



Article

The Novel Application of Deep Reinforcement to Solve the Rebalancing Problem of Bicycle Sharing Systems with Spatiotemporal Features

Baoran Pan ^{1,†} , Lixin Tian ^{1,2,*,†}  and Yingdong Pei ³

¹ School of Mathematical Sciences, Nanjing Normal University, Nanjing 210023, China; panbaoran1993@163.com

² Research Centre of Energy—Interdependent Behavior and Strategy, Nanjing Normal University, Nanjing 210023, China

³ China Telecom Research Institute, Shanghai 200120, China; peiyd@chinatelecom.cn

* Correspondence: tianlx@ujs.edu.cn; Tel.: +86-1565-165-0332

† These authors contributed equally to this work.

Abstract: Facing the Bicycle Rebalancing Problem (BRP), we established a Rebalancing Incentive System (BRIS). In BRIS, the bicycle operator proposes the method of financial compensation to encourage cyclists to detour some specific stations where the number of bikes is excessive or insufficient and access suitable stations. BRIS mainly includes two objects: the Bike Gym imitating the bicycle environment, and the Spatiotemporal Rebalancing Pricing Algorithm (STRPA) determining the amount of money which is given to the cyclist depending on time. STRPA is a deep reinforcement learning model based on the actor–critic structure, which is the core concept of this paper. In STRPA, the hierarchical principle is introduced to solve the dimensional disaster, and the graph matrix A is introduced to solve the complex node relationship. In addition, the traffic data including the bicycle have strong temporal and spatial characteristics. The gated recurrent unit (GRU), the sub-module of STRPA, can extract the temporal characteristics well, and the graph convolution network (GCN), also a sub-module of STRPA, can extract the spatial characteristic. Finally, our model is superior to the baseline model when verified on the the public bicycle data of Nanjing.

Keywords: deep reinforcement learning; public bicycle; complex network; rebalance



Citation: Pan, B.; Tian, L.; Pei, Y. The Novel Application of Deep Reinforcement to Solve the Rebalancing Problem of Bicycle Sharing Systems with Spatiotemporal Features. *Appl. Sci.* **2023**, *13*, 9872. <https://doi.org/10.3390/app13179872>

Academic Editors: Byung-Gyu Kim and Habib Hamam

Received: 16 May 2023

Revised: 5 August 2023

Accepted: 14 August 2023

Published: 31 August 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Public bicycles (with piles) and shared bicycles (without piles) are low-carbon, green, environmental, and short-range means of transportation and an important part of an urban green transportation system, fundamentally solving *the first kilometer* and *the last kilometer* [1] problems. Since 2016, over 1000 shared bike systems have been deployed in over 60 countries worldwide to promote green transportation and healthy lifestyle [2]. There is some dissatisfactory experience of bike users who need help to rent bikes from stations lacking bikes or return bikes to some stations lacking space. This reduces the usage rate of bicycles, leading to issues such as customer loss, poor management [1,3,4], and so on. The Public Bike Company (PBC) needs to rebalance the demand value of people and the supply value of bike stations to solve the dissatisfaction issue, called the Bike Rebalancing Problem (BRP), and keep afloat. Before solving the BRP, it is necessary to establish that there are three main reasons for this phenomenon. First, the behavior pattern of bike users is unidirectional rather than bidirectional, which is necessary for creating imbalances. The difference between stations with neighborhood functional areas is an inherent factor leading to imbalance. For example, cyclists travel from residential stations to official stations in the morning, and the reverse behavior occurs in the evening. Finally, the behavior of bike users displays a pattern of morning and evening peaks in the time

domain. These intensify the imbalance between supply and demand at bicycle stations. In summary, the mismatch between the demand catered to by the stations and the actual demand from users is urgent to solve. As a result, we cannot look for the method of solving the BRP from the origin. However, the BRP is crucial to ensure people's travel requirements and the quality of bicycle service. In this article, we propose a new method to solve the BRP based on Reinforcement Learning (RL) [5] theory.

According to the division of time for rebalancing tasks, the issue of the BRP is divided into the Static Bike Rebalance Problem (SBRP) and the Dynamic Bike Rebalance Problem (DBRP). The first type describes the fact that in non-peak hours of operation during the day or at night, large vehicles such as trucks transport excess bicycles to stations lacking bikes. To optimize balance time or improve user satisfaction, we find the optimal truck route and complete the redistribution of the number of bicycle station points. This type of method requires the nodes that the truck passes through to not be duplicated, which is generally a variant of TSP [6]. Another approach is to hire bicycle users to balance the number of bicycles at certain stations (or areas). Bicycle users can see the tasks published by the system on a smart app. A red envelope icon represents these tasks. The users need to click the red envelope icon to view the task content and compensation. The stations or bicycles with the red envelope attached are often jokingly referred to by users as red envelope stations or red envelope bikes. Current research on DBRP is much more scarce than that on SBRP. However, the DBRP has apparent advantages. The unit cost of truck-balanced bicycles is much higher than the unit cost of user-balanced bikes. When we adopted a user approach to reduce rebalancing costs, we established the Bike Rebalancing Incentive System (BRIS) to address BRP.

