mathbb{T}_i^t)$ denotes a station s_i with spatial-temporal feature $\mathbb{T}_i^t$ at time t, and $\mathbb{E}_f^t(i,j)$ is the edge weight between nodes s_i and s_j at t:

$$\mathbb{E}_f^t(i,j) = \frac{\mathbb{T}_{i,j}^t}{\sum_{k \in S} \mathbb{T}_{i,k}^t}, \tag{10}$$

where S is a set of bike stations.

FCG is dynamic over time, and it describes both spatial and temporal information in a graph. It can hence denote the time-varying spatial-temporal dependency between stations in terms of their flow.

2) Pattern edge generation: Stations with similar patterns are inclined to be significantly correlated. As we show in Figure 3(b), stations near a school may have similar demand-supply patterns due to the locality effect (e.g., s_1 and s_2). Moreover, stations near two remote schools may also share similar demand-supply patterns (e.g., s_1 and s_3) because schools may have similar operating hours. Unlike most works which only consider the former case (i.e., the locality effect), we propose a data driven approach to generate a pattern correlation graph to consider both cases.

Given the node features of stations, we first compute the attention coefficient $e(i,j)$ of two stations s_i and s_j as follows:

$$e(i,j) = \sigma_2([\mathbb{T}_i \cdot W_8 || \mathbb{T}_j \cdot W_8] \cdot W_9), \tag{11}$$

where $W_8 \in \mathbb{R}^{n \times n}$ and $W_9 \in \mathbb{R}^{2n \times 1}$ are learnable parameters, and $\sigma_2(\cdot)$ is the activation function. Following a prior work [11], we also use the ELU activation function as $\sigma_2(\cdot)$ in our work.

Afterwards, the attention coefficient $e(i,j)$ is passed through a normalized $softmax$ function to get the attention score:

$$\alpha(i,j) = softmax(e(i,j)) = \frac{exp(e(i,j))}{\sum_{u=1}^{n} exp(e(i,u))}. \tag{12}$$

Based on the node features and the attention scores, we define the pattern correlation graph as:

Definition 3: (*Pattern correlation graph (PCG)*) A pattern correlation graph at time t is represented as $\mathbb{G}_t^l = (\mathbb{N}^t, \mathbb{E}_l^t)$, where node $\mathbb{N}_i^t = (s_i, \mathbb{T}_i^t)$ is a station s_i with spatial-temporal feature $\mathbb{T}_i^t$ at time t, and $\mathbb{E}_l^t(i,j)$ is the edge weight for s_i and s_j which is calculated using the attention mechanism.

To summarize, the FCG and PCG consider the spatial-temporal dependency from different views. The pattern correlation graph captures the station dependency according to the correlation of their patterns, while FCG is from the view of flow dependency. Both graphs are time-dependent, and they embed the dynamic spatial-temporal dependency in a unified graph structure.

V. Spatial-temporal Dependency Learning

Based on the generated spatial-temporal graphs in Section IV, we propose a graph neural network to learn the node embedding from the generated spatial-temporal graphs. We first introduce the framework of the proposed graph neural network in Section V-A. Afterwards, we propose a flow-based aggregator to learn the dependency for the flow-convoluted graph (Section V-B), followed by an attention-based aggregator for the pattern correlation graph (Section V-C).

A. Learning framework

A graph neural network (GNN) learns the latent representation of nodes by aggregating information from their neighboring nodes in a graph. Thus, the dependency between stations is considered via the aggregation. We propose using an multi-layer structure to iteratively learn the dependency so that the influence of non-neighboring nodes can be propagated via the edges in the graph.

We use F^0 to denote the initial node features in the graph, where $F_i^0 = \mathbb{T}_i$, and $\mathbb{T}_i$ is defined in Equation 9. Given a station s_i and its neighboring nodes' features, the node feature F_i^k of s_i is updated as

$$F_i^k = \sigma(W^k \cdot Aggr(\{F_i^{k-1}\} \cup \{F_j^{k-1}, \forall s_j \in \mathcal{N}(s_i)\})), \tag{13}$$

where $\mathcal{N}(s_i)$ denotes the neighbouring stations of s_i in the graph, $Aggr(\cdot)$ is the aggregator to aggregate the node features from one's neighbouring nodes, and W^k are learnable parameters.

We illustrate the algorithm in Algorithm 1. Given a graph structure, node feature, activation function σ, aggregator $Aggr(\cdot)$, and the number of layer K, our proposed GNN learns the embedding for each node. It first initializes the node embedding (Line 3). Then, it iteratively updates the embedding of all nodes (Lines 4 - 8). In each round, it updates the node embedding by aggregating the embedding of its neighbouring

Algorithm 1: STGNN-DJD.

1 **Input:** Graph $G = (V, E)$; node features $\{\mathbb{T}_i, \forall s_i \in S\}$; activation function σ; aggregator $Aggr(\cdot)$; number of layer K.
2 **Output:**Embedding of nodes $\mathbb{F}$.
3 $F_i^0 = \mathbb{T}_i$, $\forall s_i \in S$;
4 **foreach** $k \in \{1, 2, ...K\}$ **do**
5 **foreach** $s_i \in S$ **do**
6 $F_i^k = \sigma(W^k \cdot Aggr(\{F_i^{k-1}\} \cup \{F_j^{k-1}, \forall s_j \in \mathcal{N}(s_i)\}))$,
7 **end**
8 **end**
9 $\mathbb{F} = F^k$;
10 return $\mathbb{F}$;

nodes and itself in the last round using Equation 13 (Line 6). Finally, the node embeddings are derived from the feature embeddings in the last layer (i.e., F^k). Also, mean or max pooling are the most common aggregation functions used in GNN. However, such a general aggregation function might not be suitable for capturing the characteristics of bike-sharing data. Thus, we propose customized aggregation functions for the flow convoluted and pattern correlation graphs, in Sections V-B and V-C, respectively.

