Citywide Bike Usage Prediction in a Bike-Sharing System

Yexin Li and Yu Zheng, *Senior Member, IEEE*

Abstract—To operate a bike-sharing system efficiently, system operators need to accurately predict how many bikes are to be rented and returned throughout the city. In this paper, we propose a Hierarchical Consistency Prediction (HCP) model to predict the citywide bike usage in the next period. First, an Adaptive Transition Constraint (AdaTC) clustering algorithm is proposed to cluster stations into groups, making the rent and transition at each cluster more regular than those at each single station. Second, a Similarity-based efficient Gaussian Process Regressor (SGPR) is proposed to respectively predict how many bikes are to be rented at different-scale locations, i.e., at each station, each cluster, and in the entire city. Besides largely improving the training and online prediction efficiency, our regressor considers external impacted factors, addresses the data unbalance issue, and better captures the non-linearity in spatio-temporal data. Third, we design a General Least Square (GLS) formulation to collectively improve those obtained predictions via a mutual reinforcement way. GLS makes the final predictions for rent more reasonable. Considering the causality between rent and return, a Transition based Inference (TINF) method is designed to infer the citywide bike return demand based on the predicted rent demands. Experiments on real-world data are conducted to confirm the effectiveness of our model.

Index Terms—Citywide bike usage prediction, Gaussian process regressor, Spatio-temporal dataset

1 Introduction

Bike-sharing systems are widely deployed in many major cities, e.g., New York City, Paris and Beijing, proving a healthy and convenient transportation mode to citizens. A user can rent or return a bike at a random station by swiping her membership card, generating a bike rent or return record, which includes her user ID, the bike ID, the station name, and a timestamp.

As bike usage throughout the city is skewed, a system always has some jammed stations, which do not have enough free docks, and some starved ones, which lack bikes, leading to customer loss. System operators try to solve this problem via constantly repositioning bikes among stations by trailer [15]. However, a real-time monitoring-based bike reposition strategy cannot tackle this problem satisfactorily, as it is too late to take measures after jammed and starved stations have already been observed, especially when large-scale reposition is required (i.e., there is a long distance between the stations to load and unload bikes). Therefore, accurate prediction for how many bikes are to be rented and returned throughout the city is necessary, based on which bike reposition can be conducted in advance, i.e., delivering bikes from stations which are predicted to be jammed to those that are predicted to be starved.

- *Y. Li is with the Computer Science and Engineering Department, the Hong Kong University of Science and Technology, Hong Kong. E-mail: yliby@connect.ust.hk.*
- *Y. Zheng is with the Urban Computing Business Unit, JD Finance, Beijing, China. E-mail: msyuzheng@outlook.com.*

Manuscript received 20 Aug. 2017; revised 20 Dec. 2018; accepted 28 Jan. 2019. Date of publication 12 Feb. 2019; date of current version 29 Apr. 2020. (Corresponding author: Yexin Li.)
Recommended for acceptance by Y. Zhang.
Digital Object Identifier no. 10.1109/TKDE.2019.2898831

However, predicting the citywide bike usage accurately is very challenging for *three* reasons.

First, being impacted by multiple factors, the rent demand at each station fluctuates largely. Fig. 1A shows the hourly rent demand at a random station in one week, which confirms the above claim.

- *Time*. Bike usage is largely impacted by hour of day and day of the week, e.g., bike traffic in a city spikes in morning and evening rush hours while collapsing in other periods on weekday. However, we cannot observe rush hours at weekend or on a holiday as commutes are not necessary on these days.
- *Meteorology*. Outdoor riding is significantly impacted by meteorology. Severe weather conditions prevent customers from riding but to choose other transportation modes such as taxi and bus. Besides, increasing outdoor temperature encourages more people to ride while having a negative effect when it becomes too high. Heavy wind is another negative factor for riding.
- *Event*. Unexpected events also largely affect bike usage. An accident leading to a jam may encourage more people to ride instead of taking taxi nor bus. A festival celebration may lead to a sharp return-increase at stations surrounding the celebration area before the event while a large rent-increase there after that.

Second, most transitions in a bike-sharing system seem to be random trips. According to historical data analysis, the transitions which averagely happened at least once in the morning on each weekday in Apr.-Oct. 2016 only account for 18 percent. We explain this by an example shown in Fig. 1B:

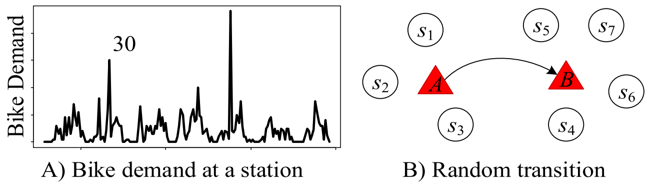

Fig. 1. Hourly bike demand and transition.

there are 12 possible inter-station transitions from A to B, among which a customer chooses a random one to take, based on which stations have available bikes or docks, making one possible frequent transition from A to B into 12 infrequent inter-station ones. Random trip issue makes it hard to estimate which station a rented bike is to be returned, thus hard to predict the return demand at each station.

Third, external factors impacting bike usage are unbalanced observed. As we can imagine, there are much more sunny hours than the rainy ones. Similar issue also exists between common hours and those with unexpected events. Besides, the temperature and wind speed almost concentrate in some ranges while rarely scatter at other values. However, traditional machine learning methods are trained to fit the major conditions while scarifying accuracy under the minor ones. Separately training a model under each condition makes the training data under the minor ones very sparse, which also compromises the model accuracy.

Considering these challenges, we propose a Hierarchical Consistency Prediction (HCP) model to predict the citywide rent and return demand. Our contributions can be summarized into *four-fold*.

- An Adaptive Transition Constraint (AdaTC) clustering algorithm is proposed to cluster stations into groups. AdaTC iteratively clusters stations based on their geographical locations and bike usage patterns, obtaining clusters with more regular rent and transition not only than each single station but also clusters obtained by other algorithms.
- We propose a Similarity-based efficient Gaussian Process Regressor (SGPR) to predict the rent demand at different-scale locations, i.e., at each station, each cluster, and in the entire city. Besides largely improving training and online prediction efficiency, SGPR considers impacted factors, addresses data unbalance issue, and captures the non-linearity in spatio-temporal data.
- A General Least Square (GLS) formulation is designed to collectively improve the obtained prediction at each location. By GLS, the original prediction at each location mutually reinforce each other, thus to obtain more reasonable and accurate results.
- We evaluate our model on real-world datasets from New York City and Washington D.C. Our model outperforms *eight* baselines significantly.

2 Model Overview

This section defines some notations (in Table 1) and terminologies used throughout our paper and overviews the framework.

TABLE 1
Notations

Notation	Description
S_1^0	Entire system
S_k^1	A cluster where $k = 1, 2, \ldots, K^1$
S_k^2	A station where $k = 1, 2, \ldots, n$
x_{kt}^L	Check-out at S_k^L in period t
y_{kt}^L	Check-in at S_k^L in period t

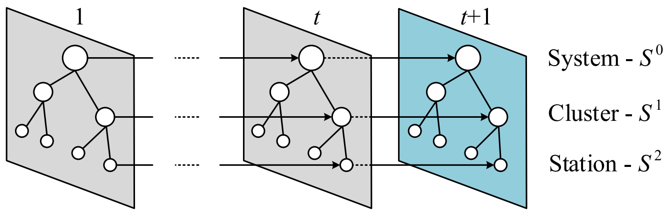

Fig. 2. Hierarchy time series prediction problem.

2.1 Preliminary

Def 1. Hierarchy. A hierarchy is made up by different-scale locations in a system, describing the spatial relationships among them. A 3-level hierarchy is shown in each slice of Fig. 2, whose root denotes the entire system, first level is made up by clusters, and leaves are stations. A node in the L level is denoted as S_k^L where $k = 1, 2, \ldots$.

Def 2. Trip. A trip is a tuple $tr = (S_o^2, \tau_o, S_d^2, \tau_d)$ describing that a bike is rented from station S_o^2 at timestamp τ_o and returned to S_d^2 at τ_d.

According to practical targets, deeper hierarchies can be designed, and the model proposed in this paper can be easily generalized to them. Considering notation brevity, a general notation S is adopted to stand for a random-scale location, i.e., a station or a cluster or the entire system, when distinguishing between them is not necessary.

Def 3. Check-out and check-in. A check-out x_{kt}^L means how many customers want to rent a bike from location S_k^L in period t, including the ones succeed or not. Similarly, a check-in y_{kt}^L describes how many customers want to return to location S_k^L in t. Simplified notations x_t and y_t are adopted to denote them at a general location S in t.

Def 4. Hierarchy time series. A check-out hierarchy time series is a sequence of hierarchies shown in Fig. 2, where each hierarchy corresponds to a specific period and each node is associated with a check-out at the corresponding location in that period. Hierarchy time series has a Hierarchical Consistency (HC) property, i.e., the value at a parent node is equal to the sum of those at its children nodes.

