E-Sharing: Data-driven Online Optimization of Parking Location Placement for Dockless Electric Bike Sharing

Pengzhan Zhou[1], Cong Wang[2], Yuanyuan Yang[1], and Xin Wei [2]

[1]*Dept. of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA*

[2]*Dept. of Computer Science, Old Dominion University, VA 23529, USA*

***Abstract*—The rise of dockless electric bike sharing becomes a new urban lifestyle recently. More than just the first-and-last mile, it offers a new modality of green transportation. However, in addition to the traditional re-balance and overcrowding problems, it also brings new challenges to urban management and maintenance. Due to the safety risks of batteries, customers are regulated to park at designated locations, which potentially causes dissatisfaction and customer loss. Meanwhile, service providers should charge those scattering low-energy batteries in time. To address these issues, we propose E-sharing, a two-tier optimization framework that leverages data-driven online algorithms to plan parking locations and maintenance. First, we balance the user dissatisfaction and the number of parking locations by minimizing their sum. To account for real-time dynamics while not losing track of the historical optimality, we propose an online algorithm based on its near-optimal offline solution. Second, we develop an incentive mechanism to motivate users to aggregate low-battery bikes together, saving the cost of bike charging. Our experiment based on the public dataset demonstrates that the online algorithm can minimize the cost from the conflicting objectives and incentive mechanism further reduces the maintenance cost by 47%.**

***Index Terms*—Smart transportation, mobile computing, electric bike sharing, online optimization, big data**

I. Introduction

Dockless bike-sharing has redefined the short-term bicycle rental business in China, and quickly expanded globally. With a GPS-based mobile app, users enjoy the flexibility to park almost anywhere. Being a green solution to the first-and-last mile problem, bike-sharing is promising to reshape the car-centric urban transportation in the future smart cities. Users can pick up a bike, ride for a while and drop at anywhere they want. Studied in [1], an average ride usually lasts within three miles. However, what if users want to ride extra miles? Those "human-powered" bicycles barely satisfy such emerging demands, especially for senior riders with physical limitations, or on hilly terrain like San Francisco. To fill this market niche, new start-up companies begin to offer an electric boost to the bicycles, i.e., the dockless *electric bike sharing* (E-bike sharing) [2]. As shown in Fig. 1, companies like XQBike [3], Lime [4], Qee Bike [5] and Bird Scooter [6] are quickly spreading regionally in the U.S., Europe and China. Powered by electricity, cyclists can enjoy an effortless longer ride, reach their destinations in the shortest route and save the time waiting for crowded transportation during rush hours.

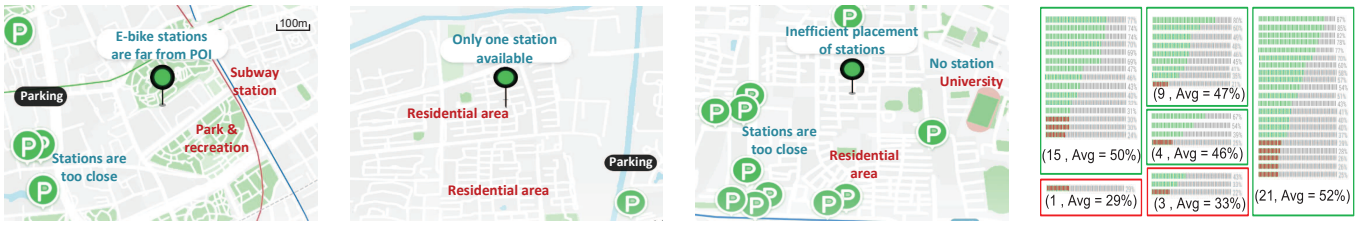
Unfortunately, the free-floating form of bike-sharing caters to the customer's needs at a cost of public space, urban management and maintenance [9]–[11]. E.g., the peak time drop-off at the connections of transportation leads to a parking turmoil and puts pressure on the public space and management. E-bike sharing faces more restrictions and challenges due to the risk of fire of lithium batteries [7]. In contrast to the traditional bikes that can be parked anywhere, for safety precautions, many municipalities do not allow E-bikes to park uncoordinately at random locations. Some cities even impound E-bikes [8]. As a result, service providers restrict customers to only drop off at the designated locations. Such regulation could easily dampen users' enthusiasm if the locations are far from their final destinations. To make the system sustainable, energy replenishment is also crucial. Building dedicated charging stations is expensive and unscalable [2]. Thus, the current methods pursue an infrastructureless solution that allows users to park at some pre-determined locations and hires operators to replenish the low-energy E-bikes. Hence, *maintenance* becomes a new challenge in the E-bike sharing systems.

(a) (b) (c)

Fig. 1: The emerging E-bike sharing around the world (a) XQBike [3] (b) Qee Bike [5] (c) Bird scooter [6].

Existing works of bike sharing mainly focus on the problems of re-balancing [9]–[11], location optimization [13]–[16] and usage prediction [17]–[19]. Re-balancing employs a truck or trike to reposition bikes from the congested locations to the empty ones. A static re-balancing problem is solved using branch-and-cut algorithm [9]. Policy iteration method is proposed to minimize the long-term customer loss due to unbalanced bike distributions [10]. An incentive strategy is designed in [11] to ask customers to help the re-balancing process. In [12], customers are incentivized heterogenously to park e-bikes at desiring locations considering varying difficulties. Another line of works attempt to optimize the location of parking spots by mining human mobility, point-of-interest data [13]–[15] or crowdsourced surveys [16]. These methods can be used to determine candidate locations but hardly capture the real-time situations. To build better re-balancing mechanisms and expand business, prediction models are constructed based on multi-modal data features using machine learning techniques [17]–[19].

Although these mechanisms are effective, they are not readily available to tackle the high maintenance cost in E-bike sharing. In particular, building an efficient system requires solutions to the following problems. First, because of restricted parking, how to minimize the number of parking locations



(a) (b) (c) (d)

Fig. 2: Existing problems in a representative E-bike sharing app [3] (a) stations are far from subway and recreation. (b) not enough stations near the residential area. (c) inefficient spatial coverage near the university (d) live energy status at some stations.

while considering user dissatisfaction. Second, unlike the existing methods that are based on offline data analysis, how to adapt to the evolving, new requests and make decisions online. Finally, how to optimize bike charging based on user behaviors.

To address these issues, we propose *E-Sharing*, a two-tier optimization framework that makes online decisions but keeps an eye on the historical patterns. The first goal is to optimize maintenance cost by minimizing the number of parking locations. This alleviates the safety risks, impact on public space, and makes maintenance more efficient (fewer locations the operator should handle). Meanwhile, user dissatisfaction should be also considered while assigning them to the closest parking given their destinations. We take both variables into consideration and formulate an optimization problem. An offline algorithm is first introduced that can make near-optimal decisions based on historical data. Under its guidance, we further develop an online algorithm to respond to the real-time data patterns. It features a statistical testing to determine whether the current data comes from the same previous distribution, and adjusts the parking locations accordingly. The second tier optimizes charging cost by incentivizing users to aggregate low-energy bikes together. Different from [11] where the incentives are extra payments made by the system, E-sharing balances the incentives with potential expenditures to accomplish an actual cost saving for the service provider. It can be integrated with any prediction engine. We implement a state-of-the-art recurrent neural network [30] for short-term demand forecast and feed the predictions into the algorithms.

The contributions of this paper are threefold.

- As an emerging type of smart and green transportation, we identify new challenges in the dockless E-bike sharing systems being the user dissatisfaction and the usage of public resources. An algorithm is proposed to balance the two, which adaptively combines the advantages from both the online and offline solutions.
- We propose a new mechanism to optimize the high cost due to charging maintenance by incentivizing users.
- We conduct extensive evaluations of the proposed framework on a public dataset [31]. The evaluation indicates over 25% saving of public resources and 47% of maintenance cost compared to the existing algorithms.