B. Flow-based Aggregator for the Flow-convoluted Graph

Since larger flow between stations indicates stronger dependency between them, we propose the flow aggregator as follows:

$$Aggr(\{F_i^{k-1}\} \cup \{F_j^{k-1}, \forall s_j \in \mathcal{N}(s_i)\}) = \sum w_{i,u} F_u^{k-1}, \tag{14}$$

where $F_u^{k-1} \in \{F_i^{k-1}\} \cup \{F_j^{k-1}, \forall s_j \in \mathcal{N}(s_i)\}$, and the weight $w_{i,u}$ is calculated according to Equation 10. The flow-based aggregator is supposed to outperform conventional aggregators (e.g., mean or max pooling) because it leverages the characteristic of flow information. We use $\mathbb{F}_i^f$ to denote the final embedding of station s_i in the flow-convoluted graph. As the inflow embedding is time-dependent, the weights of stations hence vary over time. Consequently, the dynamic dependency in terms of flow between stations could be captured.

C. Attention-based Aggregator for the pattern correlation graph

The attention mechanism is a data-driven approach to learn the dependency between any two objects, without making any assumption. Thus, we propose an attention-based aggregator to learn the node embedding in the pattern correlation graph, in which the dependency between nodes is calculated by the attention mechanism. We extend Equations 11 and 12 to a multi-layer network. Given the embedding of two stations F_i^{k-1} and F_j^{k-1}, their attention coefficient $e^k(i, j)$ is then computed as follows:

$$e^k(i, j) = \sigma_2([F_i^{k-1} \cdot W_8 || F_j^{k-1} \cdot W_8] \cdot W_9), \tag{15}$$

where $W_8 \in \mathbb{R}^{n \times n}$ and $W_9 \in \mathbb{R}^{2n \times 1}$ are learnable parameters, and $\sigma_2(\cdot)$ is the activation function. The attention score for the $k-$layer is calculated as

$$\alpha^k(i, j) = softmax(e^k(i, j)) = \frac{exp(e^k(i, j))}{\sum_{u=1}^{n} exp(e^k(i, u))}. \tag{16}$$

Based on that, the embedding of stations F^k in the pattern correlation graph is updated as

$$F^k = \sigma_2(\alpha^k \cdot \phi F^{k-1}), \tag{17}$$

where $\alpha^k \in \mathbb{R}^{n \times n}$ is the attention coefficient matrix for bike stations in Equation 16, σ_2 is the ELU activation function, and $\phi \in \mathbb{R}^{n \times n}$ are learnable parameters.

To improve the generalization of the model and to capture various dependencies, we use multiple (i.e., m) attention heads in the model; we compute multiple attention coefficient matrices using different ϕ, and then concatenate the results, as explained in Equation 18:

$$F^k = (||_{u=1}^{m} \sigma_2(\alpha^{(k,u)} \cdot \phi_u F^{k-1})) \cdot W_{10}, \tag{18}$$

where $W_{10} \in \mathbb{R}^{(m \times n) \times n}$ are learnable parameters, and $||_{u=1}^{m}$ denotes the concatenation operation to the m embedding matrices. We use $\mathbb{F}_i^p$ to denote the final embedding of station s_i in the pattern correlation graph. All in all, to jointly consider both dependencies from the flow-convoluted graph and the pattern correlation graph for a station, we concatenate its flow-convoluted graph embedding $\mathbb{F}_i^f$ and pattern correlation graph embedding $\mathbb{F}_i^p$:

$$\mathbb{F}_i = \mathbb{F}_i^f || \mathbb{F}_i^p, \tag{19}$$

where $||$ is the concatenating operation, and $\mathbb{F}_i$ is the spatial-temporal embedding for a station s_i. $\mathbb{F}_i$ jointly considers the spatial-temporal dependency in the flow-convoluted graph and the pattern correlation graph.

VI. Demand And Supply Predictor

Given the spatial-temporal embedding $\mathbb{F}_i^t$ of a station s_i at time t, we feed it to a fully connected neural network to predict the demand and supply of any station s_i at time t, i.e.,

$$(\hat{x}_i^t, \hat{y}_i^t) = \mathbb{F}_i^t \cdot W_{11}, \tag{20}$$

where $\hat{x}_i^t$ and $\hat{y}_i^t$ are the prediction results of bike demand and supply for s_i at t respectively, and $W_{11} \in \mathbb{R}^{n \times 2}$ are learnable parameters.

We jointly predict the bike demand and supply for any individual stations, and utilize the following loss function for model training:

$$\mathcal{L} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i^t - \hat{x}_i^t)^2 + \frac{1}{n} \sum_{i=1}^{n} (y_i^t - \hat{y}_i^t)^2}, \tag{21}$$

where x_i^t and y_i^t is the ground-truth of s_i's demand and supply at t, and n is the number of stations.