Fig. 2 shows a hierarchy time series prediction problem, where a sequence of historical hierarchies (in periods $1, 2, \ldots, t$) is given and that in the target period $t+1$ is required to be predicted. Note that the obtained predictions should guarantee the HC property.

Main Idea. Considering the fluctuating rent and random trip issues at each single station, we first cluster stations into clusters, obtaining a three-level hierarchy which has three kinds of locations with different scales.

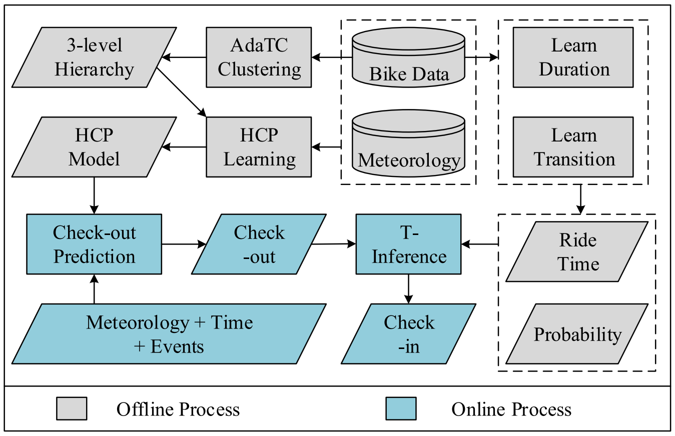

Fig. 3. Bike usage prediction framework.

- *A city*. The root of a hierarchy denotes the entire system whose rent and return are very regular. Root can bound the values at clusters by HC property.
- *A cluster*. Each cluster is made up by several close stations. Our clustering aims at making the bike usage at each cluster regular, thus can be predicted more accurately. As reposition among clusters is large-scale and time-consuming, accurate cluster-level prediction is required. Besides, check-out at each cluster not only feeds back local information to root but also bounds the values at its stations.
- *A station*. Although our model can largely improve the station-level prediction accuracy, bike usage at each station is too chaotic to be predicted as accurately as that at each cluster. However, as station-level predictions are only referred to when conducting complementary mini-scale reposition inner each cluster, which is time-efficient and can largely depend on real-time monitoring, we do not require much on their accuracy. Besides, check-out at each station can feed back local information to their parent clusters.

There are two ways to predict the check-out at different-scale locations discussed above while guaranteeing the HC property. 1) Predict the check-out at root which is then allocated to its descendant clusters and stations. This method may ignore local information from clusters and stations. 2) Predict the check-out at each station and then infer those at higher levels by summing. This method may accumulate error. Our model is to separately predict the check-out at each location and collectively improve them by a mutually reinforcement way, thus to guarantee the HC property and obtain a higher prediction accuracy.

2.2 Framework

Fig. 3 elaborates the framework of our model, consisting of an offline process and an online one.

2.2.1 Offline Process

Adaptive Transition Constraint (AdaTC) Clustering. To guarantee large-scale reposition can be effectively conducted in advance, AdaTC algorithm clusters stations into groups based on their geographical locations and bike usage patterns.

Hierarchical Consistency Prediction (HCP) Model. Obtaining a 3-level hierarchy, we formulate a check-out hierarchy time series prediction problem as Fig. 2. A Similarity-based efficient Gaussian Process Regressor (SGPR) is proposed to separately predict the check-out at each location in the hierarchy. After obtaining original predictions, we design a General Least Square (GLS) formulation to collectively improve them, making the final prediction at each location more reasonable and accurate.

Transition and ride duration learning. Considering the causality between rent and return, check-in at each location is predicted by a Transition-based Inference (T-INF) method. Two major components in T-INF are transition probability learning and ride duration learning. Each time a bike is rented, we estimate where and when the bike is to be returned to infer the check-in at each location in the future. T-INF can implicitly ensure the HC property of check-in in a hierarchy.

2.2.2 Online Process

Check-out Prediction. For a specific location S in the target period $t+1$, we extract real-time information from time, meteorology, and events to form a feature vector f_{t+1}^{S}. Learned HCP model is then adopted to predict its check-out based on the extracted feature.

Transition-based Inference (T-INF). Obtaining the predicted citywide check-out and the learned transition probability and ride duration, we design T-INF method to infer the check-in at each location S in $t+1$.

3 Offline Process

3.1 AdaTC Clustering Algorithm

Before digging into clustering methodology, we first discuss the motivation and insight.

3.1.1 AdaTC Clustering Insight

First, to make the check-out at each cluster more regular, we may want the stations in on cluster have similar check-out patterns. Second, motivated by the random inter-station transition example shown in Fig. 1B, large-scale reposition only need to guarantee that there are bikes at station S_1 or S_2 or S_3 and available docks at S_4 or S_5 or S_6 or S_7. Customers from A to B can choose where to rent and return themselves considering how many bikes and docks are at each station. Besides, supplementary mini-scale reposition among stations around A or B is also conducted to bring more convenience to users. Therefore, we respectively cluster the several stations around the origin and destination to formulate two clusters, i.e., S_1, S_2 and S_3 make up one cluster and S_4, S_5, S_6 and S_7 make up the other one. Consequently, 12 infrequent inter-station transitions merge to 1 frequent inter-cluster one. To formally formulate the idea discussed above, we summarize *three* constraints when designing the clustering algorithm.

- Geographical closeness. As customers need to choose stations nearby to rent or return, and mini-scale reposition inner each cluster should be time-efficient, stations in one cluster need to be close to each other.

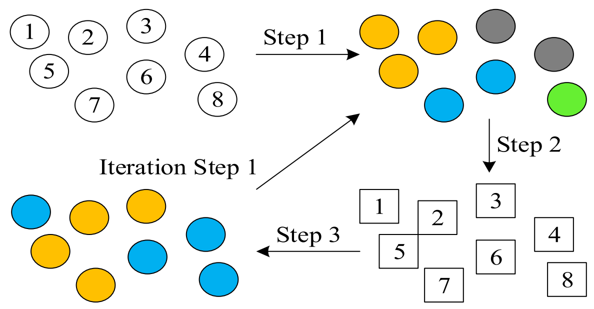

Fig. 4. AdaTC clustering algorithm.

- Similar check-out. Stations in one cluster should have similar check-out patterns, i.e., their check-out increase and decrease similarly, thus to make the check-out at each cluster more regular.
- Concentrated transition. Stations in one cluster should have similar transitions, i.e., they have similar origin and destination clusters, thus to merge infrequent inter-station transitions into frequent inter-cluster ones.

To summarize, clusters obtained under these constraints are made up by close stations with similar check-out and transition patterns. Therefore, they can be considered as function regions in a city, e.g., stations around a resident area are very possible to form a cluster. This also brings extra benefit to system management.

3.1.2 AdaTC Clustering Methodology

As shown in Fig. 4, AdaTC is an iterative process repeating three steps, i.e., Geo-Clustering, T-matrix Generation and T-Clustering.

Geo-Clustering. Stations are first clustered into K^1 groups according to their geographical locations and check-out patterns by K-mean clustering. Initially, Geo-clustering is conducted to all the stations in a system. For the following times, Geo-clustering is conducted to stations in each cluster obtained by T-Clustering proportionally, i.e., assuming the number of stations in each cluster obtained by T-Clustering is $n_1, n_2, \ldots, n_{K^2}$, geo-clustering respectively clusters the stations in each cluster into:

$$\left[\frac{n_1 \times K^1}{n}\right], \left[\frac{n_2 \times K^1}{n}\right], \ldots, \left[\frac{n_{K^2} \times K^1}{n}\right],$$

groups, where n is the total number of stations in a system and $[\cdot]$ is a rounding operator.

The dissimilarity based on which Geo-Clustering is conducted is a tradeoff between geographical distance and check-out difference. Defining the check-out pattern at a station S by Eq. (1) where u_i is the average check-out at S in the i-th time slot in Table 2, we can measure the check-out difference between two stations S_h and S_k by $dis^1_{hk} = |U_h - U_k|$. Therefore, the dissimilarity between S_h and S_k is estimated by Eq. (2) where d_{hk} is the geographical distance between them while ρ_1 a tradeoff parameter. Geo-Clustering obtains K^1 clusters denoted as $C^1_1,\ C^1_2,\ \ldots,\ C^1_{K^1}$.

$$U = \left(\frac{u_1}{\sum_{j=1:3} u_j}, \frac{u_2}{\sum_{j=1:3} u_j}, \frac{u_3}{\sum_{j=1:3} u_j}, \frac{u_4}{\sum_{j=4:5} u_j}, \frac{u_5}{\sum_{j=4:5} u_j}\right) \tag{1}$$

TABLE 2
Time Slots

Day	Time	Time Slot
Weekday	7:00am-11:00am	Morning rush hours
	12:00pm-16:00pm	Day hours
	17:00pm-21:00pm	Evening rush hours
Weekend Holiday	9:00am-17:00pm	Trips hours
	18:00pm-23:00pm	Evening hours

$$G_{hk} = \rho_1 \times d_{hk} \times dis^1_{hk}. \tag{2}$$

T-matrix Generation. Based on the K^1 clusters obtained by Geo-Clustering, we generate a T-matrix A for each station S to describe its transitions. A T-matrix has five rows respectively corresponding to the five time-slots in Table 2. Each row is made up by the *ride-to-cluster* and *return-from-cluster* probability in the specific time slot. Fig. 5 gives an example where the *ride-to-cluster* probability of a specific station S is $(0.1,\ 0.2,\ 0.3,\ 0.4)$ while its *return-from-cluster* probability is $(0.5,\ 0.2,\ 0.2,\ 0.1)$ in the morning rush hours. They respectively describe how probable that a bike rented from S is to be returned to each cluster and a bike returned to S has been rented from each cluster in the morning. They are estimated by maximum likelihood estimation [12]. Therefore, the first row in the T-matrix to S should be $(0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0.2,\ 0.2,\ 0.1)$ by concatenating these two probability vectors. Other rows in the matrix can be generated in a similar way.