The Bike Rebalancing Incentive System (BRIS) has two adversarial targets: bicycle operators and bicycle users. Bicycle operators release balance tasks and present them on red envelope station or red envelope bikes. After completing the task, bike users receive compensation from the red envelope station or red envelope bikes, while the bike sharing system (BSS) performs the rebalancing work. The red envelope setting requires us to consider the impact on the cycling environment. Firstly, the red envelope system can cause bicycles to accumulate in or evacuate a certain area quickly. Fricker and Gast found that after users participate in rebalancing work, the performance of BSS significantly improves. Secondly, the red envelope system is essentially a commodity [7]. Its price fluctuates and is not fixed. When considering whether to accept the rebalancing task, users determine a personal cost [8]. When the red envelope price is higher than the personal cost, the user accepts the task; otherwise, the user refuses the task and the supply–demand conflict in the bicycle system still exists. The price is related to the completion of the task. Afterward, the price of the red envelope task can be associated with the degree of urgency related to the state of the bicycle environment. When releasing tasks, it is necessary to consider the current situation and solve BRP from a long-term perspective [9]. Finally, the bike stations are closely connected, possibly leading to cutthroat competition between the users of red envelope stations. This article develops a price strategy for red envelope station users as an Incentive Strategy (IS). We build incentive strategies based on the idea of the Markov decision process (MDP) theory [5] and focus on the following three issues.

Question 1. *How to evaluate an incentive strategy?*

Ghosh et al. proposed an online scenario generation method that formulates rebalancing strategies based on virtual worse demand scenarios [10]. In this article, we establish a bicycle environment simulation scenario—Bike Gym. It contains the environmental information and operational rules of the entire bicycle network. Each incentive strategy results in a scenario. We can obtain user satisfaction with bicycle journeys and user acceptance of rebalancing tasks under each incentive strategy. We evaluate the effectiveness of an incentive strategy based on user feedback. We establish Bike Gym because, in the real world, once incentive strategies are formulated incorrectly, irreversible losses can occur. In

addition, each incentive strategy and its feedback are valuable experiences to learn from. We develop the optimal incentive strategy through extensive historical experience.

Question 2. *How to improve incentive strategies?*

Firstly, the flow of self-operated station points, the existing inventory and maximum storage capacity of the station, as well as nearby environmental and historical information, all directly affect the formulation of strategies. We use the multidimensional vector to represent the feature information related to the rebalancing task. The time information of stations is extracted by GRU [11], which is the variant of the long short-term memory (LSTM) [12]. In addition, the interaction between nodes, represented by the neighbor matrix A , also affects the formulation of strategies. The spatial information of stations is extracted by the graph convolution network (GCN) [13,14] with matrix A . In addition, GCN is applied to the prediction problem of bike network [15–17]. Finally, when formulating a strategy, we should consider its long-term benefits for the future rather than just current feedback. The advantage of early planning is that it can handle the {bike tide phenomenon of bicycles more calmly.

Question 3. *How to deal with dimensional disasters that may arise in incentive strategies?*

The high-dimensional node characteristic and a large number of nodes simultaneously result in very large state space and numerous incentive strategies in MDP theory [5]. Even with the help of computers, it is difficult for us to obtain the optimal solution. Seijen et al. proposed and verified that deep Q networks (DQN) [18] can decompose the entire reward function into multiple components [19]. Based on the principle of hierarchical thinking, we use a combination of low-dimension functions to approximate our model's action and value functions. It can prevent the occurrence of dimensional disasters.

In BRIS, the most crucial task in solving the rebalancing bicycle problems is formulating the optimal incentive strategy, which is also one of the core works in this article. Therefore, we propose a hierarchical reinforcement learning structure called spatiotemporal rebalancing pricing algorithm (STRPA) that can extract spatiotemporal features, as shown in Figure 1. This model can better extract spatiotemporal features related to rebalancing and optimize incentives. At last, the related work is shown in Section 2. The Bike Gym, generating interactive experiences, is explained in Section 3 and the STRPA is explained in Section 4. Section 6 is the representation of relevant results. Section 7 is the conclusion.

The main contributions of this article are as follows:

1. Two sub-modules, Bike Gym and STRPA, are established in BRIS.
2. From the perspective of allocating user demands based on the spatiotemporal characteristics of bicycles, the rebalancing bicycle problem is solved with the MDP idea.
3. The hierarchical principle is applied to solve the problem of dimensional disasters caused by complex state spaces.
4. The matrix representing the complex node relationships is introduced. The use scope of the algorithm can be broadened and applied to non-European structured networks such as complex networks. BRIS is applied as an experiment to Nanjing public bike data set in this article.