VII. Illustrative Experimental Results

In this section, we first discuss the datasets (Section VII-A), baseline approaches and evaluation metrics (Section VII-B), and hyperparameter settings in our experiments (Section VII-C). Then, we compare the performance of STGNN-DJD with the state-of-the-art approaches in two scenarios: (1) overall performance using whole day data (Section VII-D), and (2) performance at morning rush hours, i.e., 07:00 am to 10:00 am, and evening rush hours, i.e., 05:00 pm to 08:00 pm. (Section VII-E). After that, we evaluate the effectiveness of each component of STGNN-DJD (Section VII-F). Finally, we study the performance of using different aggregators in Section VII-G and the impact of hyperparameters in Section VII-H, followed by the discussion of prediction efficiency in Section VII-I.

A. Datasets

We conduct experiments on two real-world bike-sharing system datasets collected from Chicago and Los Angeles to evaluate the performance of our proposed approaches. The description of the two datasets used in our experiments is as follows:

- *Chicago*. The Chicago dataset[1] was collected from 571 bike stations in the city of Chicago, over nine months from April 1^{st}, 2018 to December 31^{st}, 2018.
- *Los Angeles*. The Los Angeles (LA) dataset[2] was collected from 83 bike stations in the city of Los Angeles over 15 months from October 1^{st}, 2017 to December 31^{st}, 2018.

In both datasets, each trip has the attributes of trip ID, bike ID, start time, end time, origin station ID, destination station ID, original station name, destination station name, etc. We use the same data process approaches following a prior work [11]. For each dataset, we performed data cleansing to filter out data with abnormal trip times (e.g., negative or more than 24 hours) or missing origin/destination stations. After filtering, the Chicago dataset contained $3,152,651$ trips and the LA dataset contained $323,645$ trips.

The time interval was set as 15 minutes in our experiments. Thus, there were 96 intervals per day for both datasets. We grouped the data by stations and time slots to obtain the demand and supply for each station and the flow information between stations. We chose the data of the first 70% of days in each dataset as the training data, the following 10% of days as the validation data, and the remaining data as the testing data. We used the Min-Max normalization to rescale the range of demand and supply in $[0, 1]$. After prediction, we recovered the results for evaluation. In the experiments, when we calculate the RMSE and MAE of our model and the baseline approaches, we exclude the results of those stations which had no demand or supply. Such is a common practice used in industry and many prior works [8], [9].

[1]https://www.divvybikes.com/

[2]https://bikeshare.metro.net/about/data/

B. Baseline Approaches and Evaluation Metric

We compare STGNN-DJD with the following state-of-the-art models:

- HA: Historical Average [43] uses the average of a station's historical demand/supply at the same interval as the prediction result.
- ARIMA: The Auto-Regressive Integrated Moving Average is a widely used time series prediction model. The size of the sliding window is set as 12 in our experiments.
- XGBoost [44]: It is a powerful approach for building supervised regression models. Historical demand and supply at the last k time slots on the same day and the same time slot in the last d days are used as features.
- MLP: Multi-layer perceptron which consists of a three-layer fully-connected neural network is used for prediction.
- LSTM: Long short-term memory is designed to model temporal dependency for prediction.
- RNN [37]: Recurrent neural networks are used for prediction.
- GCNN [45]: The conventional graph convolutional neural network is proposed in the work to predict the demand and supply of each bike station. It only considers the link correlations between stations.
- MGNN [36]: Multi-Graph Neural Networks are proposed in the work for station-based demand and supply prediction. They consider correlations between stations without graph attention.
- ASTGCN [5]: It models three temporal properties of traffic flows independently, i.e., recent, daily-periodic and weekly-periodic dependency. It mainly focuses on dependency between nearby stations.
- STSGCN [42]: It captures the complex localized spatial-temporal correlations using a synchronous graph convolution network.
- GBikes [11]: A spatial-temporal graph attention convolutional neural network is proposed in the work. It assumed that closer stations would have more dependency than distant stations, and used a predefined metric to measure the dependency in terms of distance.

As CNN-based approaches focus on coarse-grained prediction for areas and can hardly extended to docked bike demand and supply prediction, we do not include them (such as [8], [9], [15]) as comparison baseline approaches.

We use RMSE (root mean square error) and MAE (mean absolute error) as the metrics to evaluate the prediction performance of the above baseline models and STGNN-DJD, which are defined as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2 + \sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{2n}}, \quad (22)$$

and

$$MAE = \frac{\sum_{i=1}^{n}(x_i - \hat{x}_i) + \sum_{i=1}^{n}(y_i - \hat{y}_i)}{2n}, \quad (23)$$

where $\hat{x}_i^t$ and $\hat{y}_i^t$ are the prediction results of s_i's demand and supply, x_i and y_i is the ground-truth, and n is the number of stations.

C. Hyperparameter Settings

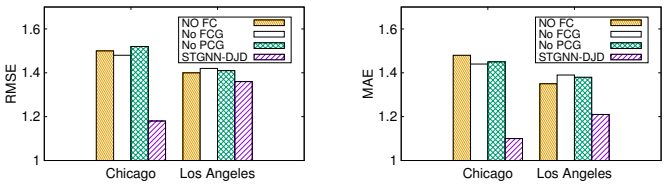
We set the hyperparameters based on the performance of the validation dataset. For temporal information, we use the flow data at the previous 96 time slots ($k = 96$) for short-term temporal dependency consideration while we use that at the same time slot in the previous 7 days ($d = 7$) to consider the long-term dependency. The number of layers for FCG and PCG is set as 2 and 3, respectively. The number of attention heads is set as 4. The batch size in our experiments is set as 32. The learning rate is set as 0.01 and the dropout rate is set as 0.2. Adam optimizer is used for model training [46]. Our model is trained and tested on a machine with an NVIDIA RTX2080 Ti GPU.