T-Clustering. After obtaining a T-matrix for each station, T-Clustering step clusters the stations in a system into K^2 groups by K-mean algorithm based on their T-matrices. Dissimilarity between two stations S_h and S_k is estimated by Eq. (3) where A_h and A_k are their T-matrices. K^2 clusters are obtained and denoted as $C^2_1,\ C^2_2,\ \ldots,\ C^2_{K^2}$. Parameters are set to ensure that $K^1 \geq K^2$.

$$dis^2_{hk} = ||A_h - A_k||_2 \tag{3}$$

Continually iterate these three steps until the iteration threshold is reached. AdaTC outputs the final K^1 clusters obtained by Geo-Clustering step as the final results.

3.2 Hierarchical Consistency Prediction Model

Obtaining a three-level hierarchy after AdaTC, predicting the citywide check-out can be formulated as a hierarchy time series prediction problem as shown in Fig. 2. A HCP model is proposed to solve this problem. HCP has two components, i.e., a SGPR to separately predict the check-out at each location and a GLS to collectively improve these original predictions. SGPR for each location S has three steps[1].

- Extract a feature for the location in each historical and the target period as $f_1, f_2, \ldots, f_t$ and f_{t+1} based on historical bike usage data and meteorology data. Each feature corresponds to an observed check-out at this location in their period, i.e., $x_1,\ x_2,\ \ldots\ ,\ x_t$ and x_{t+1}.

1. As SGPR treats each location fairly, location notation S is ignored later.

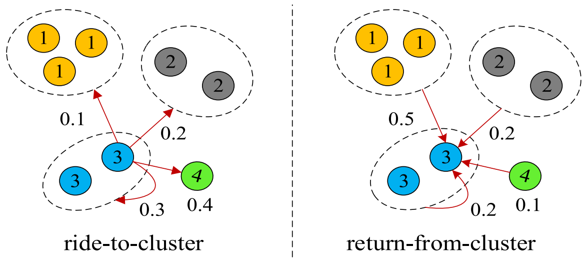

Fig. 5. Station-cluster probability in the morning.

- Measure similarity between the target period and each historical one via their features. Select the top $-I$ similar historical periods to $t+1$ which are denoted as:

$$H_{t+1} = \left\{\left(f_1^{t+1},\ x_1^{t+1}\right),\ \left(f_2^{t+1},\ x_2^{t+1}\right),\ \ldots,\ \left(f_M^{t+1},\ x_M^{t+1}\right)\right\},$$

  and named as the $M-$ similar set to the target period.
- For target period $t+1$ associated with a feature f_{t+1}, we adopt Gaussian Process Regressor (GPR) to predict its check-out based on H_{t+1} and obtain a prediction $\hat{x}_{t+1}$.

Our SGPR has *four* advantages. First, the impacted factors to bike usage are considered. As we adopt similarity to select training set for each target period, the data unbalance issue can be addressed to some extent. Second, relationship between impacted features and check-out is not assumed to be linear. As SGPR is a regressor without specifying function form, we can capture the non-linearity in spatio-temporal data. Third, training and online prediction time for a traditional GPR are respectively cube and square of the training data size. However, SGPR is trained and tested on a $M-$ similar set where $M \ll N$ (the original training data size), largely reducing the time for learning and online prediction. Lastly, the obtained prediction at each location by SGPR is not a scalar but a normal distribution, whose mean and variance can be respectively considered as the predicted check-out and its prediction confidence. Therefore, further adjustment can be collectively conducted by a mutual reinforcement way. These three steps in SGPR are to be illustrated one by one, after which GLS follows.

3.2.1 Feature Extraction

Features are extracted from external impacted factors. As discussed above, three categories of impacted factors are identified, i.e., time, meteorology and events.

Time. Two features related to time are identified, i.e., the hour of day and the day of a week. Fig. 6 shows the hourly check-out in the entire city in two weeks to confirm our claim. Besides the entire system, bike usage at a specific location also largely fluctuates by hour and day according

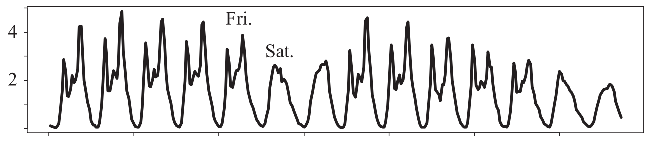

Fig. 6. Hourly check-out in citi bike system / thousand.

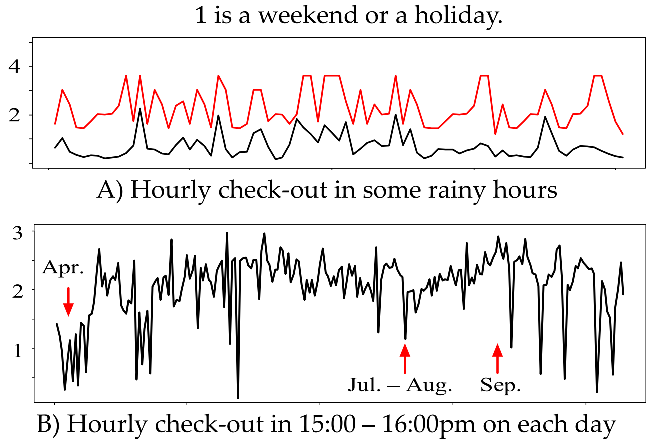

Fig. 7. Meteorology factors impact bike usage / thousand.

to its geographical properties, e.g., check-out around a resident area would be much larger in the morning rush hours than those in other periods; a station beside a tourism area would be more active at weekend or on a holiday than on weekday.

Def 5. Time Feature. A specific period t has a 2-D time feature $p_t = (p_t^1, p_t^2)$, whose two entries p_t^1 and p_t^2 respectively stand for the hour of day and the day of a week, $p_t^1 \in \{0,\ 1,\ \ldots,\ 23\}$ and $p_t^2 \in \{0,\ 1\}$ where $p_t^2 =\ 0$ means a weekday and $p_t^2 =\ 1$ is a weekend or a holiday.

Meteorology. Bike is a transportation model largely being impacted by meteorology. We identify three major related features, i.e., weather condition, temperature, and wind. Fig. 7A shows the hourly check-out in the entire system in some rainy hours (by a black curve) compared with the average check-out in the corresponding sunny hours (by a red curve). As we can see, there are much fewer users riding in rainy hours. Fig. 7B shows the hourly check-out in 15:00 - 16:00 pm from Apr. to Oct. 2016. As the temperature from Apr. to Jun. keeps increasing, more and more people choose to ride while a too high temperature in Jul. and Aug. prevents users from riding. Besides, wind is another important feature.

Def 6. Meteorology Feature. A specific period t has a 3-D meteorology feature $m_t = (m_t^1, m_t^2, m_t^3)$ whose three entries m_t^1, m_t^2 and m_t^3 respectively stand for the weather, temperature, and wind speed. According to historical data, weather conditions are summarized into four categories, i.e., clear-sunny, light-rain or snow, haze-fog, and heavy-rain or snow. Therefore $m_t^1 \in \{1,\ 2,\ 3,\ 4\}$ denotes a category index.

Event. Unexpected events also have significant effect to bike usage. As event data are too hard to collect nor measure, we adopt the recent check-out at each location to report whether there is any unexpected event there. This is because that events usually affect the bike usage at a location for a long period. If a location has anomalous check-out which is not caused by severe meteorology, we can assume that it is caused events, which may continue impacting the following periods here too. Fig. 8 shows the hourly check-out at a station in Lower Manhattan on each Saturday from Jul. to Sep. 2016. As we can see, the first three Saturdays in Aug. has anomalous bike usage because that is the time when *Summer Street* was held there. Besides, we can

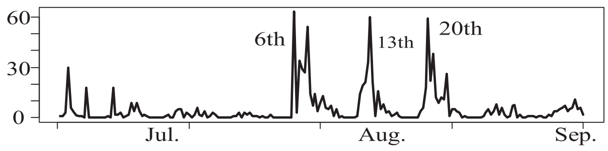

Fig. 8. *Summer street* in Aug. impacts the bike usage.

conclude that the activity impacted bike usage there for a long duration on these days.