To the best of our knowledge, this is the first work that develops an efficient algorithmic framework to offer better planning and optimized maintenance in electric bike sharing system for future smart cities.

The rest of the paper is organized as follows. Section II presents the motivation and system overview. Section III minimizes cost of maintenance and user dissatisfaction. Section IV develops the incentive mechanism. Section V evaluates E-Sharing. Section VI concludes the paper.

II. Preliminary

A. Motivation

Fig. 2 demonstrates a use case of a representative app [3] with snapshots taken in Shanghai. The app looks for the closest locations where the customers can return their E-bikes as shown in Fig. 2(a-c). We discover that parking locations (or referred as stations)[1] based on empirical observation barely catch up with the real-time demands. For example, the stations are far from the Point of Interests (POIs) such as subway stations or park/recreation (Fig. 2(a)); only one station is available across the river near a large residential area (Fig. 2(b)); stations are too close, thereby causing an insufficient spatial coverage (near the university in Fig. 2(c)). Because the service charge is metered, the static placement of stations leads to inconvenience or even complaint if no station is available nearby to return the E-bike. We further examine the energy status at some locations as shown in Fig. 2(d). Though a majority of the E-bikes have sufficient residual energy, the distribution features a tail of low-battery bikes that necessitates energy replenishment at each station. Because of continuous usage and mobility dynamics, it is quite difficult to plan a charging sequence to refill all the batteries at once.

E-Sharing addresses these issues through the two-tier framework. First, it selects candidate parking locations from a large set of available POIs [13]–[15] so the actual parking becomes *dynamic* according to live requests. When there is demand near the POIs, a new station is established according to the batch of trip requests. The location reflects the spatial-temporal consensus among the users. Second, it relocates and aggregates the tail distribution so that operators need not to visit all the locations for charging. The details are discussed in Section III.

B. System Model

Fig. 3 shows an overview of the system components: ❶ the prediction engine forecasts future usage pattern; ❷ the

[1]We use stations and parking locations interchangeably, henceforth. They refer to infrastructureless locations where the E-bikes is returned or picked up.

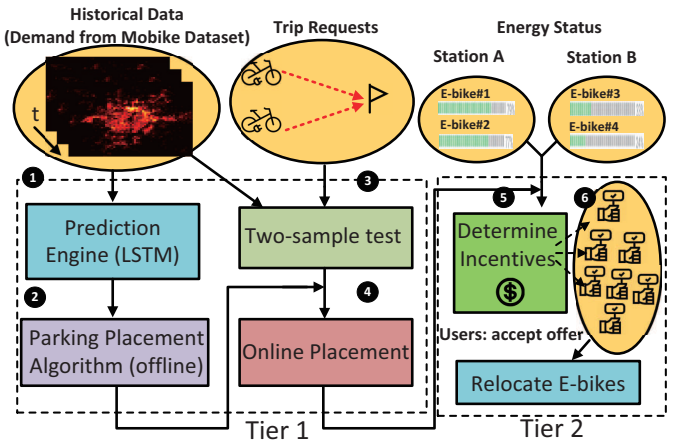


Fig. 3: System architecture of E-sharing.

TABLE I: List of important notations

Notation	Definition
f_i	Cost of public space occupation at location i
c_{ij}	Walking cost from i to j (measured in Euclidean distance)
$\mathcal{P}$	Set of parking locations being established
$g(\cdot)$	Penalty function to adjust parking deployment
k	Number of parking computed by the offline algorithm
$\mathcal{U}$	Set of incoming user requests
$\mathcal{L}$	Set of low-battery E-bikes that requires charging

forecasting results are fed into the parking placement algorithm to generate candidate stations; ❸ the system performs two-sample test periodically to compare the current distribution with the previous one; ❹ according to the test result, the online placement algorithm generates real-time decisions based on the offline solution; ❺ the system determines appropriate incentives to seek cooperation from users; ❻ users accept the offer and help the service provider relocate the low-energy bikes.

To start a new trip, users look for available E-bikes using a mobile app (similar to Fig. 2). The trip request is streamed to the server backend, calculated by E-sharing and assigned appropriate parking locations. We assume that users faithfully enter their destinations and the service provider does not exploit these location data under privacy terms and legislation. For privacy-preserving, additional security features can be introduced such as hashing/anonymizing the user information or obfuscation with location-wise differential privacy [20]. Users follow the service agreement to park at designated area and the service provider imposes an extra charge to regulate misbehavior. The service provider hires a fleet of operators for maintenance. We assume that the reserves of E-bikes are balanced, which satisfy the demand and do not overwhelm the capacity by executing the procedures in [9]–[11]. The operators either replace [3] or charge [5] the batteries (unified as charging henceforth) depending on their technologies. Normally, the operators follow a policy to refill those E-bikes with energy less than a threshold at each location (e.g., below 20%).

III. Tier One: Dynamic Parking Location Placement

The first tier of the framework studies dynamic station placement and formulate it as a *Parking Location Placement* (PLP) problem.

A. Problem Formulation

The Euclidean space of a metropolitan area is divided into grids, which represent the minimum granularity such that users all agree to walk within a grid. We use the centroid to represent each grid and all the arrivals in a grid are denoted by centroid j for simplicity. Consider an undirected graph $G = (V, E)$, where vertices are the centroid and the edges are the links from the centroid to established parking locations. A subset of $\mathcal{P}$ parking locations are selected among the set of all $\mathcal{N}$ grid locations ($\mathcal{P} \subset \mathcal{N}$). The space of $\mathcal{N}$ can be reduced to filter out those less popular locations [15]. The optimal parking locations are determined based on two major factors: *user dissatisfaction* and *space occupation*. On one hand, the business goal is to satisfy the pick-up and drop-off demands. For example, if a user wants to drop off but parking is far from the final destination, she may choose not to buy the service, similar for the pick-up.

Definition 1 (User dissatisfaction). We model user dissatisfaction c_{ij} to be proportional to the walking distance d_{ij} between her destination j and assigned parking i. Given the expected number of arrivals a_j at j, $c_{ij} = a_j \cdot d_{ij}$.

Ideally, users should be allowed to park anywhere but it certainly drives up public space occupation, and subsequently the cost of battery charging. In fact, given their surroundings, parking locations have different impact perceived from the public. E.g., crowded places such as subway station should have a high cost due to limited public space.

Definition 2 (Space occupation). We model the space occupation to be proportional to the number of parking locations. A new parking at location i is established with cost f_i.

Problem Formulation. Obviously, there exists a trade-off between the two: more parking satisfies users at an increasingly higher cost of urban space. Hence, our objective is to minimize the total cost as their sum over a period $\mathcal{T}$,

$$\mathbf{P1}: \quad \min \sum_{t\in\mathcal{T}} \Big(\sum_{i\in\mathcal{P}}\sum_{j\in\mathcal{N}} c_{ij}^t x_{ij}^t + \sum_{i\in\mathcal{P}} f_i^t\Big) \tag{1}$$

Subject to

$$\sum_{i\in\mathcal{P}} x_{ij}^t = 1; j \in \mathcal{N}, t \in \mathcal{T} \tag{2}$$

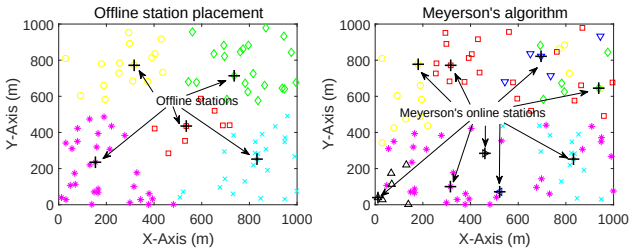
$$x_{ij}^t \le y_i^t; i \in \mathcal{P}, j \in \mathcal{N}, t \in \mathcal{T} \tag{3}$$

$$x_{ij}^t, y_i^t \in \{0,1\}; i \in \mathcal{P}, j \in \mathcal{N}, t \in \mathcal{T} \tag{4}$$

The decision variable x_{ij}^t is 1 if all of the arrivals in j are assigned to park at i. y_i^t is 1 if a parking is established at i; otherwise it is 0. Initially, the candidate parking locations are $\mathcal{N}$. The optimization problem aims to find a subset of $\mathcal{P}$ such that each arrival in t is assigned to a proper parking and the total cost is minimized.