D. Comparisons with baselines

We compare the performance of our proposed STGNN-DJD with the state-of-the-art approaches. The overall performances on the two datasets are presented in Table I. As presented in the table, our proposed approach STGNN-DJD significantly outperforms all state-of-the-art approaches on the two datasets in terms of RMSE and MAE.

In particular, traditional time series approaches (namely HA, ARIMA, XGBoost, and MLP) yield poor performances in the experiments compared with the other approaches because they only consider temporal dependency and fail to take the spatial dependency between stations into account. LSTM and RNN have similar performance to the above traditional approaches since they also solely model the temporal dependency on the historical demand and supply. Furthermore, we compare STGNN-DJD with some recent deep learning-based approaches (namely GCNN, MGNN, ASTGCN, STSGCN, and GBike). All of these approaches have significant improvement over traditional time series approaches and conventional RNN and LSTM, which illustrates the importance of capturing the spatial dependency between stations and the effectiveness of GCN for modeling spatial dependency. ASTGCN, STSGCN and GBike have further improvement than GCNN and MGNN. The reason could be that STSGCN considers the synchronous spatial-temporal correlation, and both ASTGCN and GBike incorporate GCN with attention mechanisms to consider dependency between stations, which effectively captures correlations between nodes in a graph. However, all ASTGCN, STSGCN, and GBike are inclined to focus on dependency from nearby stations. Consequently, they cannot sufficiently consider the dependency on distant stations, limiting its performance on prediction.

Compared to the state-of-the-art approaches, STGNN-DJD jointly considers spatial and temporal dependency using novel spatial-temporal graphs and well-designed graph neural networks. The flow-convoluted graph and pattern correlation graph capture both local and global spatial-temporal dependencies between stations. Therefore, it achieves the best overall performance.



(a) RMSE. (b) MAE.

Fig. 4. Comparison with variants of STGNN-DJD.

E. Rush Hours

Bike demand and supply prediction play more crucial roles at rush hours for the bike-sharing system. Thus, we further evaluate the performance of STGNN-DJD, and some baseline approaches (GCNN, MGNN, ASTGCN, STSGCN and GBikes) at rush hours. Here, we only compare the performance of STGNN-DJD with some deep learning-based approaches since they have much better overall performance than others. The rush hours are selected as 07:00 am - 10:00 am and 05:00 pm - 08:00 pm.

The results are shown in Table II. STGNN-DJD outperforms all state-of-the-art approaches at rush hours in the morning and afternoon. The improvement at rush hours is more significant than in Table I. The reason could be that there are more bike trips during the rush hours, and they provide more flow information between stations. The significant improvement at rush hours indicates the effectiveness of our graph convolution approach on the flow-convoluted graph for capturing spatial dependency.

F. Design Variations of STGNN-DJD

There are three fundamental components in STGNN-DJD to capture spatial-temporal dependency between stations: flow convolution for node feature extraction, flow-based aggregation on the flow-convoluted graph, and attention aggregation on the pattern correlation graph. To evaluate the effectiveness of each component, we compare STGNN-DJD with the following variants:

- No Flow Convolution (FC): We skip the flow convolution, and the node features are seen as learnable parameters.
- No Flow-convoluted Graph (FCG): We take away the flow-convoluted graph. Only the pattern correlation graph is used to represent the relations between stations.
- No Pattern Correlation graph (PCG): We do not consider the dependency in terms of demand-supply patterns, and only use the flow-convoluted graph to capture dependency.

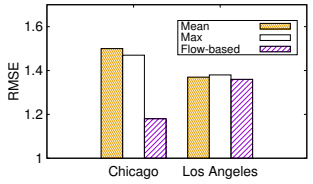
We use RMSE and MAE as metrics to evaluate the performance of variants on the two datasets. The comparison results are presented in Figures 4(a) (RMSE) and 4(b) (MAE). As shown in the figures, removing any components of the

TABLE I
COMPARISON WITH SOTA.

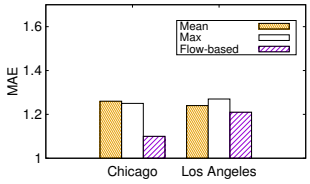
Method	Chicago		Los Angeles	
	RMSE	MAE	RMSE	MAE
HA	3.81	3.09	3.52	3.32
ARIMA	3.58	2.85	3.17	2.73
XGBoost	3.23	2.87	3.16	2.51
MLP	5.51±1.84	5.04±1.37	3.43±0.75	2.98±1.03
RNN	4.27±1.42	3.93±1.12	3.77±1.15	3.16±0.81
LSTM	3.84±1.83	3.27±1.21	3.05±0.63	2.91±0.87
GCNN	2.17±0.43	1.93±0.23	2.05±0.41	1.86±0.25
MGNN	2.24±0.52	2.08±0.31	1.99±0.44	1.81±0.34
ASTGCN	1.28±0.34	1.20±0.24	1.42±0.29	1.29±0.31
STSGCN	1.24±0.28	1.17±0.31	1.38±0.34	1.25±0.35
GBike	1.72±0.47	1.44±0.37	1.52±0.42	1.38±0.33
STGNN-DJD	**1.18±0.37**	**1.10±0.43**	**1.33±0.52**	**1.21±0.40**

TABLE II
PERFORMANCE AT RUSH HOURS.