Def 7. Event Feature. A specific location S in a specific period t has a 2-D feature $e_{St} = (e^1_{St},\ e^2_{St})$ whose two entries e^1_{St} and e^2_{St} respectively stand for the check-out at S in period $t-1$ and $t-2$. More recent periods can be incorporated although we only consider two here for simplicity.

As we can see, the time and meteorology features in a specific period are global and shared by locations in a hierarchy. However, event feature is local and specific to each location, because unexpected events usually impact some areas in a city instead of the whole one. Consequently, to each specific period t, we respectively extract a 6-D feature $f_{St} = (p^1_t, p^2_t, m^1_t, m^2_t, m^3_t,\ e^1_{St},\ e^2_{St})$ for each location S in the hierarchy.

3.2.2 Similarity Measurement

Time similarity. A discrimination function λ_0 is defined to measure the similarity between two time-features; $\lambda_0 = 1$ when they are the same, otherwise, $\lambda_0 = 0$.

Weather similarity. A symmetric matrix with six parameters as Eq. (4) is adopted to define the similarity function λ_1 for two weather conditions, e.g., to a weather pair *sunny* (the first category) and *heavy rainy* (the fourth category), their similarity is α_3 which is determined by the parameter in the first row and the fourth column.

$$\begin{bmatrix} 1 & \alpha_1 & \alpha_2 & \alpha_3 \\ \alpha_1 & 1 & \alpha_4 & \alpha_5 \\ \alpha_2 & \alpha_4 & 1 & \alpha_6 \\ \alpha_3 & \alpha_5 & \alpha_6 & 1 \end{bmatrix}. \tag{4}$$

Obviously, the more dissimilar two weather conditions are, the larger their parameter is, thus we add constraints: $\alpha_2 > \alpha_1 > \alpha_3; \alpha_5 > \alpha_4 > \alpha_1; \alpha_2 > \alpha_4 > \alpha_6; \alpha_5 > \alpha_6 > \alpha_3$.

Temperature Similarity. As temperature value is continuous, we adopt a Gaussian Kernel function λ_2 to measure the similarity between two temperatures as Eq. (5), where σ_1 is a parameter and named as the length-scale. Here $m^2_{t_1}$ and $m^2_{t_2}$ are the temperature values in t_1 and t_2.

$$\lambda_2\left(m^2_{t_1},\ m^2_{t_2}\right) = \exp\left(-\sigma_1^{-2} \times \left(m^2_{t_1} - m^2_{t_2}\right)^2\right). \tag{5}$$

Similar with temperature, we measure the similarity between two wind speeds by another Gaussian Kernel function λ_3 with parameter σ_2.

Event Similarity. Gaussian kernel function is again adopted to measure how similar two event-features are by Eq. (6) with parameter σ_3. Considering that the event feature is local and specific to each location, we adopt normalized difference as Eq. (7) to the kernel function.

$$\lambda^S_4\left(e_{St_1},\ e_{St_2}\right) = \exp\left(-\sigma_3^{-2} \times Z_{t_1t_2}\right) \tag{6}$$

$$Z_{t_1t_2} = \left|\left|e_{St_1} - e_{St_2}\right|\right|_2 \times \left|\left|e_{St_1} + e_{St_2}\right|\right|_2^{-1}. \tag{7}$$

Based on these estimated similarities related to each factor, we can estimate the final similarity between two periods t_1 and t_2 at a location S by Eq. (8).

$$\lambda(f_1, f_2|S) = \lambda_0 \times \left(\lambda_1 + \lambda_2 + \lambda_3 + \lambda^S_4\right). \tag{8}$$

Parameter Learning. Similarity estimation between two periods at a specific location is based on the parameters $\Phi_1 = \{\alpha_1,\ \alpha_2,\ \alpha_3,\ \alpha_4,\ \alpha_5,\ \alpha_6,\ \sigma_1,\ \sigma_2,\ \sigma_3\}$. However, we do not have the ground truth about how two periods are similar. Eq. (9) is adopted to approximate the similarity between t_1 and t_2 at location S.

$$sim(t_1,\ t_2|S) = 1 - \left|x^S_{t_1} - x^S_{t_2}\right| \times \left|x^S_{t_1} + x^S_{t_2}\right|^{-1} \tag{9}$$

Here $x^S_{t_1}$ and $x^S_{t_1}$ denote the check-out at S in t_1 and t_2.

Eq. (9) means that whether two periods are similar or not can be reported via their check-out. For a specific location S, we choose its most dissimilar period-pairs to obtain a set D^1_S and the most similar ones to obtain D^2_S. Each similar period-pair is also required to be selected from the same month because of the nonmonotonicity property across months as shown in Fig. 7B, where two periods with similar check-out in different months may have very different values. By assuming the monotonicity in each month, periods in the same month with similar check-out should have similar features. Parameters in Φ_1 can then be trained by Eq. (10), where $D_S = D^1_S \cup D^2_S$; δ is a discriminant function that $\delta = 1$ when $(t_1,\ t_2) \in D^1_S$ while $\delta = -1$ otherwise.

$$\min \sum\nolimits_S \sum\nolimits_{D_S} \delta(t_1,\ t_2) \times \lambda(f_1, f_2|S). \tag{10}$$

3.2.3 Similarity-Based Gaussian Process Regressor

A Gaussian Process Regressor (GPR) [3] can be considered as a distribution of functions without a specific form, which is commonly adopted to solve nonlinear regression problems. A GPR has two components, i.e., a mean function μ and a covariance function κ whose most common settings are as Eqs. (11) and (12); a and $\boldsymbol{\sigma}_4$ are parameters.

$$\mu\ (f) = 0 \tag{11}$$

$$\kappa\ (f_1,\ f_2) = a^2 \times \exp\left(-\boldsymbol{\sigma}_4^{-2} \times (f_1 - f_2)^2\right) \tag{12}$$

Our SGPR is a natural extension of GPR. To a specific location whose feature vector in the target period is f_{t+1}, we first generate its M-similar training set:

$$H_{t+1} = \left\{\left(f^{t+1}_1,\ x^{t+1}_1\right),\ \left(f^{t+1}_2,\ x^{t+1}_2\right),\ \ldots,\ \left(f^{t+1}_M,\ x^{t+1}_M\right)\right\}.$$

GPR is then adopted to predict the value at f_{t+1} based on H_{t+1} by the following equations [3], where $\hat{x}_{t+1}$ is the predicted check-out at the specific location in $t+1$.

$$\hat{x}_{t+1} \sim \boldsymbol{N}\left(\mu_{t+1},\ c^2_{t+1}\right) \tag{13}$$

$$\mu_{t+1} = \boldsymbol{k}_{t+1}\ (\boldsymbol{K}_{t+1})^{-1}\boldsymbol{X}_{t+1} \tag{14}$$

$$c^2_{t+1} = a^2 - \boldsymbol{k}_{t+1}(\boldsymbol{K}_{t+1})^{-1}(\boldsymbol{k}_{t+1})', \tag{15}$$

Here $(\boldsymbol{k}_{t+1})'$ is the transpose of $\boldsymbol{k}_{t+1}$; $\boldsymbol{k}_{t+1}$ and $\boldsymbol{X}_{t+1}$ and $\boldsymbol{K}_{t+1}$ are respectively obtained by Eqs. (16), (17) and (18).

$$\boldsymbol{k}_{t+1} = \left[\kappa\big(f_{t+1}, f_1^{t+1}\big),\ \kappa\big(f_{t+1}, f_2^{t+1}\big), \ldots,\ \kappa\big(f_{t+1}, f_M^{t+1}\big)\right] \tag{16}$$

$$\boldsymbol{X}_{t+1} = \left[x_1^{t+1},\ x_2^{t+1},\ \ldots,\ x_M^{t+1}\right]' \tag{17}$$

$$[\boldsymbol{K}_{t+1}]_{ij} = \kappa\Big(f_i^{t+1},\ f_j^{t+1}\Big),\ i, j\ =\ 1,\ 2,\ \ldots,\ M. \tag{18}$$

A SGPR has parameters $\Phi_2 = \{a, \boldsymbol{\sigma}_4\}$ which are learned by minimizing the total regression error on its validation set. Considering the specificity of each location in a hierarchy, we separately learn a SGPR for each location, otherwise the prediction accuracy may be compromised if different locations share a common SGPR.