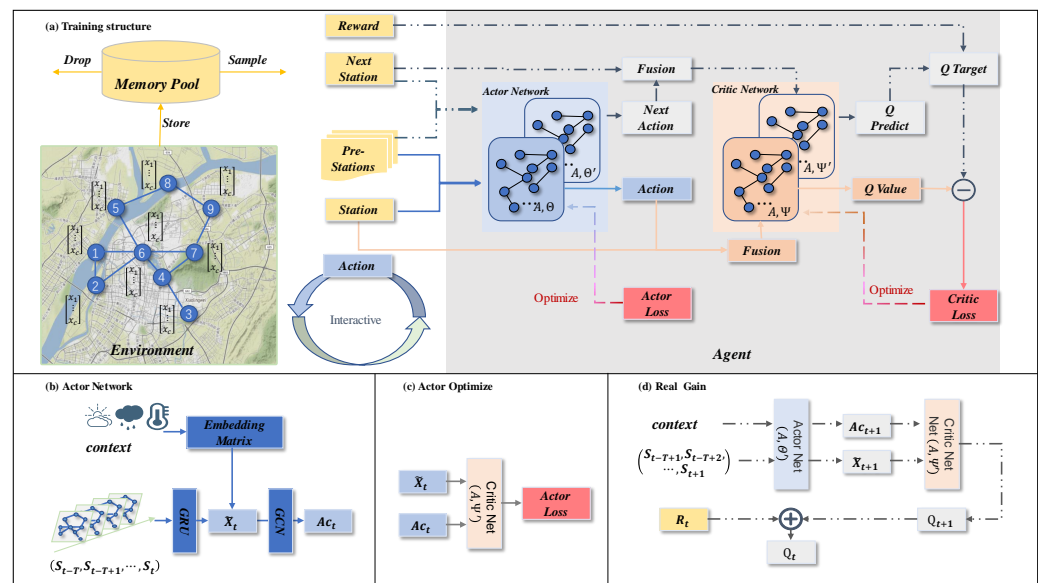


Figure 1. STRPA model. In (a), there are three parts: Environment, Agent, and Memory Pool. The blue and green thick arrows indicate the continuous interaction between the environment and agent, and the interaction results are stored in the memory pool. The agent includes two structures: the actor module and the critic model. The actor model formulates the pricing strategy of the stations, and the critic model evaluates the quality of the strategy. In (b), the process of extracting spatiotemporal data features and formulating action strategies are shown in detail. The process of optimizing the actor model by actor loss is shown in (c) and the process of obtaining the long-term benefits of the Markov chain as the target is shown in (d).

2. Related Work

2.1. Bike Rebalancing Work

Static rebalancing work usually occurs during non-peak hours of day or night, when the supply and demand of bicycles remain unchanged or have little change. The method of picking up or dropping off a certain number of bikes through trucks is often described as a vehicle routing problem (VRP) with delivery and extraction, which was first proposed by Dantzig and Ramser [20]. In the early stages, it was required that nodes in the vehicle path could only be accessed once. The goal of solving the single-vehicle static reposition problem proposed by Szeto et al. is to minimize the weighted sum of unmet customer demand and operational time on the vehicle route [21]. They proposed an enhanced version of chemical reaction optimization (CRO) to handle vehicle routes and obtain better route solutions. Szeto and Shui investigated a bike repositioning problem that determines the routes of the repositioning vehicles and the loading and unloading quantities at each bike station to minimize first the positive deviation from the tolerance of total demand dissatisfaction and then service time [22]. Recently, many researchers have proposed VRP variants, allowing multiple-time access to nodes in the path. Seijen et al. dealt with a particular case of SBRP in which only a single vehicle is available, and the objective was to find a least-cost route that meets the demand of all stations and does not violate the minimum (zero) and maximum (vehicle capacity) load limits along the tour [19]. Wang and Szeto proposed combining BRP with environmental issues. They aimed to minimize the total CO₂ emissions of all repositioning vehicles and discussed the characteristics of the problem and the factors that affect the CO₂ emissions [23]. Afterward, more complex rebalancing environments were explored. Liu et al. studied a bike repositioning problem with multiple depots, multiple visits, and multiple heterogeneous vehicles [24]. Another type of rebalancing work occurs during the operation of Bike Sharing Systems (BSS) during the day, commonly referred to as dynamic rebalancing issues. Caggiani et al. added a predictive system to the rebalancing problem based on the vehicle path, dynamically redistributing bicycles at fixed daytime gaps to achieve high user satisfaction [25]. The above methods are passive based on

existing scenarios to solve the rebalancing problem. Fortunately, Ghosh et al. developed a scenario generation approach based on an iterative two-player game to compute a strategy of repositioning by assuming that the environment can generate a worse demand scenario (out of the feasible demand scenarios) against the current repositioning solution [10]. This online and robust repositioning approach could also minimize the loss in customer demand.

Some researchers noticed bike users in the system and introduced them to rebalancing the bike system. Daniel Chemla discussed for the first time the user private costs, user choices, and pricing methods for incentive strategies. This literature assumes that users would complete rebalancing tasks based on the principle of greed when the rewards given by the system exceed their private costs [8]. Fricker and Gast proposed a stochastic model of a homogeneous bike-sharing system and studied the effect of the randomness of user choices on the number of problematic stations [7], i.e., stations that, at a given time, have no bikes available or no available spots for bikes to be returned to. Singla et al. presented a crowdsourcing mechanism that incentivizes the users in the bike repositioning process by providing them with alternate choices to pick up or return bikes in exchange for monetary incentives. They designed the complete architecture of the incentive system which employs optimal pricing policies using the approach of regret minimization in online learning. The architecture was the first dynamic incentive system for bike redistribution ever deployed in a real-world bike sharing system [4]. Cheng et al. designed a dynamic bidding-model-based incentive mechanism (BIM) to progressively determine the incentive price, based on which users are assigned the rebalancing task [26]. Some studies have also focused on the level of the number of stations in BRP. Li and Liu investigated a static bike rebalancing problem with optimal user incentives. It is worth noting that they proposed a bi-level variable neighborhood search algorithm to solve large problems [27].