	Method	Chicago		Los Angeles	
		RMSE	MAE	RMSE	MAE
Morning	GCNN	2.31±0.61	2.07±0.41	2.27±0.83	2.01±0.47
	MGNN	2.29±0.86	2.08±0.67	2.12±0.88	1.94±0.56
	ASTGCN	1.18±0.36	0.94±0.32	1.39±0.31	1.15±0.27
	STSGCN	1.16±0.39	1.01±0.43	1.24±0.36	1.13±0.41
	GBike	1.87±0.62	1.64±0.43	1.55±0.85	1.29±0.72
	STGNN-DJD	**0.73±0.43**	**0.82±0.28**	**0.90±0.08**	**0.88±0.06**
Evening	GCNN	3.18±0.60	2.96±0.52	3.15±0.56	2.92±0.53
	MGNN	2.96±0.62	2.67±0.54	2.31±0.66	2.18±0.61
	ASTGCN	2.37±0.42	2.04±0.33	1.48±0.34	1.17±0.21
	STSGCN	2.28±0.39	1.98±0.43	1.52±0.41	1.21±0.37
	GBike	2.53±0.61	2.25±0.63	1.73±0.79	1.58±0.62
	STGNN-DJD	**1.92±0.26**	**1.46±0.37**	**1.12±0.29**	**1.05±0.23**

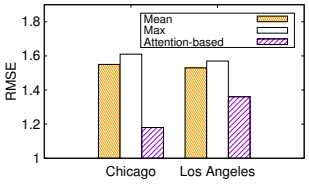


(a) RMSE.

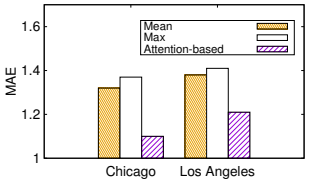


(b) MAE.

Fig. 5. Comparison of different aggregators in the flow-convoluted graph.



(a) RMSE.



(b) MAE.

Fig. 6. Comparison of different aggregators in the pattern correlation graph.

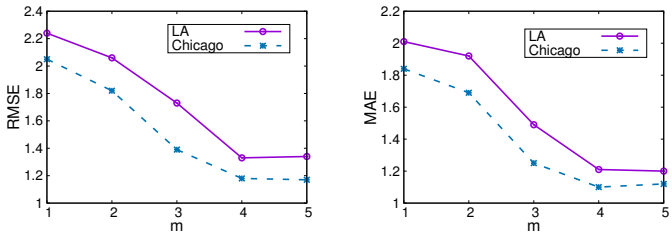
proposed solution results in inferior performance. Since incorporating graph neural networks with flow convolution makes spatial-temporal dependency be jointly learned, the performance of our solution without flow convolution drops significantly, indicating the importance of considering the spatial-temporal node features and the effectiveness of our proposed flow convolution approach. The significant performance degradation in No flow-convoluted graph and No pattern correlation graph shows that our proposed graph neural networks effectively capture inter-station dependency.

G. Aggregator Study

To capture the spatial-temporal dependency between stations, we designed the flow-based aggregator for the flow-convoluted graph and the attention-based aggregator for the pattern correlation graph. In our experiments, we compare the performance of our proposed aggregators with two widely used aggregators [47] to evalute their effectiveness:

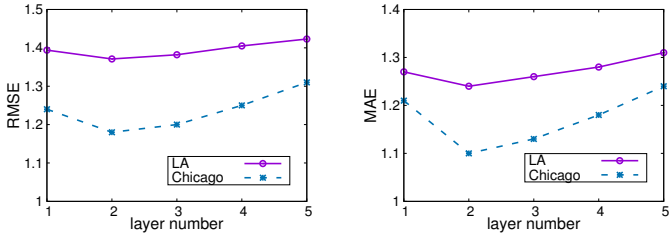
- Mean Aggregator: It takes the element-wise mean of the node embedding of one's neighboring nodes and itself.
- Max Aggregator: We feed the embedding of one's neighboring nodes and itself into a fully-connected neural network independently, and apply an element-wise max-pooling operation to aggregate information.

We first use the above two aggregators to replace the flow-based aggregator. The results of RMSE and MAE on the Chicago dataset and the Los Angeles dataset are presented in Figures 5(a) and 5(b). The flow-based aggregator outperforms the other two aggregators on both datasets. In particular, the



(a) RMSE. (b) MAE.

Fig. 7. Impact of head number m on the performance.



(a) RMSE. (b) MAE.

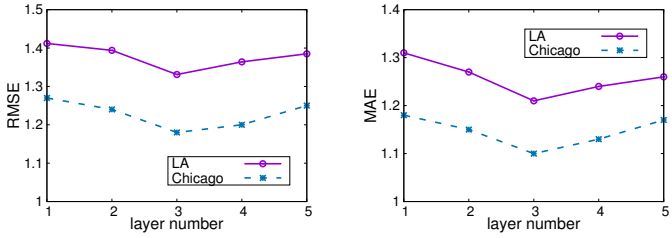
Fig. 8. Impact of FCG layer number on the performance.

difference is more significant on the Chicago dataset. The reason is that the flow-based aggregator relies on the flow between stations, and there are more trips in the Chicago datasets than in the Los Angeles dataset. The results illustrate the effectiveness of the proposed flow-based aggregator.