3.2.4 GLS to Guarantee the HC Property

Obtained SGPR is adopted to predict check-out at each location in the target period. Each location can get a normal distribution whose mean is considered as the predicted check-out and variance is the prediction confidence. As these mean values cannot guarantee the HC property, GLS formulation is designed to collectively adjust them by Eq. (19), where ω is a parameter; $\boldsymbol{I}$ is an identity matrix.

$$\bar{\boldsymbol{X}}_{t+1} = \boldsymbol{X}^* \ \times \boldsymbol{H} \tag{19}$$

$$\boldsymbol{X}^* = \arg\min_{\boldsymbol{X}}\big(\boldsymbol{X}\boldsymbol{H}\boldsymbol{U}_{t+1}^{-1} - \boldsymbol{I}\big) \sum\nolimits_{t+1}^{-\omega} \big(\boldsymbol{X}\boldsymbol{H}\boldsymbol{U}_{t+1}^{-1} - \boldsymbol{I}\big)'. \tag{20}$$

Here $\boldsymbol{U}_{t+1}$ is a vector made up by the mean values corresponding to $(S_1^0,\ S_1^1,\ \ldots,\ S_m^1,\ S_1^2,\ \ldots,\ S_n^2)$ in $t+1$. Matrix $\boldsymbol{H}$ describes the hierarchy structure, e.g., to the hierarchy in Fig. 2, its structure matrix is as Eq. (21). Confidence matrix Σ_{t+1} is a diagonal one whose entries are the confidence values corresponding to $\boldsymbol{U}_{t+1}$. $\bar{\boldsymbol{X}}_{t+1}$ is the final prediction for the check-out hierarchy in $t+1$.

$$\boldsymbol{H}\ = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{21}$$

GLS is based on the prediction confidence at each location, i.e., the larger the confidence, the smaller the modification to the original prediction. By improving the original predictions at different-scale locations collectively via a mutual reinforcement way, GLS considers both global and local information in a system, improving the prediction accuracy and guarantee the HC property.

3.3 Inter-Cluster Transition Learning

Considering the causality between rent and return, predicting where and when a rented bike is to be returned is very important for check-in inference. We only discuss how to learn inter-cluster transition probability and ride duration here while the inter-station ones can be obtained in a same way.

3.3.1 Inter-Cluster Transition Matrices

From one cluster to another one, the transition probability in t describes how probable that a bike rented from the first cluster in t is to be returned to the second one. Only rent time is constrained to t while the return time can be a random possible value. Based on the transition probability introduced above, we define the inter-cluster transition matrix in t as TR_t whose entry $(TR_t)_{ik}$ describes the transition probability from cluster i to k in period t.

As we can imagine, the inter-cluster transition probability varies from time to time. Besides, transitions are impacted by events. However, meteorology does not obviously affect the transitions any more, e.g., when the weather is severe, most customers change to another transportation mode instead of changing their destinations. Therefore, to estimate the transitions, we only consider two factors, i.e., time and event.

Time factor. For simplicity, we assume that the transition probability is constant in each time slot in Table 2. Therefore, for each time slot, we estimate the transition matrices under normal conditions without unexpected events by maximum likelihood estimation [12] on the historical bike usage data generated in this time slot. Data generated under or not under events are not distinguished in this step as unexpected events rarely happen, thus almost not impact the estimation results. Inter-cluster transition matrix for a normal period is $\overline{TR}_t$.

Event factor. As discussed previously, event data are too hard to collect, thus we adopt transitions in the most recent period to report whether any unexpected event happened at any location. To a specific cluster S in t, whether its transition pattern is impacted by events is judged by Eq. 22 where $(\overline{TR}_t)_S$ denotes the row in matrix $\overline{TR}_t$ corresponding to cluster S while Tr_t^S is the practically observed transition probability at S in t; ω is a threshold.

$$\left\|\left(\overline{TR}_t^1\right)_S - Tr_t^S\right\|_2 > \omega \tag{22}$$

If Eq. (22) is satisfied, unexpected events happened at cluster S in t, thus we update the normal inter-cluster transition matrix $\overline{TR}_{t+1}$ by replacing its row corresponding to cluster S with its most recent transition probability Tr_t^S.

3.3.2 Inter-Cluster Ride Duration

Meteorology, time, and events are no longer important impacted factors to ride duration, e.g., traffic jam usually leads to longer travel time for vehicles but not bikes, severe weather conditions which may significantly impact the ride duration make users choose another transportation mode instead of spending much more time to ride. Therefore, we learn a constant ride duration distribution for each pair of clusters.

According to historical data, majority of the trips can be completed in less than one hour, thus we constrain the ride duration between any pair of clusters to a common interval $I = (0,\ 3600]$, where second is the unit. Considering two clusters S_u and S_v, we discretize I into tiny time windows

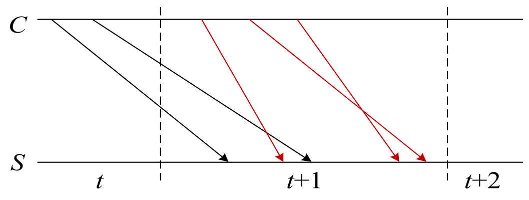

Fig. 9. Components making up check-in.

as $\{I_1,\ I_2,\ \ldots,\ I_T\}$ and obtain a multinomial distribution D_{uv} to measure the ride duration between S_u and S_v. The $i-$th entry in D_{uv} means the probability that a trip between cluster S_u and S_v may take $time \in I_i$ seconds. Each tiny time window is set as 1 minute in our work.

4 Online Process

Online process to predict the check-out at each location is intuitive. Although we can also adopt HCP to predict check-in, it does not consider the causality between rent and return. A T-INF method is designed to predict the check-in at each location. We only elaborate how to infer that for each cluster here while that for station is the same.

Denote the target period as $t+1 = (\tau, \tau+1]$, check-in to cluster S in $t+1$ is made up by two parts as Fig. 9.

- Black arrows stand for the bikes which have been rented before τ and are to be returned to S in $t+1$. Assume those bikes which are rented before τ and have not been returned yet as $B = \{b_1,\ b_2, \ldots\}$. To the jth bike in B, its origin cluster is denoted as S^j. As the maximum ride duration is constrained to one hour, each bike in B is surely to be returned in $t+1$. Based on inter-cluster transition probability matrix, we estimate the expectation of bikes in B that are to be returned to S in $t+1$ by Eq. (23).

$$E_{t+1}^{S1} = \sum\nolimits_{b_j \in B} p(S^j \to S|TR_t), \tag{23}$$

- Red arrows denote the bikes which are to be rented in $t+1$ and are to be returned to cluster S before $\tau+1$. Dividing the target period $(\tau, \tau+1]$ into tiny time windows as $(\tau, \tau+\Delta]$, $(\tau+\Delta,\ \tau+2\times\Delta]$, $\ldots$, $(\tau+1-\Delta,\ \tau+1]$ where Δ is a tiny time length such as 1 minute. Assuming the predicted check-out at a random cluster Λ in period $t+1$ is $\bar{x}_{t+1}^{\Lambda}$, we can estimate its check-out in each tiny time window as $\bar{x}_{t+1}^{\Lambda} \times \Delta$. A bike rented from Λ in the rth time window has $1 - r \times \Delta$ time left to be returned to S before $\tau+1$, thus its corresponding return probability is calculated by Eq. (24). Therefore, expectation of bikes to be rented in $t+1$ and returned to S in $t+1$ can be estimated by Eq. (25).

$$P\ (r,\ S,\ t+1) =\ p(\Lambda \to S|TR_{t+1}) \times \int_0^{1-r\times\delta} D_{\Lambda\to S} \tag{24}$$

$$E_{t+1}^{S2} = \sum\nolimits_{\Lambda}\sum\nolimits_{r} \bar{x}_{t+1}^{\Lambda} \times \Delta \times P(r,\ S,\ t+1), \tag{25}$$

Consequently, the check-in at S in $t+1$ can be predicted as the sum of these two expectations as $\bar{y}_{t+1}^{S} = E_{t+1}^{S1} + E_{t+1}^{S2}$.

TABLE 3
Real-World Datasets

Data Sources			NYC	D.C.
Bike Data	# Stations		389	334
	# Records		6,075,887	1,737,461
Meteorology Data	Weather (# hours)	①	4,520	4,712
		②	214	337
		③	361	43
		④	41	44
	Temperature / °C		(−3, 36)	(3, 37)
	Wind speed / mph		(0, 21)	(0, 24)

5 Evaluation

Experiments are conducted to confirm the effectiveness of our model. Each period is set as 1 hour. As there is not a ground truth to the real check-out nor check-in at each station in each period, methods in previous works [10], [13], [26] based on station status data[2] are adopted to approximate them by Eq. (26). Here o_t^S means how many bikes have been rented from S in t and TL_t^S is the time length in period t when station S has available bikes. Check-in is approximated similarly.

$$x_t^S = o_t^S \ \times |t| \times \left(TL_t^S\right)^{-1} \tag{26}$$

5.1 Metrics and Baselines

Real-world data from Citi Bike system in New York City and Capital Bikeshare system in Washington D.C. are adopted to evaluate our model.