2.2. Reinforcement Learning

The Markov decision process theory (MDP) [5], which is the core idea to solve the bike rebalancing work in this paper, contains two interactive objects: agent and environment. When the agent formulates a strategy based on the current state of the environment, feedback is given immediately after the environment executes the strategy. Afterward, the agent calculates the long-term benefits of each strategy to evaluate its strengths and weaknesses and optimize it in the right direction. The detailed theory can be found in Section 4.1. When MDP is applied to bicycle networks, this article views bicycle operators as agents and considers bicycle networks, including all bicycles, stations, and users, as environments. Bike operators develop red envelope stations or red envelope bikes based on the current environment, where users obtain relevant information and perform rebalancing tasks based on greed, changing the state of the environment. The driving route directly manifests the environment, and user satisfaction is feedback from the environment. Some researchers have noticed that the MDP idea has a more subjective initiative in BRP and solves the imbalance between supply and demand in bicycle networks by influencing them. However, when applying MDP, we often face the problem of dimensional disasters due to the large quantity level and complex environment of bicycle environments. Seijen et al. proposed the hybrid reward architecture (HRA) based on deep Q-network (DQN) [18]. HRA took a decomposed reward function and learned a separate value function for each component reward function. Because each component typically only depended on a subset of all features, the corresponding value function could be approximated more easily by a low-dimensional representation, enabling more effective learning [19]. Pan et al. developed a novel deep reinforcement learning algorithm called Hierarchical Reinforcement Pricing (HRP), which builds upon the Deep Deterministic Policy Gradient algorithm (DDPG) [28]. HRP captures spatial and temporal dependencies using a divide-and-conquer structure with an embedded localized module to incentivize users to rebalance bike systems [9]. According to the bike riding route, bike stations can be divided into origin and destination stations. Duan and Wu noticed this and extended the structure of HRP. The new method integrated the origin and destination incentives adaptively by adjusting the detour level at

the origin and/or destination by avoiding bike underflow and overflow [29]. The process of solving the problem of bicycle rebalancing is called the Bicycle Rebalancing Incentive System (BRIS). BRIS focuses on two parts: the environment generates an interactive experience, and the agent develops incentive strategies.

3. Bike Gym

Here, we design a general environment of a bicycle network—Bike Gym. Why do we define the bicycle environment with a Bike Gym? The gym package is imported when we solve RL with the programming tool Python. The gym owns some environment cases and creates some new environments. Therefore, we name the part of the imitating environment the Bike Gym. Bike Gym is a program used in a computer to create a simulation environment for a system. Many relevant definitions and theoretical foundations need to be used in this process. Therefore, we use the functions in the program as partitions and introduce their corresponding knowledge points. The running framework of Bike Gym is demonstrated by Algorithm 1. It mainly has initialization (initialize function), execution (step function), resetting (reset function), storage (memory pool), and other abilities. The initialization is corresponding to the procedure of the initialize function. It defines the structure of variables; for example, bike trajectory (seen in Definition 1). It defines the bike context environment, containing locations of bikes, weather data, week data, and so on. It defines the meaning of variables; for example, time distance (seen in Definition 2). It defines the value of variables; for example, the delta in Equation (1). It is worth mentioning that we imported a large amount of real bicycle trajectory data during this process, which allowed the filtering and constructing of new bicycle trajectory data based on parameters. Execution corresponds to the procedure of the step function. The current state S_t becomes the next state S_{t+1} after action A_{c_t} , and then the system can provide feedback R_t . That is the core content of the gym, including the operation rules of the bicycle network, described in Section 3.2 (seen in Algorithm 1). Resetting corresponds to the procedure of reset function. The chain of $\dots, S_t, A_{c_t}, S_{t+1}, R_t, \dots$ (seen in Figure 2) is not infinite in the real world, and we need different values of the initial state. Therefore, the current state can be interrupted, and the initial state and the context information can be reset; for example, the data pertaining to a given week. Storage is corresponding to the procedure of the memory pool. Experience bar $S_{t-K}, S_{t-K+1}, \dots, S_t, A_{c_t}, S_{t+1}, R_t$ is built and stored in the memory pool. The memory pool has a fixed capacity. The old experience bar is deprecated, with new experience added when the pool is full. The rule also ensures that the model is constantly learning new experiences. Refer to Section 3.4 for detailed information.

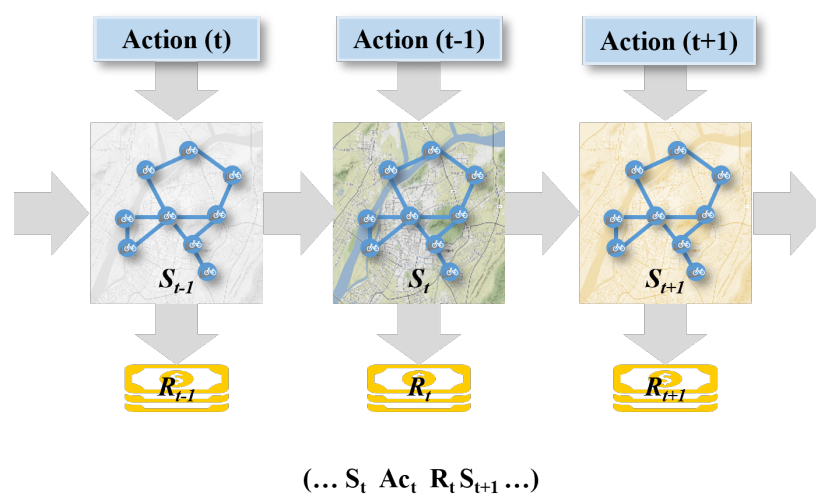


Figure 2. MDP chain.

Algorithm 1 Bike Gym.