Then, we replace the attention-aggregator with the mean aggregator and max aggregator. We present the comparison results in Figures 6(a) and 6(b). The performance of the attention aggregator is superior to that of others in terms of RMSE and MAE, which demonstrates the effectiveness of our proposed attention-based aggregator.

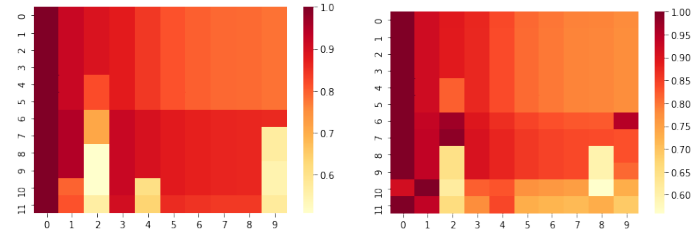
H. Hyperparameters

Since a multi-head attention mechanism is used to capture the station patterns for the pattern correlation graph, we evaluate the impact of the head number m on the RMSE and MAE. The results of RMSE and MAE versus head number m are presented in Figures 7(a) and 7(b), respectively. As shown in Figure 7(a), as the head number m increases, the RMSE on the two datasets declines, illustrating that multi-head attention could capture various dependencies and improve the prediction



(a) RMSE. (b) MAE.

Fig. 9. Impact of PCG layer number on the performance.



(a) Dependency from one station to others. (b) Dependency from others to one station.

Fig. 10. Visualization of dependency of a station from/to the 10 nearest stations from 7:00 am to 10:00 am using an existing approach. The x-axis is the 10 nearest stations, ordered by distance and the y-axis is the time slots from 7:00 am to 10:00 am. A darker color represents higher dependency.

performance. The improvement is not significant when $m > 4$. The reason could be that each head focuses on different parts of the input data. When the heads are many, some of them may focus on the same pattern. Thus, the improvement diminishes when m becomes large (e.g., $m > 4$ in our experiment). Similar results of MAE can be observed in Figure 7(b). Thus, we have $m = 4$ as the default parameter in our experiments.

Furthermore, we evaluate the impact of the layer number for FCG and PCG on the RMSE and MAPE. The results of RMSE and MAE versus FCG layer number are presented in Figures 8(a) and 8(b). Our model achieves the best performance when the FCG layer number is 2. The RMSE and MAPE versus PCG layer number in Figures 9(a) and 9(b) reveals that our model has the best performance when the PCG layer number is 3. The results indicate that stacking layers for graph neural network can enlarge its receptive field and hence improve its performance, but too many layers may introduce more learnable parameters and lead to difficulty in model training.

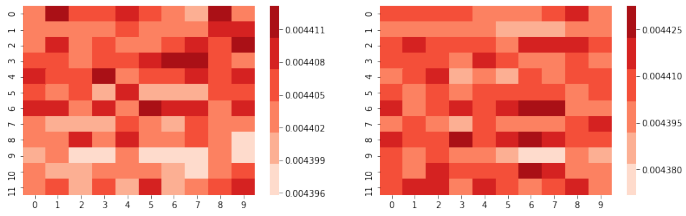
I. Prediction Efficiency

Our model is trained offline, and it does not need to be retrained for online prediction. The average prediction time for all stations in LA and Chicago datasets at a time slot is around 0.014 seconds and 0.038 seconds. Such prediction time is much lower than the duration of a time slot, demonstrating that our model can be applied in real world application.

VIII. Case Study

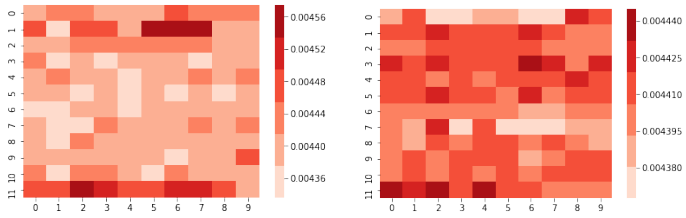
In this section, we study the dependency between stations via some visualization results. Firstly, we show that existing works fail to consider the dependency between distant stations. Then, we demonstrate that the dependency between a pair of stations varies over time, and the dependencies vary for different pairs of stations, even at a single time slot. After that, we discuss the relationship of the dependency between stations and their distance.

When considering the dependency between stations, prior works usually assumed the locality effect, i.e., strong dependency for close stations, while neglecting the dependency for the distant ones. We visualize the dependency between the Wabash Ave & Grand Ave station in Chicago and its ten



(a) Dependency from one station to others. (b) Dependency from others to one station.

Fig. 11. Visualization of dependency of a station from/to the 10 nearest stations from 7:00 am to 10:00 am. The x-axis is the 10 nearest stations, ordered by distance and the y-axis is the time slots from 7:00 am to 10:00 am. A darker color represents higher dependency.



(a) Dependency from one station to others. (b) Dependency from others to one station.