Bike usage data contain many historical trips describing where and when each bike is rented and returned. Meteorology data include records describing the weather condition, temperature and wind speed in each period. Our experiments are only conducted to the principle systems which cover the urban centers, because station-status data are only available there. More data details are summarized in Table 3 where ① to ④ respectively denote the four weather categories. Historical data from Apr. to Aug. 2016 are set as training data while those in Sep. are adopted for validation. Model is tested on data in Oct.

Metric. Mean Absolute Error Rate (MAER) and Root Mean Squared Error Rate (RMSER) are respectively defined as Eqs. (27) and 28, where $M^* = \sum_t \sum_S 1$ is the testing data size, x_t^S is the ground truth of check-out or check-in at S in t while $\bar{x}_t^S$ is the corresponding prediction.

$$e_1 = \frac{1}{M^*}\sum\nolimits_t \sum\nolimits_S \left|\bar{x}_t^S - x_t^S\right| \ \times \frac{1}{x_t^S} \tag{27}$$

$$e_2 = \sqrt{\frac{1}{M^*}\sum\nolimits_t \sum\nolimits_S \left(\left(\bar{x}_t^S - x_t^S\right) \times \frac{1}{x_t^S}\right)^2}\ . \tag{28}$$

2. https://groups.google.com/forum/#!forum/citibike-hackers

TABLE 4
HC Property Comparison

Model	SHA	SARIMA	ARIMAX	WKNN
HC Property	✓	×	×	×
Model	LR	MSP	UP	HCP
HC Property	×	✓	✓	✓

Geo-Clustering algorithm (GC). Geo-Clustering step in AdaTC algorithm. GCI means the parameter ρ_1 in Eq. (2) is set to zero while $\rho_1 = +\infty$ for GCII.

Seasonal Historical Average (SHA). Bike usage at a location is predicted by the average of historical usage there in the same hour of day and on the same day of a week.

Seasonal ARIMA (SARIMA) [1]. SARIMA is a commonly used prediction method for time series.

ARIMAX. SARIMA with exogenous variables that are discussed in our paper.

WKNN [10]. Given a location and $t+1$, we select its very similar periods and estimate the bike usage there and then by the weighted average over these similar periods.

Linear Regressor (LR). Linear regression on the impacted factors discussed previously.

Multi-Similarity based Prediction (MSP) [9]. Model in conference paper, which is a top-down hierarchy time series prediction strategy.

UP. UP is a bottom-up hierarchy time series prediction strategy, which first predicts values at leaves and then adds certain of them to obtain those in upper levels.

If the HC property can be guaranteed by each prediction model is summarized in Table 4.

5.2 Experiment Results

Experiments on data from Citi Bike are detailly analyzed and discussed; then we show some prediction results on data from Capital Bikeshare to confirm that our model is widely applicable.

5.2.1 *Clustering Algorithm*

AdaTC algorithm has three parameters ρ_1, K^1, and K^2. Parameter ρ_1 is to tradeoff the geographical distance and check-out dissimilarity, while K^1 and K^2 respectively determine how many groups the stations are clustered into in Geo-Clustering and T-Clustering step. For simplicity, we tune them one by one.

Conduct Geo-Clustering by setting $\rho_1 \in \{0,\ 0.1,\ \ldots,\ 1.5\}$ and $K^1 \in \{40,\ 41,\ \ldots,\ 60\}$. Clustering results are then measured by the average inner-cluster geographical distance and check-out dissimilarity. Figs. 10A and 10B show these two metrics respectively averaged over $K^1 \in \{40,\ \ldots,\ 60\}$ for each specific ρ_1 value. As we can see, an increasing ρ_1 makes the distance increase while the dissimilarity to decrease. This is very reasonable considering its tradeoff role. Parameter value $\rho_1 = 0.4$ is chosen not only because it is an elbow point in B) but also that its corresponding distance in A) is acceptable.

Set $\rho_1 = 0.4$ and conduct Geo-Clustering. Estimate the above two metrics for each $K^1 \in \{40,\ \ldots,\ 60\}$ and respectively show them in Figs. 10 C and 10D. As we can see, there is not an obvious elbow point. However, as a large value for K^1 may compromise the regularity of bike usage at each cluster, we select the parameter value $K^1 = 50$ to ensure that each cluster is neither too large, thus can constrain the inner-cluster distance, nor too tiny to ensure that the bike usage in each cluster is regular.

To tune another parameter K^2, a new metric is required to measure how concentrated the inter-cluster transitions are. For simplicity, we measure this by first summing the largest one hundred entries in the inter-cluster transition matrix in each time slot, and then average these sums over time slot, to obtain the final concentration metric, the larger, the better. Setting $\rho_1 = 0.4$ and $K^1 = 50$, AdaTC is conducted for each $K^2 \in \{20,\ 21,\ \ldots,\ 40\}$. How K^2 impacts the inner-cluster distance, check-out dissimilarity, and transition concentration are respectively shown in Figs. 10E, 10F, and 10G. As we can see, when K^2 increases, inner-cluster distance keeps decreasing. However, there is a tradeoff between inner-cluster dissimilarity and transition concentration, i.e., an increasing K^2 makes the dissimilarity increase although the concentration increases too. Considering this tradeoff, parameter value $K^2 = 30$ is chosen to guarantee that the dissimilarity and concentration are both acceptable.

However, different parameter values can be selected according to specific requirements and targets of the practical problem, e.g., if making the check-out patterns in each cluster similar is more important, we can decrease K^2 to

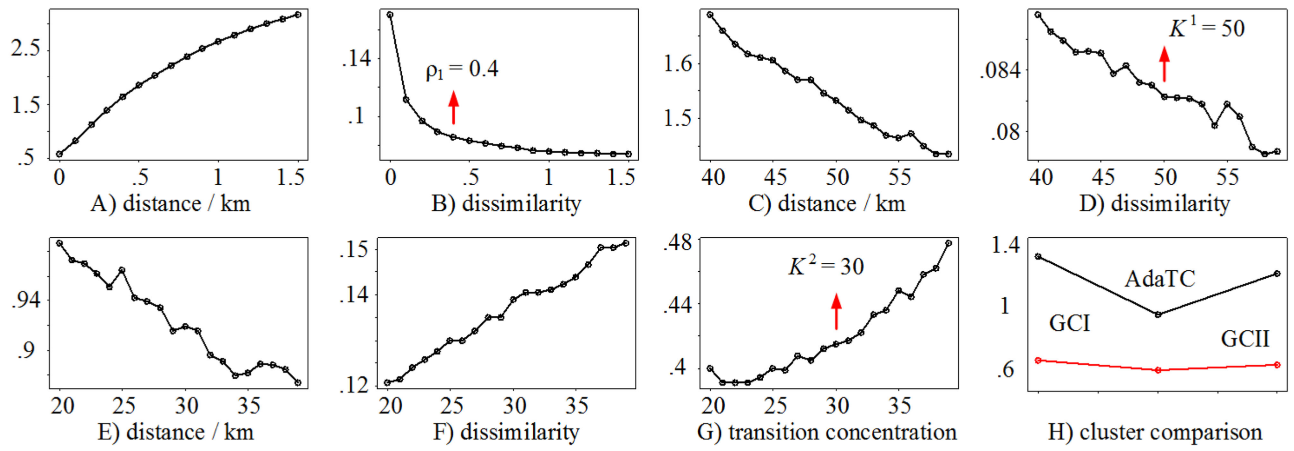

Fig 10. Parameter tuning and clustering algorithm comparison.

TABLE 5
Clustering Result Comparison

Algorithm	GCI	AdaTC	GCII
Distance / km	0.53	**0.91**	3.66
Dissimilarity	0.17	**0.14**	0.07
Concentration	0.39	**0.41**	0.23

reduce the dissimilarity by sacrificing some transition concentration.

Obtaining these three parameters, we conduct AdaTC, GCI and GCII respectively. Clustering results are summarized in Table 5. As we can see, GCI can guarantee stations in one cluster to be close but cannot reduce their check-out dissimilarity. GCII ensures stations in one cluster have similar check-out patterns while performing the worst considering the other two metrics. AdaTC is a tradeoff of them, making every metric acceptable. Besides, AdaTC can reduce cluster-level prediction error, which is to be discussed in Section 5.2.3.

5.2.2 Check-Out and Check-In Prediction Results

Check-out at each location is predicted by HCP and *seven* baselines. Table 6 summarizes the prediction error of each model; values highlighted by grey correspond to anomalous periods, which are defined as those whose weather condition is not in the first category sunny; other values correspond to all testing periods; – means the model is not applicable here or the prediction results here are directly adopted from another baseline, thus is not necessary to list again. As we do not require much for station-level prediction accuracy, results at cluster and in the whole city are discussed first. Some results of station-level prediction are to be shown later.