-
- 1: Initialize bike temporal network \mathcal{G} and set context feature $cont$
 - 2: Initialize $startflow$ and $endflow$ with 0 which represents user choice
 - 3: Hyperparameter Node number N , Time period \mathcal{T} , penalty Rd
- Input:** current state s , action a
- Output:** next state s' , reward r
- 4: Extract bicycle usage status from bicycle trajectory, obtain the time behavior sequence $\mathcal{L} = [(T_t, V_i, \mathbb{V})], (t = 1, 2, \dots, \mathcal{T}, i = 1, 2, \dots, \mathcal{N})$, where \mathbb{V} elements mean a user borrowed or returned a bike on station V_i at time T . M represents the behavior of borrowing or returning.
 - 5: **for** time $i = 1, 2, \dots, \mathcal{N}$ **do**
 - 6: Match a user and user u location to node V ;
 - 7: According to Equation (2), obtain user *generalized cost*;
 - 8: Recommend suitable sites based on user location and car usage patterns for user u ;
 - 9: Calculate the user's *utility cost* and obtain the user's *choice* according to Equation (3) or Equation (4);
 - 10: Obtain current *rewards* based on user satisfaction.
-

3.1. Gym Initialization

Popular bikes are divided into public bikes (with piles) and shared bikes (without piles). The public bike takes the station as the node in the network, and the shared bike often takes the artificially divided grid as the node in the complex network theory [30]. The matrix of the complex network can uniformly express the relationship of nodes. To make Bike Gym imitate the environment that is usually run, it must be ensured that the data are actual data and that the environment operation rules conform to reality. Next, the relevant information used in Bike Gyms is defined.

Definition 1 (Bike trajectory). *Take the bicycle station as the node in the network, and all nodes in the network are represented by the set $v = \{v_1, \dots, v_n\}$. Each bicycle track can be represented by a four-tuple $e = (v_i, v_j, t_i, t_j)$, where v_i indicates the source station, v_j means the destination station, t_i indicates the start time, t_j indicates the termination time.*

Temporal network \mathcal{G} [31] contains a node set V and an edge set E . The nodes represent the bike stations, and the edges represent the bike trajectories. Significantly, the edge contains time information.

Definition 2 (Time distance). *The average riding time between nodes is the time distance of nodes. $T_{dist}(v_i, v_j)$ represent the time distance between node v_i and node v_j .*

The smart system recommends nearby nodes to users based on the time distance between nodes. A new bike path is generated in the bike environment if a user receives the recommendation. The geographical distance between nodes is not the linear distance on the map but the distance of the actual cycling path. Because the cycling speed is stable, the cycling distance and time are directly proportional. We use cycling time to represent the distance between nodes. Each node has a digital label, name, longitude and latitude coordinates, as well as other information, so the Baidu Maps app can also be used to select an appropriate route to calculate the time distance between nodes. The applied matrix T_{dist} represents the time distance between nodes. In a complex network, if the node v_i and node v_j are not related, the relationship is indicated by 0; otherwise, it is indicated by a nonzero value. Now, the relationship between nodes is determined based on their time distance. If node v_i can affect node v_j , there is a connection between the two nodes, and node v_i is called the neighbor of node v_j . The formula is as follows.

Definition 3 (Neighbors of nodes). If the relationship between node v_i and node v_j is less than δ , node v_i and v_j have a neighbor relationship; otherwise, they do not.

$$\mathbb{A}(v_i, v_j) = \begin{cases} 1, & a_{i,j} \leq \delta \\ 0, & \text{else.} \end{cases} \quad (1)$$

For public bikes with piles, the station's capacity is the maximum value of vehicle storage. For shared bikes without piles, the actual geographical area of the storage location is the maximum value of vehicle storage. C_i indicates that the station v_i can store the maximum number of bicycles. $N_i(t)$ indicates that station v_i owns the number of bicycles at times t . Therefore, there are $0 \leq N_i(t) \leq C_i$. $CT(t)$ represents the external environment characteristics of the entire bicycle network at times t .

Our Bike Gym stores much historical data on bike trajectory and basic information about the bicycle environment. Therefore, the simulation system can extract historical bike paths and generate new ones according to the current environment.

3.2. Operation Rules

For users, every trajectory contains an origin node and a destination called an OD pair. The method of this paper adjusts OD pairs by giving users a bonus. As shown in Figure 3, node v is the old origin node of the user, and there are two candidate nodes, w_1 and w_2 . Users think conscientiously that node w_2 is selected in their best interests. Solving the user demand or assigning the bike number of nodes is also achievable through detouring some special nodes. In this process, Bike Gym receives the incentive strategy and determines the price for every node, which is shown to users. Whether the user accepts the task is related to the user's private cost. Since private costs are opaque, we made the following assumptions.

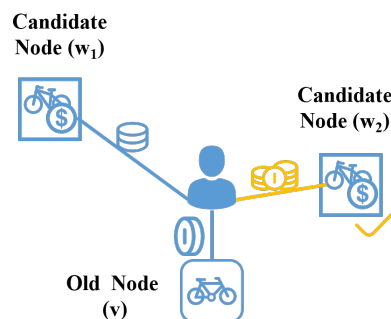


Figure 3. Detour Rule.

Assumption 1. We extend the user cost of the convex structure proposed by [4] and introduce the variable of user age. The cost of walking distance of x from user u_i^k can be expressed as follows:

$$C_{k,i}(x) = \begin{cases} 0, & 0 < x < \delta_1 \\ \alpha x^2, & \delta_1 < x \leq \delta_2 \\ +\infty, & \text{else,} \end{cases} \quad (2)$$

where α is the coefficient. $C_{k,i}(x)$ represent user's generalized cost. If the distance between a user and a red envelope bike is lower than b_1 , the user's private cost is thought to be zero. If the distance is very large and exceeds the user's endurance distance b_2 , users do not choose to tour a node.