Fig. 12. Visualization of dependency of a station from/to the 10 nearest stations from 03:00 pm to 06:00 pm. The x-axis is the 10 nearest stations, ordered by distance and the y-axis is the time slots from 03:00 pm to 06:00 pm. A darker color represents higher dependency.

nearest stations (from 07:00 am to 10:00 am) using an existing approach [11] in Figure 10. A dark color indicates prominent dependency. As shown in the figure, the prior approach assumes that closer stations always have higher dependency than distant stations. We argue that this assumption may not be valid for docked bike systems. We will demonstrate that dependency may not be strong for close stations, while that of distant ones may not be negligible.

Recall that we use an attention mechanism to capture dependency between stations in the pattern correlation graph, in which the dependency between stations is represented as their attention scores. We first calculate the attention scores between a bike station and its ten nearest stations from 07:00 am to 10:00 am and from 03:00 pm to 06:00 pm. Then we visualize their dependencies at different time slots. The duration of a time slot is 15 minutes following the previous experiments.

We present the visualization result of the Wabash Ave & Grand Ave station in Chicago using our approach in Figures 11 and 12, in which the x-axis is the ten nearest stations, ordered by distance, and the y-axis is the 12 time slots (i.e., 7:00 am to 10:00 am in Figure 11 and 03:00 pm to 06:00 pm in Figure 12). In Figures 11(a) and 12(a), a cell (x_i, y_j) indicates the influence from the target station to the i-th station at the j-th time slots, while a cell (x_i, y_j) in Figures 11(b) and 12(b) represents the influence from the i-th station to the target station. A dark color represents prominent dependency.

As shown in Figure 11(a), the color of cells in the same column (i.e., cells with the same x_i) are different, indicating that the influence from one station to another varies over time. Moreover, the color of cells in the same row (i.e., cells with the same y_i) are also different, illustrating that the influence from one station to other stations is different even at the same time. A similar observation can be found in Figures 12(a), 11(b), and 12(b). The results demonstrate that our proposed model could capture the dynamic dependency between different pairs of stations and at different time slots.

Furthermore, contrary to the assumption that has been made in prior works, the dependency between two stations does not monotonically decrease based on the stations' distance. As shown in Figure 11(a), the color of the grid at position $(9, 11)$, which is the 10th nearest station at the 11th time slot, is darker than the grid at position $(0, 11)$, which is the closest station at the 11th time slot. This finding indicates that even though distance is an essential aspect of spatial dependency, the dependency between stations might not always be inverse proportional to their distance. Similar findings can also be observed in Figures 12(a), 11(b), and 12(b). The visualization results confirms that the locality assumption does not always hold for a docked bike system and our approach is effective to capture both local and global dependency.

IX. Conclusion and Future Works

We propose STGNN-DJD, a spatial-temporal graph neural network to solve the docked bike demand and supply prediction. It employs novel spatial-temporal graphs and an effective graph neural netowrk to consider the joint spatial-temporal dependency between stations regarding their flow relationships and demand-supply patterns. Unlike prior works which consider the spatial and temporal dependencies sequentially in a decoupled manner, STGNN-DJD achieves lower RMSE and MAE results compared to the baselines in two real-world datasets because of the joint spatial-temporal modeling and customized aggregation functions. We have also provided a case study to demonstrate the importance of considering both local and global dependencies between stations. Moreover, we also show that the dependency between stations varies over time and the dependency are various for different pairs of stations at a single time slot.

We discuss below the possible future directions of the work. One is to extend STGNN-DJD for multi-step prediction. A simple way to extend our approach for multiple slot prediction is replacing the model output $\{O_t, I_t\}$ as $\{O_t, \cdots O_{t+k}, I_t, \cdots, I_{t+k}\}$ in both training and prediction phases. We will study as a future work more sophisticated approaches for multi-step prediction considering dynamic and joint spatial-temporal dependency. Another direction is to design efficient prediction models for both training and prediction phases, and provide detailed experimental comparison.

Acknowledgment

This work was supported, in part, by Hong Kong General Research Fund (under grant number 16200120).

REFERENCES

[1] https://www.statista.com/statistics/868126/global-bikesharing-market size/, "Global bike-sharing service market size between 2020 and 2026," *Statista*, 2021.

[2] J. Bao, T. He, S. Ruan, Y. Li, and Y. Zheng, "Planning bike lanes based on sharing-bikes' trajectories," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 1377–1386.

[3] H. Liu, T. Li, R. Hu, Y. Fu, J. Gu, and H. Xiong, "Joint representation learning for multi-modal transportation recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1036–1043.

[4] Y. Li, Z. Zhu, D. Kong, M. Xu, and Y. Zhao, "Learning heterogeneous spatial-temporal representation for bike-sharing demand prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1004–1011.

[5] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 922–929.

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

[7] M. Li and Z. Zhu, "Spatial-temporal fusion graph neural networks for traffic flow forecasting," in *Proceedings of the 27th ACM International Conference on knowledge Discovery and Data Mining*. Singapore: ACM, 2021.

[8] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial-temporal similarity: A deep learning framework for traffic prediction," in *2019 AAAI Conference on Artificial Intelligence (AAAI'19)*, 2019.

[9] H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Thirty-Second AAAI Conference on Artificial Intelligence*. Louisiana, USA: AAAI, 2018, pp. 2588 – 2595.

[10] X. Wang, Y. Ma, Y. Wang, W. Jin, X. Wang, J. Tang, C. Jia, and J. Yu, "Traffic flow prediction via spatial temporal graph neural network," in *Proceedings of The Web Conference 2020*, 2020, pp. 1082–1092.