Algorithm 1: 1.61-factor Parking Placement Algorithm (Offline)

Input: Set of grid locations, $\mathcal{N}$.
Output: Set of parking locations $\mathcal{P}$ and $\mathcal{B}_i$, $i \in \mathcal{P}$.
while $t \neq |\mathcal{T}|$ **do**
  while $\mathcal{N} \neq \emptyset$ **do**
    Find $i^* = \arg\min_{i \in \mathcal{N}} \big(\sum_{j \in \mathcal{B}_i} c_{ij}^t + f_i^t - \sum_{j \in \mathcal{B}_i'} (c_{i'j}^t - c_{ij}^t) \big) / |\mathcal{B}_i^t|$.
    Deploy i^*, connect $\forall j \in \mathcal{B}_i^t \bigcup \mathcal{B}_i'$ to i^*, $\mathcal{N} \leftarrow \mathcal{N} - \mathcal{B}_i^t$.
  Output $\mathcal{P}$; restore $\mathcal{N}$ to initial values.



(a) (b)

Fig. 4: Examples of solving PLP (a) offline algorithm (b) online algorithm (56% cost increase compared to the offline algorithm).

B. Offline Placement Algorithm

The PLP problem is analogous to the *Facility Location Problem* [21]–[24], which is unfortunately NP-hard. The original problem is studied in the context of logistics that finds the minimal total cost to open a set of facilities that minimize the sum of transportation cost and facility cost. Due to NP-hardness, no polynomial-time algorithm exists unless P equals NP. A few approximation algorithms are studied [21]–[24]. In [21], a 3.16-approximation algorithm is first proposed and improved by [23] to 1.61, which is very close to the theoretical inapproximation bound 1.46 of this problem [24]. The algorithm reassigns destination requests to new parking locations whenever a lower cost is found during the process. For each parking i, an intermediate set $\mathcal{B}_i$ is introduced to denote those grid locations that have been assigned to it ($\mathcal{B}_i \subseteq \mathcal{N}$). The 1.61-factor algorithm iterates through $\mathcal{N}$ to select i^* with minimum average cost as the new parking locations,

$$i^* = \arg\min_{i \in \mathcal{N}} \big[\sum_{j \in \mathcal{B}_i} c_{ij}^t + f_i^t - \sum_{j \in \mathcal{B}_i'} (c_{i'j}^t - c_{ij}^t) \big] / |\mathcal{B}_i^t|, \quad (5)$$

and then update the requests assigned to the new parking. The iteration stops after all of the grids in $\mathcal{N}$ have been assigned to one of the parking in $\mathcal{P}$. The time complexity is $\mathcal{O}(N^3)$ and the procedure is summarized in Algorithm 1.

C. Online Placement Algorithm

The offline algorithm achieves near-optimality when the successive occurrences are known. While such future occurrences are not known beforehand, a method is to predict the trip requests using machine learning as they should exhibit certain spatial-temporal regularity [17]–[19]. Nevertheless, machine learning relies on the assumption that the test samples are independently drawn from the same distribution at training time because machine learning is good at interpolation rather than extrapolation (when data comes from different distributions). In bike sharing system, it is common to encounter fluctuations that are temporary, or even difficult to predict ahead of time. For instance, events such as concerts or sports games might lead to short-time demand surge at previously unexpected locations. Traffic reroute due to road work or accident may not be reflected by historical data either. Once the current distribution changes due to demand dynamics, suggestions based on the historical prediction become less desirable. A prediction-free solution is to make decisions online as discussed next.

Once a request is received, the system makes irrevocable decisions immediately, without any knowledge of the successive requests. Based on the requesting destination, the system decides where the user should park, i.e., whether the destination would be selected as a new parking with occupation cost f_i or simply assigned to an existing one with dissatisfaction cost c_{ij}. The system does not know or attempt to predict destinations for future requests. An online algorithm is proposed by Meyerson [25]. The algorithm is further extended for clustering online advertisement in [26].

We compare the offline [23] and online [25] algorithms in Fig. 4. A stream of 100 random arrivals in a square field ($1000 \times 1000m^2$) is simulated. For the offline algorithm in Fig. 4(a), 5 parking locations are established and the cost of dissatisfaction, space occupation, and their sum are 16795, 25000 and 41795 respectively. The costs are all measured in the distance unit of m, where the space occupation cost is converted from monetary cost to equal walking distance of users (e.g. 1 $ equal to 1000 m). For the online algorithm in Fig. 4(b), 9 parking locations are established and their relevant cost is 25400, 40000 and 65400 respectively (56% increase of total cost from its offline version). Clearly, though the online algorithm offers a feasible solution, it is worse than the offline one. Such gap is analyzed below.

Theorem 1. No online solution for solving the PLP is $\mathcal{O}(1)$-*competitive* compared to the offline optimal solution.

Proof: Denote the i-th request comes with the coordinates $(2^{-i}, 2^{-i})$, and the cost of opening a new parking at each coordinate is $f = 2$. The walking distance for the i-th user to the origin is $\sqrt{2^{-2i} + 2^{-2i}} = \sqrt{2} \cdot 2^{-i} < f$. Therefore, the optimal solution is to place the only parking at coordinate $(0, 0)$. By doing so, the total cost with n incoming requests is no greater than $2 + \sqrt{2} \sum_{i=1}^{n} (\frac{1}{2})^i = 2 + \sqrt{2} - \sqrt{2} \cdot (\frac{1}{2})^n$. Assume there is an online algorithm which is *b-approximation* to the optimal offline solution, then the number of stations chosen by it must be no greater than $(2+\sqrt{2})b$ (a finite number). Denote the last parking chosen by the *b-approximation* is located at $(2^{-j}, 2^{-j})$, then for all of the incoming requests after the j-th request all have walking distance larger than $\sqrt{2} \cdot 2^{-i-1}$. As the number of requests approaches infinity, the sum of cost is arbitrarily large. Apparently, the algorithm can not be *b-approximation*, since its total cost should be bounded by $(2+$

Algorithm 2: Parking Placement Algorithm with Deviation Penalty (Online)

Input: Streaming trip requests $\mathcal{U}$, parking computed by Algorithm 1 $k = |\mathcal{P}|$, historical data $H(x, y)$, cost coefficient β.
Output: Assigned parking in $\mathcal{P}$, $\forall u \in \mathcal{U}$.
$w^* \leftarrow \min_{p,p' \in \mathcal{P}} ||p - p'||/2$, $a \leftarrow 0$.
$f_i \leftarrow f_i \cdot w^*/k$, $\forall i$; $g(\cdot) \leftarrow TypeIIPenaltyFunction$.
for $u(i, j) \in U$ **do**
  $\mathcal{P} \leftarrow \mathcal{P} \bigcup \{i\}$ with $prob = \min(g(i, j)c_{ij}/f_i, 1)$; $a \leftarrow a + 1$
  if $a \geq \beta k$ **then**
    $a \leftarrow 0$; $f_i \leftarrow 2 \cdot f_i$, $\forall i$
    Peacock's KS-test $\Rightarrow D = sup_{x,y}|H(x, y) - G(x, y)|$
    $D \Rightarrow g(i, j) \in \{TypeI, II, IIIPenaltyFunctions\}$
  $p^* \leftarrow \arg\min_{\forall p \in \mathcal{P}} ||u - p||$, $p^* \leftarrow u$; $G \leftarrow G \bigcup \{i\}$

$\sqrt{2})b$ under the hypothesis, which results a contradiction. Thus no online algorithm for solving PLP can be $\mathcal{O}(1)$-*competitive*. ■

The above analysis suggests that, for PLP, it is not possible for an online algorithm to produce results as good as its offline counterpart, not even close with constant approximation ratios. Yet, such gap is expected and not too pessimistic for an online problem. To find specific problems using the online algorithm, we test the method in [25] and find that: 1) the algorithm may be biased to satisfy current demands and blindly generate too many new parking locations; this directly leads to inefficient use of public space and potentially higher maintenance cost; 2) it may create parking at less desirable locations since immediate decisions have to be made; e.g., when early requests are geographically scattered, new parking is opened regardlessly, despite a better decision is to assign them to the later ones that may be closer to all the users. These findings suggest that making decisions purely online could be far from optimal in some cases. So, both online or offline methods have their pros and cons in practice. To combine their best advantages, we propose a new algorithm next.