Definition 4 (Action). $Ac(t) = \{p_i(t)|v_i \in V\}$ represents the action which Bike Gym implements at time t , where $p_i > 0$ represents the users borrowing bikes at station v_i and obtaining the task bonus, $p_i < 0$ represents the users returning bikes at station v_i and obtaining the task bonus.

u_i^k represents the k_{th} user neighbor of station v_i who can look up the incentive information on the smart application. Based on greedy psychology, users choose the stations they benefit most from, and the new stations can replace the old stations. The effective cost of the users is equivalent to the difference between the money given by the operator and the task cost, i.e., $|p_j| - C_{k,j}(x)$. The scheme which users select on the new station during the source or the destination detour to obtain maximized benefits can be expressed as

$$v_{start} = \arg \max_{v_i} (p_j^+ - C_{k,j}(x)), \text{ where } v_j \in N(v_i), \quad (3)$$

$$v_{end} = \arg \max_{v_i} (p_j^- - C_{k,j}(x)), \text{ where } v_j \in N(v_i), \quad (4)$$

where $\arg \max$ is the inverse of the maximum function. $p_j^+ - C_{k,j}(x)$ and $p_j^- - C_{k,j}(x)$ represent users utility cost. Equations (3) and (4) are the rules of users choosing candidate nodes.

Assumption 2. Because our goal is to reduce the cost of bicycle rebalancing by encouraging users, there is a ceiling of B for daily balancing costs. B is lower than the cost of rebalancing through a truck. Therefore, the balance cost of nodes in the whole network in one day is constrained by B . The formula is as follows:

$$\sum_{t=1}^T \sum_{i=1}^N |P_i(t)| \leq B. \quad (5)$$

When Bike Gym accepts the incentive strategy, each user can make the best choice based on greed. However, not every user can execute this optimal selection. Users are constrained by the total bike rebalancing cost. When the rebalancing cost from Bike Gyms is insufficient, rewards cannot be promised to users after performing tasks. At this time, the agreement between users and agents is broken. Bike Gym also adds contextual environment information such as weather and temperature. If it rains suddenly when the user is performing a rebalancing task, the user fails to complete it. In addition, users may also be unable to complete a task when there is no bike available at a station or no parking space at a return station. Therefore, the bike environment needs to provide different feedback on completing the task.

$$\mathcal{R}_i^t(u_k) = \begin{cases} 0, & v_{new} = v_{old} \\ 1, & v_{new} \neq v_{old} \text{ and borrow (or return) successful.} \\ -M, & \text{task unsuccessful} \end{cases} \quad (6)$$

$\mathcal{R}_i^t(u_k)$ represents user u_k providing feedback on node v_i . The basic standard is that users can borrow or return a bike successfully. Users participate in the rebalancing work by detouring the nodes. The feedback is represented as 1, and if users borrow or return bikes unsuccessfully, the feedback is represented as a negative value.

3.3. System Dynamic

$J_i(t)$ indicates the bike's number of users picking up bikes on node v_i during time t . $R_i(t)$ indicates the bike number of users picking back bikes on node v_i during time t . $SF_{i,j}(t)$ means requests station v_j changed to station v_i . $\tilde{J}_i(t)$ means new request station v_i after price incentive. $EF_{i,j}(t)$ means return station v_j changed to station v_i . $\tilde{R}_i(t)$ means new return station v_i after price incentive. Considering the bicycle infrastructure, actual road conditions, and user psychology, Bike Gym simulates the bicycle itinerary of stations in the city. The bicycle request behavior, the return behavior, and the number of bicycles stored at the site are evolving, as shown in the following formula:

$$\tilde{J}_i(t) = J_i(t) - \sum_j SF_{i,j}(t) + \sum_j SF_{j,i}(t), \quad (7)$$

$$\tilde{R}_i(t) = R_i(t) - \sum_j EF_{i,j}(t) + \sum_j EF_{j,i}(t), \quad (8)$$

$$N_i(t+1) = N_i(t) + \tilde{R}_i(t) - \tilde{J}_i(t). \quad (9)$$

$\sum_j SF_{i,j}(t)$ represent the user demands of node i to move to other nodes during the t time period. $\sum_j SF_{j,i}(t)$ represent the demands of other nodes besides node v_i to move to node v_i . Therefore, the left of Equation (7) represents the new demands of the user during the t period. The meaning of Equation (8) is the same as that of Equation (7). What is more, the time sequence $\mathcal{L} = [(T_i, V_i, \mathbb{V})]$ represents all behaviors of node v_i during the t time period; for example, $[(t^1, v_0, 1), (t^2, v_0, 1), (t^3, v_0, -1), \dots]$. The sequence represents one bike borrowed at t^1 , one borrowed at t^2 , and one returned at t^3 , and so on. The total number of bikes borrowed on node v_i during t period is $\tilde{J}_i(t)$. Equation (9) represents the number change of node v_i .

Definition 5 (State). We use the discretization method to divide the time evenly into T parts, and all periods are represented by the set $\mathcal{T} = \{1, 2, 3, \dots, T\}$. The state S_t of the environment at the time t is characterized by tuple $S_t = (J(t), R(t), N(t), C(t), CT(t))$, where $C(t)$ represents the maximum capacity of the station v_i and is a constant. $CT(t)$ represents the weather and other context environments. The context environment can be considered invariant when the overall system is suitable for riding bikes.

Definition 6 (Reward). Based on current state S_t and action Ac_t , Bike Gym returns reward $R_t = \{r_i(t) | v_i \in V\}$.