[11] S. He and K. G. Shin, "Towards fine-grained flow forecasting: A graph attention approach for bike sharing systems," in *Proceedings of The Web Conference 2020*, 2020, pp. 88–98.

[12] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Seattle, Washington: ACM, 2015, pp. 1–10.

[13] L. Chen, D. Zhang, L. Wang, D. Yang, X. Ma, S. Li, Z. Wu, G. Pan, T.-M.-T. Nguyen, and J. 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*, 2016, pp. 841–852.

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

[15] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Thirty-First AAAI Conference on Artificial Intelligence*. California USA: AAAI, 2017, pp. 1655 – 1661.

[16] T. Li, J. Zhang, K. Bao, Y. Liang, Y. Li, and Y. Zheng, "Autost: Efficient neural architecture search for spatio-temporal prediction," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Virtual Conference: ACM, 2020, pp. 794–802.

[17] P. Vogel and D. C. Mattfeld, "Strategic and operational planning of bike-sharing systems by data mining–a case study," in *International conference on computational logistics*. Springer, 2011, pp. 127–141.

[18] P. Vogel, T. Greiser, and D. C. Mattfeld, "Understanding bike-sharing systems using data mining: Exploring activity patterns," *Procedia - Social and Behavioral Sciences*, vol. 20, pp. 514–523, 2011.

[19] Y. Tong, Y. Chen, Z. Zhou, L. Chen, J. Wang, Q. Yang, J. Ye, and W. Lv, "The simpler the better: a unified approach to predicting original taxi demands based on large-scale online platforms," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. Halifax, NS, Canada: ACM, 2017, pp. 1653–1662.

[20] J. Liu, L. Sun, W. Chen, and H. Xiong, "Rebalancing bike sharing systems: A multi-source data smart optimization," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1005–1014.

[21] Y. Zhang and Y. Xie, "Forecasting of short-term freeway volume with v-support vector machines," *Transportation Research Record*, vol. 2024, no. 1, pp. 92–99, 2007.

[22] Y. Xu, Q.-J. Kong, R. Klette, and Y. Liu, "Accurate and interpretable bayesian mars for traffic flow prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2457–2469, 2014.

[23] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys (CSUR)*, vol. 51, no. 5, pp. 1–36, 2018.

[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[25] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[26] R. Guo, Z. Jiang, J. Huang, J. Tao, C. Wang, J. Li, and L. Chen, "Bikenet: Accurate bike demand prediction using graph neural networks for station rebalancing," in *2019 IEEE SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI*, 2019, pp. 686–693.

[27] G. Xiao, R. Wang, C. Zhang, and A. Ni, "Demand prediction for a public bike sharing program based on spatio-temporal graph convolutional networks," *Multimedia Tools and Applications*, 2020.

[28] Z. Fang, Q. Long, G. Song, and K. Xie, "Spatial-temporal graph ode networks for traffic flow forecasting," in *Proceedings of the 27th ACM International Conference on knowledge Discovery and Data Mining*. Singapore: ACM, 2021.

[29] H. Shi, Q. Yao, Q. Guo, Y. Li, L. Zhang, J. Ye, Y. Li, and Y. Liu, "Predicting origin-destination flow via multi-perspective graph convolutional network," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020, pp. 1818–1821.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[31] H. Lin, R. Bai, W. Jia, X. Yang, and Y. You, "Preserving dynamic attention for long-term spatial-temporal prediction," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Virtual Conference: ACM, 2020, pp. 36–46.

[32] H. Lin, W. Jia, Y. You, and Y. Sun, "Interpretable crowd flow prediction with spatial-temporal self-attention," *arXiv preprint arXiv:2002.09693*, vol. [cs.LG], 2020.

[33] Y. Zhou, J. Li, H. Chen, Y. Wu, J. Wu, and L. Chen, "A spatiotemporal attention mechanism-based model for multi-step citywide passenger demand prediction," *Information Sciences*, vol. 513, pp. 372–385, 2020.

[34] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 1082–1090.

[35] T. Qin, T. Liu, H. Wu, W. Tong, and S. Zhao, "Resgcn: Residual graph convolutional network based free dock prediction in bike sharing system," in *2020 21st IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2020, pp. 210–217.

[36] D. Chai, L. Wang, and Q. Yang, "Bike flow prediction with multi-graph convolutional networks," in *Proceedings of the 26th ACM SIGSPATIAL international conference on advances in geographic information systems*, 2018, pp. 397–400.

[37] Y. Pan, R. C. Zheng, J. Zhang, and X. Yao, "Predicting bike sharing demand using recurrent neural networks," *Procedia computer science*, vol. 147, pp. 562–566, 2019.

[38] H. Yuan, G. Li, Z. Bao, and L. Feng, "An effective joint prediction model for travel demands and traffic flows," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 348–359.

[39] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*. Vancouver, BC, Canada: ICLR, 2018, pp. 1 – 16.

[40] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions,"

in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[42] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 914–921.

[43] J. E. Froehlich, J. Neumann, and N. Oliver, "Sensing and predicting the pulse of the city through shared bicycling," in *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

[44] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[45] L. Lin, Z. He, and S. Peeta, "Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach," *Transportation Research Part C-emerging Technologies*, vol. 97, pp. 258–276, 2018.

[46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv: Learning*, 2014.

[47] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in neural information processing systems*, 2017, pp. 1024–1034.