D. Our approach: Online algorithm with deviation penalty

The above analysis indicates that usage pattern should follow a certain distribution in most of the time with possible spatial-temporal shift. In [27] the offline and online placement of charging stations serving electric vehicles are independent of each other. We propose a new algorithm that can guide the online allocation with the offline solutions (based on predictions). Two metrics from the offline PLP are re-used: number of parking $k = |\mathcal{P}|$ and their location set $\mathcal{P}$. By referencing to its number, we avoid creating too many parking; using their locations as landmarks ensures established parking does not deviate too much from the historical patterns, but still offers the flexibility to react to the real-time dynamics.

First, the customer initiates a request with a destination. When this request arrives at the server, the walking cost c_{ij} between destination i and the closest parking j is measured (user dissatisfaction). The cost of opening a new parking at i is f_i. Initially, the opening cost is small so the system is encouraged to open new parking. The algorithm keeps checking the number of parking that has been already opened. If it exceeds βk, the opening cost f_i is doubled. $\beta \geq 1$ is a ratio chosen as the input. The cost of opening a new parking grows exponentially until it is prohibitive compared to assigning users to existing parking locations[2]. The decision is made stochastically. With probability $prob = \min(g(i, j)c_{ij}/f_i, 1)$, the destination request i is opened as a new parking or assigned to an existing one j with probability $1-prob$. f_i is the opening cost that increases over time. $g(i, j)$ is a *penalty function* that can take different forms,

$$\textbf{Type I}: \quad g(i,j) = \frac{1}{c_{ij}/L + 1}, c_{ij} \geq 0 \tag{6}$$

$$\textbf{Type II}: \quad g(i,j) = \begin{cases} 1 - c_{ij}/L, & 0 \leq c_{ij} \leq L \\ 0, & c_{ij} > L \end{cases} \tag{7}$$

$$\textbf{Type III}: \quad g(i,j) = e^{-c_{ij}^2/L^2}, \ c_{ij} \geq 0. \tag{8}$$

L is the level of tolerance (measured in distance). It defines how much the system allows the parking established by the online algorithm to deviate from the prediction. Large L permits higher level of tolerance to such deviations. Once the data exhibits a significant divergence, the system could increase L and fit such shift. L is scaled back when the distribution returns to the previous one. To examine whether the new trip requests come from the same distribution as the previous one, we adopt the Peacock's two-dimensional Kolmogorov-Smirnov test (*ks-test*) [28], which is efficient and distribution-free (independent of specific theoretical distributions)[3].

2D KS-Test. The server maintains a stack of historical data, from which the offline PLP has been computed. Denote the cumulative distribution function of the historical data as $H(x, y)$ and the current data as $G(x, y)$. x, y are the latitude and longitude coordinates of the trip destinations. The test computes the largest absolute difference between the two cumulative probability distribution as a measure of misfit,

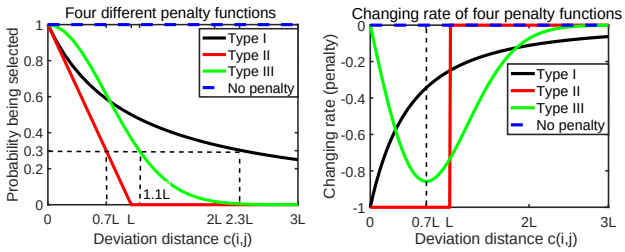
$$D = sup_{x,y}|H(x, y) - G(x, y)| \tag{9}$$

where $sup_{x,y}$ is the supremum of the set of distances (the least upper bound). To find the largest difference for the cumulative distributions in 2D, the Peacock's ks-test enumerates through all four combinations $(x < X, y < Y), (x < X, y > Y), (x > X, y < Y), (x > X, y > Y)$ and selects the largest among the four differences as the final statistic. If D returns 0, the two distributions are considered as similar; or dissimilar, if D returns 1. For n locations, it generates $\mathcal{O}(n^2)$ quadrants so comparing all n requires $\mathcal{O}(n^3)$ time.

Penalty Functions. The penalty functions allow various degrees of location shift stochastically. If destination i falls into

[2]The algorithm also handles the cases when customers pick up all the E-bikes from a station. In this case, the station is removed from $\mathcal{P}$. The algorithm can still establish a station at this location depending on the requests later.

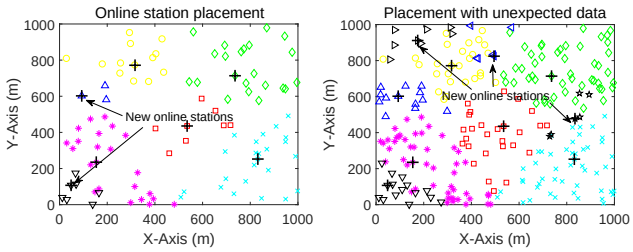
[3]Although high-dimensional ks-test remains a challenge due to the curse of dimensionality [29], the solution in low dimensions like the 2D geographical space still has good usability.



(a) (b)

Fig. 5: Designs of different penalty functions (a) the probability that a new parking is established (b) the first-order derivatives (changing rate).

the grid of established parking j, $c(i,j)=0$ and $g(i,j)=1$ for all three cases. In other words, no penalty is imposed because the destination is very close to the offline solutions. If the destination deviates further from the intended parking j (an increase of $c(i,j)$), $g(i,j)$ declines and it is less likely to open a new parking at i (higher penalty). Fig. 5 illustrates these functions with their first-order derivatives. Type II is designed to plunge much faster than the others; beyond the tolerance level of L, it eliminates the probability to 0 of opening a new parking outside j. Type I applies modest decline and maintains the probability over 0.2 even when the cost $c(i,j)$ goes beyond $3L$. Type III is between the other two. Both I and III maintain some tail probabilities to make it possible to deploy some parking that are far from j. When the temporal shift is significant in terms of large requests, new parking locations would be permitted outside the predicted region of majority. Different penalty functions best adapt to different similarities (derived via 2D ks-test) between the current and historical distributions respectively, which is further discussed in Section V-B and V-C. The online parking placement algorithm is summarized in Algorithm 2, and its time complexity is $\mathcal{O}(n^3)$.



(a) (b)

Fig. 6: Examples of solving PLP (a) proposed algorithm (23% cost reduction from [25]) (b) handling new arrivals from unknown distribution.

Example. Fig. 6 demonstrates an example of running the proposed algorithm. In Fig. 6(a), 7 parking locations (including 2 established online) are opened and the cost of dissatisfaction, space occupation, and their sum are 15542, 35000 and 50542 respectively, a reduction of 23% compared to Fig. 4(b) using [25]. When the new arrivals are drawn from an unknown distribution, Fig. 6(b) shows that 3 more (online) stations are introduced.