Based on the observed status S_t , a set of actions Ac_t is passed to the step function of Bike Gym. According to the operation law of the environment, the step function outputs reward R_t and updates the system state to S_{t+1} . It is worth noting that the next state here is obtained not by calculating the transition probability but by system simulation. We put the interactive experience into the memory pool, including the historical state, action, reward, next state, and stage. The phase here is 0, which means there is a next state. If it is 1, the next is the termination phase.

3.4. Memory Pool Initialization

Our goal is to propose a new pricing algorithm in the bicycle rebalancing incentive system (BRIS) under a limited budget and in consideration of long-term interests to promote the rebalancing of bicycle supply and demand. Therefore, the optimization objectives are as follows:

$$object = \max \sum_{t=0}^T \sum_{i=1}^N \sum_{k=1}^{K_{it}} \mathcal{R}_i^t(u_k). \quad (10)$$

In the face of the number of city-level stations and detailed time division, although the critical network eventually guides us to obtain the improved incentive strategy, the tortuous process experiences a lot of troubleshooting tests and consumes a lot of memory. The efficiency is improved if a correct direction is provided initially. Therefore, we conduct relevant operations.

- step 1 The range of incentive strategies is limited, with a maximum value p_{max} and minimum p_{min} . For the convenience of calculation, the incentive action of the station is discontinuous, with $p \in \bar{Ac} = \{p_{min}, p + \delta_p, \dots, p_{max} - \delta_p, p_{max}\}$.
- step 2 Bike trajectories are not affected by the time interval size. The range of time intervals is expanded in the initial exploration process of action. The stations are classified into four categories according to the frequency of vehicle borrowing and returning at stations and the differences in traffic flow by bike operator. We select stations

of key protection types and their neighbors as a subset of sites for testing (seen in Section 5.1).

- step 3 On the screen, we randomly test a set of finite actions. Store valid action sets according to feedback from Bike Gym.
- step 4 The neighborhood search algorithm [27] is used to expand the time domain, vertex domain, and action domain, and a larger and finer screen is used to select good feedback rewards at the intersection and save the filtering results.
- step 5 For all stations, the action value at the screen node is produced by the previous process and remains unchanged. In contrast, the action value at other stations is generated randomly. The action of all stations is transferred to the step function. The generated interactive data are stored in the memory pool.

4. Spatiotemporal Rebalancing Pricing Algorithm

In this chapter, we mainly introduce the ways in which the agent in MDP determines incentive strategies. The basic principles of MDP are presented in Section 4.1. We propose a deep reinforcement learning algorithm—a spatiotemporal rebalancing pricing algorithm (STRPA) based on an actor–critic framework [5,27,32]. STRPA corresponds to the role of the agent in MDP. STRPA includes two sub-modules: actor module and critic module. The actor module extracts the spatiotemporal characteristics of the bicycle network to determine the current incentive strategy (also called the action in this article). In contrast, the critic module evaluates the long-term benefits of the current incentive strategy. Finding the best incentive strategy is the goal of our model. The optimization methods for two sub-modules are described in Section 4.4. After extensive practical experience and model optimization, the incentive strategy determined by the improved model better solves the problem of bicycle rebalancing.

4.1. Markov Chain

A four-tuple (S, Ac, R, γ) represents a Markov process [5], where S represents the state space, Ac represents the action space, R represents the immediate reward space, and γ represents the discount factor. As shown in Figure 3, after the current state S_t receives action Ac_t , the environment provides feedback $R(S_t, Ac_t)$, and the current state is transferred to S_{t+1} . As time goes on, a Markov chain can be obtained. Our bicycle rebalancing aims to consider the long-term benefits after performing actions in the face of the current state. In time series prediction, the greater the amount of future prediction results, the worse the accuracy. In addition, the current available gain [5] is more important than the future gain. Therefore, γ represents a gain–loss factor relative to future time.

Lemma 1. *The future gain of any Markov chain can be a formula, and the recursive relation can also be a formula, as follows:*

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots = R_t + \gamma G_{t+1}. \quad (11)$$

If we can determine the future gain of the Markov chain, we can choose the optimal incentive strategy (Action) at each time stage to obtain a Markov chain $S_0, A_0, R_0, S_1, \dots, S_n$ with the largest gain. In this Markov chain, we can obtain the incentive strategy of the whole stage, thus completing our rebalancing task. Determining the future benefits of each Markov chain is a huge project so that we can acquire the gain expectation of future possible states.

Lemma 2. *The state-action value function [5] of the Markov chain can be a formula, and the recursive relation can also be a formula, as follows:*

$$V(S_t, Ac_t) = \mathbb{E}[R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \dots | S_t, Ac_t] = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} | S_t, Ac_t\right]. \quad (12)$$

Note that $V(S_t, A_t)$ is a conditional expectation for state S_t and action A_t .

The state space and action space in MDP are continuous high-dimensional spaces. After we use the numerical discretization method to change the infinite action space into a finite set, even if the optimal action is selected by the instantaneous reward rather than the state-action value function, the workload of selecting the optimal action needs a lot of computation and storage space. Furthermore, the probability function cannot determine the action, for it is unknown. We must estimate the probability of selection action through random strategies with many samples. This work is huge, and the results are random. We use the deterministic strategy to describe the relationship between the current state and action, that is, $A_t = \pi(S_t)$. This part is defined as the *actor module* (as shown in Figure 1). To obtain the optimal strategy, we use the state-action value function to guide the promotion of the optimal strategy. This part is defined as the *critic module*.