As shown in Table 6, SHA and SARIMA perform much worse than others as they only consider the hour of day and day of a week as impacted factors but not the meteorology nor events. WKNN is similar with SHA as they both predict by average value. However, as WKNN considers more external impacted factors, it largely outperforms SHA, confirming that the factors we discussed previously are reasonable and really impact the bike usage in a system. ARIMAX and LR, which consider those external factors too, performs much better than WKNN. As both them are regression method, they are more capable to capture how external factors impact bike usage, especially for those anomalous periods whose factors change largely. Comparing ARIMAX and LR, we conclude that LR is better, especially for suddenly change points in the time series, e.g., the anomalous ones; this may be because that ARIMAX cannot capture the influence from events but cares more about seasonal periods.

TABLE 6
Prediction Error for Check-Out in Citi Bike

Model	City		Cluster		City		Cluster	
	MAER	RMSER	MAE	RMSER	MAE	RMSER	MAE	RMSER
SHA	0.4	1.05	0.73	2.21	1.77	2.7	2.47	5.43
SARIMA	0.41	1.09	0.75	2.18	1.84	2.84	2.57	5.32
ARIMAX	0.27	0.67	0.55	1.42	1.09	1.71	1.57	3.27
WKNN	0.3	0.85	0.62	1.78	1.4	2.2	2.01	4.32
LR	0.25	0.44	0.61	1.28	0.56	0.9	1.19	2.39
MSP	-	-	0.52	1.09	-	-	0.93	1.94
UP	0.26	0.36	0.64	1.33	0.48	0.62	1.13	2.19
HCP	**0.19**	**0.28**	**0.51**	**0.99**	**0.4**	**0.51**	**0.79**	**1.39**

TABLE 7
Prediction Error for Check-In in Citi Bike

Model	City		Cluster		City		Cluster	
	MAER	RMSER	MAE	RMSER	MAE	RMSER	MAE	RMSER
SHA	0.37	0.9	0.66	1.79	1.5	2.31	2.09	4.16
SARIMA	0.39	1.03	0.7	1.95	1.71	2.66	2.29	4.64
ARIMAX	0.28	0.7	0.53	1.4	1.12	1.78	1.45	3.19
WKNN	0.28	0.74	0.56	1.49	1.2	1.9	1.69	3.51
LR	0.26	0.38	0.55	1.08	0.49	0.69	0.96	1.75
MSP	-	-	0.5	1.01	-	-	0.82	1.61
UP	0.27	0.35	0.66	1.37	0.46	0.58	1.12	2.16
HCP	**0.21**	**0.29**	0.48	0.92	**0.39**	**0.54**	0.78	1.5
TINF	-	-	**0.43**	**0.69**	-	-	**0.6**	**0.95**

MSP, UP and HCP can guarantee the HC property in a hierarchy, each of which corresponds to one kind of hierarchy time series prediction strategy, i.e., MSP is top-down; UP is bottom-up; HCP is optimal combination. MSP is similar with WKNN, i.e., both them predict by weighted average. However, MSP does not predict cluster-level nor station-level bike usage directly, but by allocating the total usage in the whole city to each cluster or station. As we can see, MSP not only beats WKNN but also outperforms ARIMAX and LR, confirming that leveraging the total bike usage to bound those at each cluster and station is useful. However, UP does not perform satisfactorily, which we think is also reasonable. As bike usage at each station is too fluctuating to predict accurately, it is almost impossible to accurately predict those in higher levels by adding the certain of them. Our proposed model HCP which predicts bike usage at each location by an optimal combination strategy performs the best. As HCP considers both global information, i.e., the total usage in the city, and local information, i.e., the usage at each station, via a mutual reinforcement way based on their confidence, it is reasonable that it beat MSP and UP. Besides improving prediction accuracy, HCP can guarantee the HC property, thus bring extra practical system management benefit.

Check-in prediction error of each model for all testing periods and the anomalous ones are summarized in Table 7. Comparison and discussion to Table 7 are similar with those to Table 6. Main difference is that there is another method, i.e., TINF. As we can see, TINF outperforms the seven baselines and HCP significantly, for both common and anomalous periods; this is because TINF additionally considers the causality between check-out and check-in.

Station-level prediction MAER are shown in Table 8. Although our model has already improved the station-level prediction accuracy, they cannot be as good as those at cluster. According to historical data in Citi Bike, 70 percentages of the hourly station-level check-out is not larger than 5. Besides, considering its largely fluctuating property, it is reasonable that station-level check-out is almost impossible to be predicted accurately. On the other hand, station-level

TABLE 8
Station-Level Prediction MAER in Citi Bike

Model	SHA	SARIMA	ARIMAX	WKNN	LR	MSP	UP	HCP	TINF
Check-out	0.85	0.85	0.68	0.76	0.82	0.69	0.69	**0.68**	-
Check-in	0.73	0.74	0.6	0.64	0.69	0.62	0.59	0.6	**0.54**

prediction is only referred to when conducting complementary mini-scale reposition inner cluster, whose average distance is hundreds of meters which is short to a reposition vehicle. In practical system operation, mini-scale reposition can largely depend on real-time monitoring considering its high efficiency; therefore, we do not discuss station-level bike usage prediction much.

5.2.3 Prediction Based on Different Clustering Results

To confirm that AdaTC is a good clustering algorithm which can help to improve prediction accuracy, we do:

- Adopt HCP to predict cluster-level check-out respectively based on clustering results obtained by GCI, AdaTC and GGII.
- Adopt TINF to predict cluster-level check-in respectively based on the above three clustering results.

Fig. 10H shows the prediction RMSER for check-in, where the red curve is for all testing periods while the black one is for anomalous periods. As we can see, AdaTC outperforms GCI and GCII. This is reasonable as none of GCI nor GCII considers the transition concentration among clusters, thus cannot predict the inter-cluster transition accurately, which is important in TINF.

Considering space limit, we ignore the figure for check-out prediction, but summarize their results as GCII->AdaTC>GCI where > means *better than*; this is because that GCII can guarantee the regularity of check-out at each cluster, thus making prediction easier. However, AdaTC can also guarantee the regularity to some extent. Considering both check-out and check-in, AdaTC is the most appropriate clustering algorithm.

5.2.4 Prediction Results on Data from DC

To confirm that our model is widely applicable, experiments on real-world data from Capital Bikeshare are conducted. Stations in DC are clustered into 40 groups.

TABLE 9
Cluster-Level Prediction Error in Capital Bikeshare

Model	Check-out				Check-in			
	MAER	RMSER	MAE	RMSER	MAE	RMSER	MAE	RMSER
SHA	0.63	1.76	2.64	6.11	0.59	1.83	2.59	6.49
SARIMA	0.62	1.72	2.59	5.95	0.58	1.77	2.53	6.26
ARIMAX	**0.51**	1.04	1.06	2.51	0.48	1.05	1.09	2.76
WKNN	0.58	1.48	2.15	4.86	0.55	1.53	2.08	5.2
LR	0.68	1.26	0.76	1.41	0.62	1.09	0.7	1.43
MSP	0.59	1.08	0.73	1.39	0.55	1.02	0.72	1.5
UP	0.61	1.12	0.74	1.41	0.58	1.05	0.73	1.52
HCP	0.55	**0.94**	**0.71**	**1.02**	0.54	0.99	0.73	1.34
TINF	-	-	-	-	**0.45**	**0.66**	**0.66**	**0.87**

Cluster-level check-out and check-in prediction error are summarized in Table 9. As we can see, HCP for check-out and TINF for check-in outperform other prediction baselines significantly too. Results for city-level and station-level prediction are not shown here for space limit, however, they also support the conclusion that our model largely outperform the baselines.

6 Related Work

Related studies are summarized into two groups, i.e., bike usage mining and system operation.

Jon Froehlich et al. [4] provided a spatio-temporal analysis to station-level bike usage, i.e., the check-out pattern and available-bike pattern at each station, on data from Bicing in Barcelona. The paper also identified shared behaviors across stations and how these behaviors are related to locations, neighborhoods and time by a clustering technique. Besides, the authors compared four simple prediction models, i.e., last value, historical mean, historical trend and Bayesian network, to predict the number of available bikes in a coming period. Kaltenbrunner et al. [5] also worked on data from Bicing. They detected the spatio-temporal mobility pattern in a system to predict the number of available bikes at each station. Borgnat et al. [6] provided spatio-temporal analysis to Velo'v which is a bike-sharing system in Lyon. The authors predicted the total check-out of a system in each hour of a day by a combination model, i.e., a non-stationary amplitude for a given day added with a fluctuation at each specific hour in a day. Ride duration and inter-station transition are also investigated. Vogel et al. [7] predicted the check-out at each station by adopting a time series analysis method similar with [6] but they also considered the effect from weather factors. Yoon et al. [8] presented a personal journey advisor application to a bike-sharing system. One main technique in their work to predict the number of available bikes and docks at each station is an ARIMA model which considers spatial interaction and temporal factors. The authors evaluated their model on Dublin system to confirm a good performance of their journey advisor. Raviv et al. [22] proposed an inventory model to design the bike inventory for each station in each period thus to minimize the customer loss and dissatisfaction there. Numerical study on real-world data from Capital Bike-Share is presented in their work. Their system simulation is also based on station-level check-out and check-in prediction. On the historical bike usage data from Citi Bike, Li et al. [9] proposed a more flexible model which considers two kinds of external impacted factors, i.e., the time and the meteorology, to predict the check-out and check-in at each station cluster. Liu et al. [10] and Yang et al. [11] further developed this work later to predict the station-level bike usage. Instead of targeting at bike usage prediction, Chen et al. [14] proposed a dynamic cluster-based model to predict which cluster is to be over-demand in the future, thus more attention can be paid to them in practical system operation.