Algorithm 3: Incentive Mechanism

Input: Set of parking $\mathcal{P}=\{p\}$, set of low-energy E-bikes $\mathcal{L}$, incoming requests $\mathcal{U}$, charging threshold γ, and incent levels α.
Output: Relocation results for $\mathcal{P}$.
for $p \in \mathcal{P}$ **do**
  $p.status = abundant$.
  if $|\mathcal{L}_p|/|p| < \gamma$ **then**
    $p.status = low$
for $p \in \mathcal{P}$ **do**
  if $p.status = abundant$ **then**
    $\mathcal{S} = \{p' | p'.status = lack, \forall p' \in \mathcal{P}\}$;
    $p^* \leftarrow \arg\min_{\forall p' \in \mathcal{S}} c_{pp'}$.
    for $u(i,j,l) \in \mathcal{U}$ **do**
      if $j = p$ & $l.status = lack$ **then**
        Incentivize $\alpha(q+td)/|\mathcal{L}_p|$ for u to park at p^*.
        if u *accepts* **then**
          $p^* \leftarrow p^* \bigcup \{l\}$, $p \leftarrow p \setminus \{l\}$.

IV. Tier Two: Incentivizing Users for Charging Optimization

The first tier of the framework optimizes the locations of parking regarding user dissatisfaction. E-bike sharing relies on a high utilization and turnover rate to be profitable, which necessitates timely energy replenishment and efficient operation. This section studies the second tier to further optimize the charging operations.

A. Charging Cost

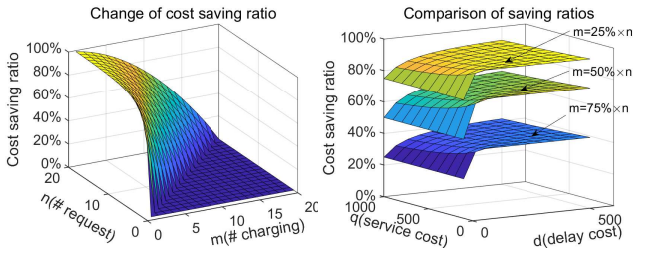
Given the energy status, the operator sojourns at stations for energy replenishment. For each stop, there are some associated timing and monetary cost (e.g. parking tickets at crowded POIs). Denote this by q as the service cost per parking. Serving locations one after another leads to a delay for the later ones in the sequence, denoted by d per parking. This delay can be monetized by converting the demand that the system may have missed because of low energy. Refilling/replacing a battery takes charging cost b so l_i bikes take bl_i at location i. For each parking, the charging cost is modeled as bl_i+q+td if parking i is served in the t-th position in the charging sequence. The *average charging cost* for each E-bike is $b+q/l_i+td/l_i$, which decreases when more E-bikes are serviced at each location. The total cost C for n stations with $l=\sum_{i=1}^{n} l_i$ bikes is,

$$C = nq + \sum_{i=1}^{n} l_i b + \sum_{t=1}^{n} td = nq + lb + \frac{n^2-n}{2}d. \quad (10)$$

The operator's responsibility is to replenish all the E-bikes with low-energy. As seen from Fig. 2(d), the tails of low-energy bikes scatter among different locations and make charging quite inefficient. If the they are aggregated, the total charging cost would be reduced (increase of denominator l_i per parking).

B. Estimate Cost Saving

To see how much cost saving aggregations can offer, denote m as the number of maintenance locations ($m < n$). The



(a) (b)

Fig. 7: Cost saving ratios by applying incentives (a) Change of saving ratio with m and n (b) Change of saving ratio with q and d for different m.

optimized charging cost $C^* = mq + hb + (m^2 - m)d/2$. Therefore, the ratio of cost saving is estimated as,

$$\frac{C - C^*}{C} = 1 - \frac{mq + (m^2 - m)d/2}{nq + (n^2 - n)d/2}. \tag{11}$$

To illustrate the relations between cost metrics and savings, we depict numerical results in Fig. 7. Fig. 7(a) shows that for fixed n, smaller m has quadratically higher cost saving, e.g., $m/n = 0.65$ brings about 50% cost saving. Fig. 7(b) shows the relation between the cost saving and the service/delay cost. For fixed m, once the delay cost increases from a small value, cost saving climbs up sharply at the beginning, and slowly when service cost increases. If the system can allocate a portion of expenditures from the charging cost as incentives to motivate the users for aggregation, is this strategy more cost-effective?

C. Online Incentive Mechanism

We develop an online incentive mechanism to reduce the number of locations m that require charging. Providing incentive is an appealing strategy since: 1) on the customer's side, they earn some rewards, which reduces their payment; 2) on the service provider's side, operators can charge E-bikes together with a lower average cost. In particular, we study how much incentive should be paid in order to have an overall cost saving.

Our objective is to incentivize the users to aggregate low-energy bikes together at some locations k such that a majority of them has energy below the threshold. The energy status of the E-bikes are streamed back to the server [3]. We denote these low-energy bikes by $\mathcal{L}_i$. When a user wants to pick up an E-bike from i with a destination parking of j (computed by Algorithm 2), the system encourages her to ride a low-energy bike $l \in \mathcal{L}_i$ to a neighboring location k by offering an incentive v. The system should ensure the mileage between i and k does not deplete the residual battery. The neighboring location is chosen such that the milage between i to j and i to k is identical. This prevents the users from not considering the offer because of additional milage charge. v is upper-bounded by the cost saved when $\mathcal{L}_i$ are relocated to k so that only location k need service. If the user declines the offer, the system continues to query users arriving later until $\mathcal{L}_i \to \emptyset$ ultimately. The cost saving is upper bounded by,

$$\Delta_i = (b|\mathcal{L}_i| + q + td) - b|\mathcal{L}_i| = q + td. \tag{12}$$

Thus, if $v < (q + td)/|\mathcal{L}_i|$ and $|\mathcal{L}_i|$ users accept the offer, the operator no longer need to visit i, but k instead. The user's behavior of accepting or declining an offer is modeled as,

$$f_u(i, j, k) = \begin{cases} 1, \text{if } c_{kj^*} < c_u \text{ and } v_u^* \le v; \\ 0, otherwise. \end{cases} \tag{13}$$

For each user u, if the extra walking cost c_{kj^*} to the final destination j^*, is less than the accepted maximum walking distance c_u and the incentive v is larger than the accepted minimum reward v_u^*, the offer is accepted; otherwise, it is declined. In principle, to reach an equilibrium, the server should negotiate with the user for several rounds. However, bike-sharing system necessitates real-time decisions and users are not patient to participate in any extended bidding process. For better user experience and fairness, the system issues a uniform offer to all the users $v = \alpha(q + td)/|\mathcal{L}_i|$, $0 < \alpha < 1$. α is a parameter that balances the level of cooperation with the incentives paid by the system. It is set by the operator according to the live demand and supply. E.g, during rush hours, users may be reluctant to walk extra distance (small c_u and high v_u^*). It is difficult to attract users with a small incentive so a slightly larger α can be given. On weekends, users may be more willing to cooperate and earn rewards, thereby a smaller α can be set. The online incentive mechanism is summarized in Algorithm 3.

Remarks: There could be some outliers during operation: 1) no user takes the offer; 2) the new destination is packed with low-energy bikes and undermines usability. For the first case, the system can raise α to attract more users. However, this may cause the payment to exceed the budget (cost saved) and some E-bikes cannot be relocated under the budget. In this case, the algorithm resumes at its best effort to reduce the remaining low-energy E-bikes. Then the operator can skip those locations with only a few ones left. Meanwhile, the algorithm can gradually assign low-energy bikes to these locations, so they have higher chance to be charged during the next service period. For the second case, locations with aggregated low-energy bikes should be scheduled in the upcoming charging service so it would only have minimum and temporary impact on usability.

V. Evaluation

To evaluate the proposed algorithms in realistic settings, the experiments are mainly based on the Mobike dataset [31] which consists of data from the traditional "human-powered" bikes. This dataset is applicable to E-bikes because: 1) there lacks public dataset for E-bikes due to its nascency; 2) POIs are the same so customers have similar destinations around the POIs regardless of the bikes they ride; 3) traditional bikes are allowed to park anywhere and these locations reflect the actual destinations, which facilitate the calculation of user dissatisfaction in E-sharing if stations were established around the actual destinations.