Studies in system operation mainly focus on bike reposition including static reposition, dynamic reposition and user-based reposition, among which the first two are conducted by system operators while the third one is based on bike users. Static bike reposition means the reposition process is conducted when the system is closed or in the night while the dynamic one means repositioning when system operates. To solve these two problems, optimization models [17], [16], [19], [20], [23], [24] based on check-out and check-in prediction are usually adopted. The objectives are to minimize the customer loss, customer dissatisfaction or the total reposition distance, etc. Li et al. [13] proposed a reinforcement learning based model to conduct dynamic bike reposition in Citi Bike to minimize the total customer loss in a long period. User-based reposition approaches [24], [27], [28], [29] mean that system operators offer money to users and ask them to rent or return at specific stations. User-based strategies are more flexible in practical system operation although it is hard to estimate how much money to offer.

7 Conclusions

Our work solves the citywide bike usage prediction problem in a bike-sharing system. An AdaTC clustering algorithm is first proposed to cluster stations in a system into min-clusters. A HCP model considering external impacted factors is then proposed to predict the check-out throughout a city. HCP can guarantee the HC property of a hierarchy thus largely improves the prediction accuracy. To predict the check-in at each location, we further design a TINF method to infer how many bikes are to be returned there in the future. TINF considering the causality between rent and return, thus can obtain better performance than baselines. Experiments on real-world data are conducted to confirm the effectiveness of our model. For future work, we mean to generalize our model to consider more spatio-temporal data sources and deeper hierarchies. Real event data can be collected and considered too.

References

[1] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. *Times Series Analysis: Forecasting and Control*, Hoboken, NJ, USA: Wiley, 2015.

[2] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to Linear Regression Analysis*, Hoboken, NJ, USA: Wiley, 2012.

[3] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, Cambridge, MA, USA: MIT Press, 2006.

[4] J. Froehlich, J. Neumann, and N. Oliver, "Sensing and predicting the pulse of the city through shared bicycling," in *Proc. 21st Int. Jiont Conf. Artif. Intell.*, 2009, pp. 1420–1426.

[5] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Condian, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive Mobile Comput.*, vol. 6, pp. 455–466, 2010.

[6] P. Borgnat, P. Abry, P. Flandrin, C. Robardet, J. B. Rouquier, and E. Fleury, "Shared bicycles in a city: A signal processing and data analysis perspective," *Adv. Complex Syst.*, vol. 14, pp. 415–438, 2011.

[7] P. Vogel and D. Mattfeld, "Strategic and operational planning of bike-sharing systems by data mining–a case study," in *Proc. Int. Conf. Comput. Logistics*, 2011, pp. 127–141.

[8] J. W. Yoon, F. Pinelli, and F. Calabrese, "Cityride: A predictive bike sharing journey advisor," in *Proc. IEEE 13th Int. Conf. Mobile Data Manage.*, 2012, pp. 306–311.

[9] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proc. 23rd SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2015, Art. No. 33.

[10] J. Liu, L. Sun, W. Chen, and H. Xiong, "Rebalancing bike sharing systems: A multi-source data smart optimization," in *Proc. SIGKDD*, 2016, pp. 1005–1014.

[11] Z. Yang, J. Hu, Y. Shu, P. Cheng, J. Chen, and T. Moscibroda, "Mobility modeling and prediction in bike-sharing systems," in *Proc. ICMSAS*, 2016, pp. 165–178.

[12] F. W. Scholz, "Maximum likelihood estimation," *Encyclopedia of Statistical Sciences*, Wiley, New York, 1985.

[13] Y. Li, Y. Zheng, and Q. Yang, "Dynamic bike reposition: A spatio-temporal reinforcement learning approach," in *Proc. SIGKDD*, 2018, pp. 1724–1733.

[14] L. Chen, D. Zhang, L. Wnag, D. Yang, X. Ma, S. Li, Z. Wu, G. Pan, T. Nguyen, and J. Jakubowicz, "Dynamic cluster-based over-demand prediction in bike sharing systems," *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, 2016, pp. 841–852.

[15] S. Ghosh, P. Varakantham, Y. Adulyasak, and P. Jaillet, "Dynamic repositioning to reduce lost demand in bike sharing systems," *J. Artif. Intell. Res.*, vol. 58, pp. 387–430, 2017.

[16] S. Ghosh, M. Trick, and P. Varakantham, "Robust repositioning to counter unpredictable demand in bike sharing systems," in *Proc. 25th Int. Joint Conf. Art. Int.*, 2016, pp. 3096–3102.

[17] J. Shu, M. C. Chou, Q. Liu, C. Teo, and I. Wang, "Models for effective deployment and redistribution of bicycles within public bicycle-sharing systems," *Oper. Res.*, 2013, vol. 6, pp. 1346–1359.

[18] E. O'Mahony and D. B. Shmoys, "Data analysis and optimization for (citi) bike sharing," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 687–694.

[19] J. Schuijbroek, R. Hampshire, and W. V. Hoeve, "Inventory rebalancing and vehicle routing in bike sharing systems," *Eur. J. Oper. Res.*, vol. 257, pp. 992–1004, 2017.

[20] E. Miller-Hooks and R. Nair, "Fleet management for vehicle sharing operations," *Trans. Sci.*, vol. 45, pp. 524–540, 2010.

[21] R. Nair, E. Miller-Hooks, R. C. Hampshire, and A. Busic, "Large-scale vehicle sharing systems: Analysis of velib," *Int. J. Sustainable Transp.*, vol. 7, pp. 85–106, 2013.

[22] T. Raviv, O. Kolka, "Optimal inventory management of a bike-sharing station," *IIE Trans.*, vol. 45, pp. 1077–1093, 2013.

[23] T. Raviv, M. Tzur, and I. A. Forma, "Static repositioning in a bike-sharing system: Models and solution approaches," *EURO J. Transp. Logistics*, vol. 2, pp. 187–229, 2013.

[24] S. Ghosh and P. Varakantham, "Incentivizing the use of bike trailers for dynamic repositioning in bike sharing systems," in *Proc. 27th Int. Conf. Autom. Planning Scheduling*, 2017.

[25] J. Liu, Q. Li, M. Qu, W. Chen, J. Yang, H. Xiong, H. Zhong, and Y. Fu, "Station site optimization in bike sharing systems," in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 883–888.

[26] J. Liu, L. Sun, Q. Li, J. Ming, Y. Liu, and H. Xiong, "Functional zone based hierarchical demand prediction for bike system expansion," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 957–966.

[27] D. Chemla, F. Meunier, T. Pradeau, R. W. Calvo, and H. Yahiaoui. Self-service bike sharing systems: Simulation, repositioning, pricing, Tech. Rep. hal-00824078, 2013.

[28] C. Fricker and N. Gast, "Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity," *Euro J. Transp. Logistics*, vol. 5, pp. 261–291, 2016.

[29] A. Singla, M. Santoni, G. Bartok, P. Mukerji, M. Meenen, and A. Krause, "Incentivizing users for balancing bike sharing systems," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 723–729.

[30] W. Billy, "Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling," *Transp. Res. Rec.: J. Transp. Res. Board*, pp. 194–200, 2001.

Yexin Li is currently working toward the PhD degree in Computer Science and Engineering department, the Hong Kong University of Science and Technology. She was an intern in Urban Computing group, Microsoft Research Asia. Her research interest focuses on reinforcement learning, Gaussian process and spatio-temporal data mining.

Yu Zheng is the vice president and chief data scientist at JD Finance, passionate about using big data and AI technology to tackle urban challenges. He is the general manager of the Urban Computing Business Unit and serves as the director of the Urban Computing Lab at JD Group. Before Joining JD Group, he was a senior research manager at Microsoft Research. He is also a chair professor at Shanghai Jiao Tong University, an adjunct professor at the Hong Kong University of Science and Technology. He currently serves as the editor-in-chief of *ACM Transactions on Intelligent Systems and Technology* and has served as chair on more than 10 prestigious international conferences, e.g., as the program co-chair of ICDE 2014 (Industrial Track), CIKM 2017 (Industrial Track), IJCAI 2019 (Industrial Track). In 2013, he was named one of the Top Innovators under 35 by MIT Technology Review (TR35) and featured by Time Magazine for his research on urban computing. In 2014, he was named one of the Top 40 Business Elites under 40 in China by Fortune Magazine. In 2017, he is honored as an ACM distinguished scientist. He is a senior member of the IEEE.