Dataset. The Mobike dataset contains 3.2M bicycle trips from May, 10th to May, 24th in 2017 in Beijing, China. Each trip

TABLE II: Comparison of RMSE of different prediction algorithms (back - # backward time steps in hrs, wz - time window, p - lag order, d - degree of differencing.)

LSTM	back=24	back=12	back=6	back=3	back=1
1-layer	46.3	33.3	42.2	45.4	68.7
2-layer	35.1	29.1	37.8	43.1	68.6
3-layer	36.7	36.6	35.6	42.2	68.7
MA	wz=1	wz=2	wz=3	wz=4	wz=5
	47.9	60.5	66.9	70.7	72.8
ARIMA	p=2	p=4	p=6	p=8	p=10
d=0	42.5	42.4	39.7	35.1	38.3
d=1	42.9	42.2	40.4	37.8	40.3
d=2	48.7	47.8	53.7	45.3	39.9

contains ⟨*order id, user id, bike id, bike type, starting time, starting location, ending location*⟩. The locations are *geohashed*. We re-interpret them into the corresponding latitudes and longitudes, divide all the trips into non-overlapping bins based on the ending locations, i.e., each bin contains trips ended within a $100 \times 100m^2$ grid. 23.9K bins are generated. We establish an energy model based on the data crawled from XQbike App (E-bike), which simulates the energy status of E-bikes. By tracing each *bike id* with the energy status, locations, the model can closely estimate the residual energy of E-bikes.

Experimental Parameters. We aggregate the geohashed grids into a field of $3 \times 3km^2$ and unify the unit of cost into meters. The cost of space occupation is uniformly randomly distributed with mean of 10 (km). The walking cost is measured by Euclidean distance. The cost can be converted into the monetary cost based on space rental and service charge/customer loss in practice. The tolerance L of the penalty function is set to $200m$. The system also has unit delay cost of \$5 and unit energy cost of \$2 when charging the E-bikes. The prediction engine is developed with Tensorflow and tested on Nvidia Tesla P100 GPU and the backend algorithms are developed in Python and MATLAB.

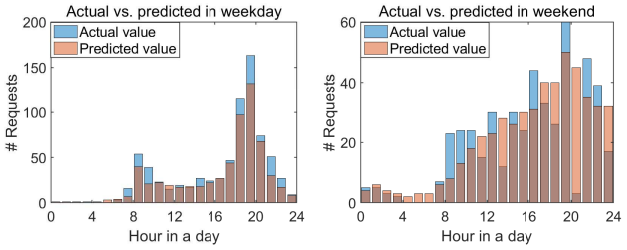
A. Prediction Engine

As an integral part of the system, we first implement and evaluate the predictions. For each grid, given the current time t, it forecasts the future k steps $\{t+1 : t+k\}$ given the data of $\{1 : t\}$. The problem is a sequence learning problem and we utilize a Long Short Term Memory (LSTM) network [30] to forecast the future arrivals. LSTM is the state-of-the-art recurrent neural network that surpasses traditional structures. Each LSTM cell consists of a set of gates to remember and forget relevant information towards minimizing the loss objective. We stack 128 LSTM cells as the hidden layer and extend the depth of the network by increasing the number of layers. Since the data distribution of weekends is different from the weekdays (validated by ks-test next), we process the two-week data as the following. Weekdays are split as 7 days for training and 3 days for testing; weekends are split as 3 days for training and 1 day for testing. Fig. 8 shows the actual number of requests vs. the predictions for a weekday and a weekend among the test data. The Root Mean Square Error (RMSE) is used as the performance measure,

$$RMSE(h^*) = \sqrt{E\left[(h^* - h)^2\right]}. \tag{14}$$

h^* is the predicted number of requests. h is the actual number of requests, and $E\left[(h^* - h)^2\right]$ is the expectation of the square error.

To evaluate the power of different prediction algorithms, we compare LSTM with statistical time-series methods of Moving Average (MA), and Auto-Regressive Integrated Moving Average (ARIMA) [32]. The prediction of trip requests in the next 1 to 6 hours are shown in Table II. LSTM is evaluated with different number of hidden layers and backward time steps (back). MA is evaluated with different window sizes (wz), and ARIMA is evaluated with different lag order (p) and degree of differencing (d). 2-layer LSTM with backward of 12 hours results the minimum RMSE of 29.1, while 1 layer has insufficient representational power and 3 layers slightly overfit. We can see that LSTM provides an average of 30% improvement of RMSE compared to statistical methods, thus used in the rest of the evaluations.



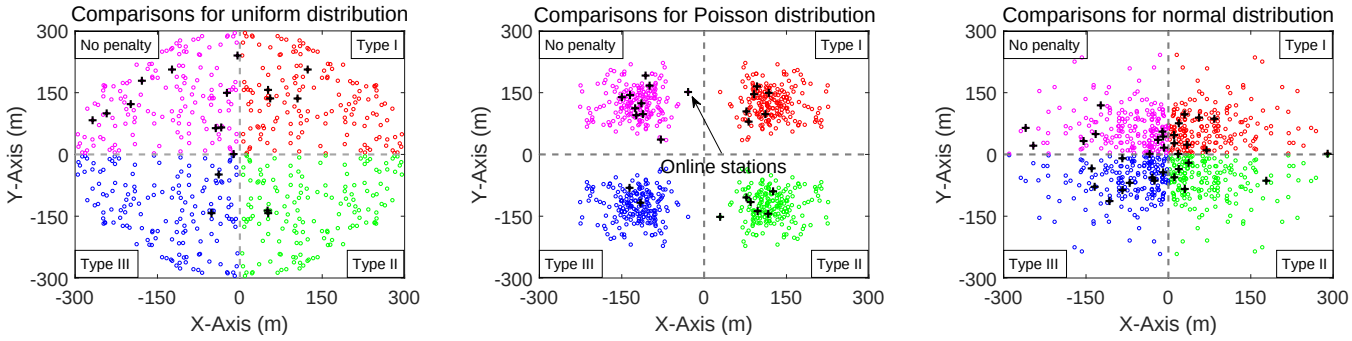
(a) (b)

Fig. 8: Actual requests vs. prediction (a) weekday (b) weekend.

B. Deviation Penalty

Next, we evaluate the design of penalty functions in Section III-C. To understand how the penalty functions respond to the more general cases, we conduct tests on synthetic data using random distributions. Figs. 9(a-c) show three distributions: *uniform*, *poisson* and *normal*, which correspond respectively to an increasing similarity between the actual requests and the predicted requests (the offline derived parking locating at the origin). The results are averaged over 100 tests and summarized in Table III with the bold indicating the minimum cost among the penalty functions. Due to space limit, we visualize all the cases together by dividing each figure into four sectors as *no penalty*, *Type I-Type III* clockwise (origin at the center), and all four sectors have the same distribution about 200 requests. The results are analyzed below.

Uniform: Type I generates less parking compared with *no penalty*. Type II tends to aggregate all the parking around the origin. Type III generates more parking in further distances to the origin compared with Type II. As verified in Table III, it shows that Type II yields the minimum cost of public space and Type I generates the minimum total cost. Since Type I



(a) (b) (c)

Fig. 9: Parking generated under different random distributions for the penalty functions (a) uniform (b) poisson (c) normal.

function increases the penalty smoothly, its range of tolerance is large enough to accept uniformly random arrivals that might occur anywhere in the area, hence the minimum total cost. Based on this observation, it is tempting to conclude that if the distribution of requests fits into the tolerance range of the function, it is prone to have lower cost. This is validated by the following cases.

Poisson: Requests concentrate in the mid-range from the origin which fall into the tolerance range of Type III and Table III shows that it has the minimum total cost. *Normal*: Requests aggregate around the origin, which fits into the tolerance range of Type II. Likewise, Table III shows that Type II generates minimum total cost as well. *No penalty* generates the minimum walking cost as shown in the table, since it has higher probabilities to establish new parking.

From the discussions above, we conclude that the tolerance range should fit to the mean and expand to cover the standard deviation of a random distribution in order to have optimized cost at runtime. Here, we illustrate the designs of three functions and their power to fit different kinds of distributions. It is interesting that we can design the penalty function as high-order polynomials to approximate an incoming distribution in any reasonable shape. We intend to investigate this in future.

TABLE III: Cost of different penalty functions under uniform, normal, and Poisson distributions (in km).

distr. & cost		No Penalty	Type I	Type II	Type III
uniform	Walking	**3.78**	6.23	16.54	12.41
	Public space	23.22	8.22	**3.26**	5.12
	Total	27.0	**14.45**	19.80	17.53
Poisson	Walking	**3.63**	4.35	5.01	6.02
	Public space	23.54	14.31	14.58	**5.35**
	Total	27.17	18.66	19.59	**11.37**
normal	Walking	**3.01**	3.11	6.47	4.23
	Public space	22.81	24.12	**12.11**	20.21
	Total	25.82	27.23	**18.58**	24.44

C. KS-Test of Request Distributions

To evaluate the effectiveness of Peacock's 2D ks-test, we measure the similarity between the distributions of requests in Table IV. The percentage of similarity is computed as $100(1-D)\%$, where D is computed from (9), with closer

TABLE IV: Similarity (%) between distributions of requests.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
Mon	——	94.1	94.4	94.7	97.2	57.8	69.4
Tue	94.1	——	95.3	90.6	94.7	60.6	61.8
Wed	94.4	95.3	——	92.4	92.2	65.6	70.6
Thu	94.7	90.6	92.4	——	96.5	62.9	74.1
Fri	97.2	94.7	92.2	96.5	——	69.4	79.4
Sat	57.8	60.6	65.6	62.9	69.4	——	88.9
Sun	69.4	61.8	70.6	74.1	79.4	88.9	——

to 100% representing perfectly similar. Comparison is made between the same time (hour interval) through different days, and the result is averaged over 24 hrs. We observe that the weekdays and weekends have higher degrees of similarity among themselves (darker background color). This matches with our intuition since demand locations should be aggregated around working places during weekdays, whereas different locations such as recreation, restaurants, parks are more popular during weekends. Interestingly, Mon and Fri are more similar than the rest, and the same for Tue-Thu. One possible explanation is that being the first and last working days, people are in a transition from relaxing to work or the opposite.

The above measurement suggests important thresholds and how they can be co-designed with different penalty functions in Eqs.(6)-(8). We categorize three cases: 1) above 95% (*very similar*); 2) from 80% to 95% (*similar*); 3) below 80% (*less similar*). The proposed online algorithm switches between different types of penalty functions according to periodic measurements. 1) *very similar*: type II is applied to make the new parking close to those from the offline solutions (highest penalty of deviations); 2) *similar*: type III is applied to allow moderate deviations; 3) *less similar*: type I is applied because it tolerates larger deviation with less penalty.

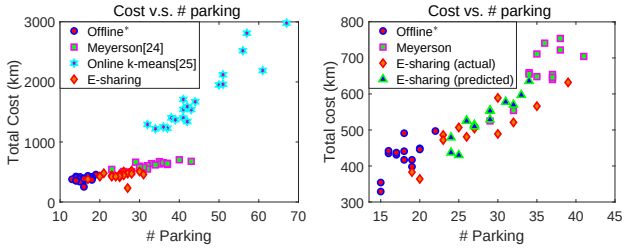
D. Parking Placement Problem

Next, we evaluate the proposed online algorithm (in E-sharing) against the offline algorithm [23] as an upper bound (assuming all the future requests are known in advance; $*$ denotes the solution is near-optimal). Then we compare with

TABLE V: Comparison of # parking and different cost (km).

	# parking	walking	space	total
Offline*	16.0*	242.5*	151.0*	393.5*
Meyerson	32.9	297.4	311.9	609.3
Online k-means	45.2	1326.7	427.6	1754.3
E-sharing (actual)	**25.3**	**220.8**	**239.2**	**460.0**
E-sharing (predicted)	26.0	234.1	253.5	487.6

two other competitive algorithms: Meyerson [25] and online k-means [26]. We demonstrate the relations between the number of parking generated and the total cost in Fig. 10(a) by selecting some random grid locations. Each point represents a selected grid and a PLP is solved independently. It is observed that the Meyerson's algorithm [25] tends to establish more stations than ours but some of them are redundant. The online k-means algorithm establishes even more stations at a high cost of public space. On the other hand, E-sharing is very close to the near-optimal offline solution. This validates its power to utilize predictions and flexibility to adapt to changes. Prediction brings small errors to the actual values, and subsequently impacts on the decision making. Fig. 10(b) shows such impact (online k-means is not plotted due to its poor performance). By comparing the *actual* and *predicted* values, we observe that the errors only yield minor bias in decision making (discussed next). Our algorithm surpasses the Meyerson's algorithm and is within 20% and 25% from the near-optimal offline solutions with and without predictions, respectively.

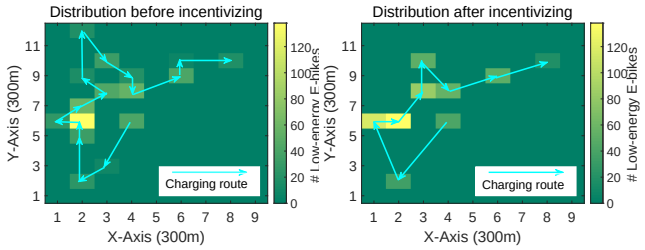


(a) (b)

Fig. 10: Comparison of the total cost defined by Eq. (1) vs. number of parking (a) actual requests (b) predicted requests.

Table V shows overall results and details the cost of walking (user dissatisfaction) and public space occupation. Compared to [25] and [26], it has 25% and 74% less total cost, and 23% and 44% less parking, thus much less impact on the public space. It is worth mentioning that E-sharing achieves even better walking cost than the near-optimal offline results because it is more inclined to satisfy the real-time customer demands. When there are some temporary shift of the demand locations, our algorithm will selectively establish some stations depending on the density of these shift. With LSTM, the prediction error only yields 6% higher cost than the case with perfect knowledge. This indicates that powerful models with better representational capability should be used. Moreover, for each user, we compute the average walking distance (about $180m$ of 2-min walk), which should be acceptable to most of the users.

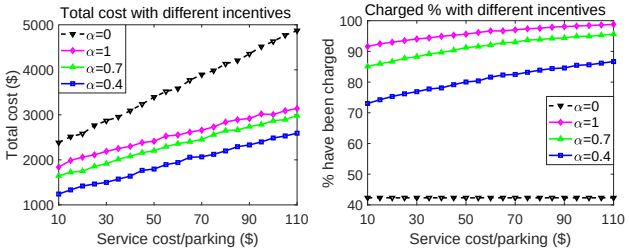
E. Incentivized Parking Relocation



(a) (b)

Fig. 11: Comparison of the distributions of low-energy E-bikes (a) before incentivizing (b) after incentivizing .

We evaluate cost saving of the proposed incentivizing mechanism. Fig. 11 shows examples of energy status before and after incentives are given. We model the energy distribution based on the data captured from the app of [3]. The heatmap shows where low-energy bikes are concentrated. Without incentives, customers tend to use high-energy E-bikes, consume energy and leave at any station. After incentives are given, some users take the rewards to help the system aggregate low-energy bikes towards designated parking. The operator traverses through all the demand sites with the shortest route by solving the Traveling Salesman Problem (TSP) [33] similar to the routing of Mobile Chargers in the Wireless Rechargeable Sensor Networks (WRSN) [34]. A reduction of charging sites and length of moving path is found in Fig. 11(b).



(a) (b)

Fig. 12: Total charging cost and percentage of E-bikes have been charged with different incentives (a) Total cost (b) Percentage have been charged.

TABLE VI: Comparison of charging costs (in $) and charging distance (in km) for different incentivizing levels α.

	$\alpha = 0$	$\alpha = 1$	$\alpha = 0.7$	$\alpha = 0.4$
Service cost	1500	**540**	**540**	**540**
Delay cost	1500	**180**	**180**	**180**
Energy cost	571	548	522	**461**
Incentives	0	1244	1052	**714**
Total cost (sum above)	3571	2512	2294	**1895**
% have been charged	42.3	**96.0**	91.5	80.8
Moving distance (km)	17.1	**14.1**	**14.1**	**14.1**

Recall that the parameter α determines the amount of incentives that the system is willing to give ($\alpha = 0$ represents

no incentive). Fig. 12 shows the total charging cost and the percentage of E-bikes that are charged vs. service cost. The total cost is the sum of cost of service, delay, energy and incentives and the results are summarized in Table VI. Incentives certainly help reduce total cost, especially at places where the service cost is high (e.g. populated downtown area). A higher α motivates more users to cooperate at a cost of higher system expenditures. We see from Fig. 12(a) that a moderate selection of $\alpha = 0.4$ achieves less cost compared to the rest.

Meanwhile, system utility is impacted by the energy levels of the E-bikes. We evaluate this metric by measuring the percentage of E-bikes that have been scheduled under the charging policy. That is, in a fixed amount of working hours, the operator forms a TSP route through all the demand sites and conduct charging in a paralleled manner at each location. The percentage of charged E-bikes is plotted in Fig. 12(b). The value (utility) is proportional to the incentives (improved over 75% with only a small level of incentives $\alpha = 0.4$). Further, the results also suggest a trade-off between potential customer loss due to low battery and cost of charging. A solution is to schedule the operators more frequently during rush hours to the low-energy demand sites.

The average results and a breakdown of the total cost are summarized in Table VI. Incentives can save about 64% service cost, and 88% delay cost. $\alpha = 0.4$ accomplishes the minimum cost in our evaluation by saving 47% total cost. With incentives, the operator also saves 17.5% in distance (potential gas mileage and time waiting for traffic).

VI. Conclusion

In this paper, we optimize the dockless electric bike sharing systems. Our goal is to address the emerging problems of the high maintenance overhead and space occupation with a balance of the user experience. We propose E-sharing, a two-stage optimization framework that can 1) form dynamic decisions of parking locations based on both historical data distributions and real-time demands, 2) incentivize customers to aggregate low-energy bikes for charging optimization, 3) integrate with predictions. The extensive experiments are conducted on large-scale public dataset from Mobike. We implement an LSTM network to predict short-term trip requests. Guided by the predictions, we demonstrate that the proposed algorithm can reduce the user dissatisfaction and space occupation by 25%. By incentivzing the users, 47% of maintenance cost can be saved. These findings are important and timely for optimizing the bike rental businesses for smart and green transportation.

VII. Acknowledgments

This work was supported in part by the U.S. National Science Foundation under grant numbers CNS-1730291 and CCF-1850045.

References

[1] Bike Share in the U.S., 2017, https://nacto.org/bike-share-statistics-2017/

[2] S. Ji, C. Cherry, L. Han and D. Jordan, "Electric bike sharing: simulation of user demand and system availability", *Journal of Cleaner Prod.*, vol. 85, pp. 250-257, 2014.

[3] XQbike, https://www.xqchuxing.com/

[4] Lime electric bike, https://www.li.me

[5] Qee Bike, https://www.qeebike.com

[6] Bird electric scooter, https://www.bird.co

[7] Q. Wang, P. Ping, X. Zhao, G. Chu, J. Sun and C. Chen,"Thermal runaway caused fire and explosion of lithium ion battery", *Journal of Power Sources*, vol. 208, pp. 210-224, 2012.

[8] Norfolk says no to scooters, "https://wtkr.com/2018/11/13/norfolk-has-560-bird-scooters-impounded-company-owes-over-93k-for-them/".

[9] D. Chemla and F. Meunier and R. W. Calvo, "Bike sharing systems: solving the static rebalancing problem", *Discrete Optimization*, vol. 10, no. 2, pp. 120-146, 2013.

[10] Y. Li, Y. Zheng and Q. Yang, "Dynamic bike reposition: a spatio-temporal reinforcement learning approach", *ACM KDD*, 2018.

[11] A. Singla, M. Santoni, G. Bartok, P. Mukerji, M. Meenen and A. Krause, "Incentivizing users for balancing bike sharing systems", *AAAI*, 2015.

[12] P. Zhou, X. Wei, C. Wang, and Y. Yang, "Explore truthful incentives for tasks with heterogenous levels of difficulty in the sharing economy", *IJCAI*, 2019.

[13] J. Liu, Q. Li, M. Qu and W. Chen, "Station site optimization in bike sharing systems", *IEEE ICDM*, 2015.

[14] Z. Liu, Y. Shen and Y. Zhu, "Where will dockless bikes be stacked? - parking hotspots detection in a new city", *ACM KDD*, 2018.

[15] L. Chen, et. al., "Bike sharing station placement leveraging heterogeneous urban open data," *ACM Ubicomp*, 2015.

[16] S. He and K. Shin, "(Re)Configuring bike station network via crowd-sourced information fusion and joint optimization",*ACM Mobihoc*, 2018.

[17] Z. Yang, J. Hu, Y. Shu, P. Cheng, J. Chen and T. Moschibroda, "Mobility modeling and prediction in bike-sharing systems", *ACM Mobisys*, 2016.

[18] Y. Li, Y. Zheng, H. Zhang and L. Chen, "Traffic prediction in a bike-sharing system", *ACM SIGSPATIAL*, 2015.

[19] N. Gast, G. Massonnet, D. Reijsbergen and M. Tribastone, "Probabilistic forecasts of bike-sharing systems for journey planning, *ACM CIKM*, 2015.

[20] Y. Xiao and L. Xiong, "Protecting locations with differential privacy under temporal correlations," *ACM CCS*, 2015.

[21] D. Shmoys, E. Tardos and K. Aardal, "Approximation algorithms for facility location problems," *ACM STOC*, 1997

[22] K. Jain and V. Vazirani, "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation," *JACM* vol. 48, no. 2, pp. 274-296, 2001.

[23] K. Jain, M. Mahdian, E. Markakis, A. Saberi and V. Vazirani, "Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP," *JACM*, vol. 50, no. 6, pp. 795-824, 2003.

[24] S. Guha and S. Khuller, "Greedy strikes back: improved facility location algorithms," *Journal of Algorithms*, vol. 31, no. 1, pp. 228-248, 1999.

[25] A. Meyerson, "Online facility location," In *IEEE FOCS*, 2001.

[26] E. Liberty, R. Sriharsha, and M. Sviridenko, "An algorithm for online k-means clustering," *ALENEX*, Society for Industrial and Applied Mathematics, pp. 81-89, 2016.

[27] P. Zhou, C. Wang, and Y. Yang, "Design and Optimization of Electric Autonomous Vehicles with Renewable Energy Source for Smart Cities", *IEEE INFOCOM*, 2020.

[28] J. Peacock, "Two-dimensional goodness-of-fit testing in astronomy," *Mon. Not. of R. astr. Soc.*, vol. 202, no. 3, pp. 615-627, 1983.

[29] D. W. Scott, "Multivariate density estimation: theory, practice, and visualisation," *John Wiley and Sons Inc*, 2007.

[30] S. Hochreiter, J. Schmidhuber, "Long short-term memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[31] Mobike Big Data Challenge, "https://biendata.com/competition/mobike", 2017.

[32] D. Asteriou, S. Hall, "ARIMA models and the box–Jenkins Methodology", *Applied Econometrics*, pp. 265–286, Palgrave MacMillan, 1997.

[33] S. Skiena,"Traveling Salesman Problem,"*Algorithm Design Manual*, pp. 319-322, Springer, 1997.

[34] P. Zhou, C. Wang, and Y. Yang, "Self-sustainable sensor networks with multi-source energy harvesting and wireless charging", *IEEE INFOCOM*, 2019.