4.2. Actor Module

In Figure 1b, we demonstrate the framework of the actor module. The actor module is no longer a traditional MLP but introduces GRU [11], embedded models, and GCN modules [13,14]. The input data of the actor module are the state in the environment, defined as a multidimensional vector in Section 3.3. In addition, we found that the more time information is contained, the more beneficial it is for the model to develop incentive strategies. When this conclusion maps to the real world, the state change in the bike network has a certain delay due to the bike journey with time span; it is not instantaneous. Facing the historical state sequence, we use the GRU module to extract relevant time characteristics.

$$X_t = \text{GRU}(S_{t-K}, S_{t-K+1}, \dots, S_t), \quad (13)$$

where $0 \leq K \leq T$ indicates the number of historical stages.

In addition, the context information in the bicycle environment, such as weather and temperature, is also related to time. We encode the context information and add it to the feature after extracting time information through the embedded model E .

$$\tilde{X}_t = X_t + E. \quad (14)$$

The attributes of nodes that constitute the state vector change along with the relationship between nodes. The environment part of Figure 1a shows the partial node relationship diagram. Information that can be obtained from the diagram is $N(v_6) = \{v_1, v_4, v_5\}$ and $N(v_4) = \{v_3, v_7\}$. The incentive policy of node v_6 receives the influence of v_1, v_4, v_5 and then the secondary effects of v_2, v_3, v_5, v_7 . Therefore, the graph structure in the entire network eventually affects the incentive actions for each station. We define matrix A to represent the relationship between nodes in the network. \tilde{A} represents the relational matrix with self-edges, with $\tilde{A} = A + I$, where I is the identity matrix. In graph theory, A represents the relationship between first-order neighbors, and A^2 represents the relationship between second-order neighbors. The graph convolution network contains the information of matrix A , so it can combine the characteristics of nearby nodes. Therefore, the formula of the graph convolution network is as follows:

$$h^{(l)} = \text{GCN}(h^{(l-1)}) = \sigma(Lh^{(l-1)}\Theta_l + b_l) \in R^{N \times C'}, \quad (15)$$

where L is the Laplacian matrix of A when $h^{(l-1)} \in R^{N \times C}$ is the input of the l_{th} graph convolutional layer, $h^{(l)}$ is its output, $\Theta_l \in R^{C \times C'}$, $b_l \in R^{C'}$ are learnable parameters, and σ represents the activation function, for example, \tanh . $h^{(0)}$ represents \tilde{X}_t . Not only does graph convolution network extract the spatial characteristics of nodes, but it also reduces the difficulties caused by dimensions. Due to the limitation of matrix A , the price (action) of every node is just related to several adjacent nodes rather than the whole nodes.

Actor module contains GRU, the embedded model, and GCN. The output is the incentive strategy (*action*). When the difference between any two incentive strategies Ac and Ac' is very small, the feedback gap from Bike Gym is not large during the model's learning process. Therefore, we add a small discrete function after the action value, preventing Bike Gym from falling into a dead cycle. The incentive strategy is inputted into the step function of Bike Gym.

4.3. Critic Network

As mentioned in Section 4.1, we need a *state-action value function* to evaluate the current state and action (incentive strategy) to guide the optimization of the actor module. In the critic module, we establish a deep network $Q_t = Q(S_t, Ac_t; \Phi)$ to approximate the real state-action value function $V(S_t, Ac_t)$. According to Lemmas 1 and 2, our deep network should also have the following relationships:

$$Q_t = R_t + \gamma Q_{t+1}. \quad (16)$$

Next, $Q_t = Q(S_t, Ac_t; \Phi)$ needs to be calculated. Similarly, facing the state space and action space of the bicycle environment, the dimension problem is still the focus of the solution. Seijen et al. proposed an HRA structure based on DQN [18], which decomposes the overall reward function into multiple components and uses the low dimensional function to approximate the overall value function [5]. The value function of each component depends on the feature subset, and it makes the overall value function smoother. Pan et al. [9] proposed an HRP structure [9] based on a DDPG [28] framework combining the factors of time and space. When the hierarchical structure is adopted, and the low-dimensional function is used to approximate the high-dimensional value function, a general formula is proposed to approximate the whole station-action value function, as follows:

$$V(S, A) \cong \sum_{i=1}^N v(s_i, a_i) + \sum_{i=1}^N f(N(s_i, a_i)). \quad (17)$$

The left side of Equation (17) represents the overall state-action value function [5]. The first part on the right represents the sum of the value functions of each node, and the second part on the right represents the sum of the value functions of local nodes. It can be seen from the formula that the optimization of the overall value function should focus on the sum of the value functions of local nodes.

In the bicycle network, the cross-neighbors of stations are defined, and the value function of the feature subset of the neighbors is used to smooth the overall value function. Based on the above theory and practice, we propose a general Equation (17) to approximate the state-action value function. In this module, the formula used is as follows:

$$V(S_t, Ac_t) = \sum_{i=1}^N (S_t || Ac_t) * \varphi_1 + \sum_{i=1}^N A \times (S_t || Ac_t) * \varphi_2, \quad (18)$$

where φ_1, φ_2 are learnable parameters. $||$ is a parallel symbol, indicating that variables S_t and Ac_t are transferred at the same time. The first item on the right side of the formula represents the sum of the independent value functions for all nodes. The second item on the right side of the formula represents the value function from the combined value function from the local nodes of every node. The processing idea of the critic module in this paper are somewhat similar to those of HRP, but our formula is more widely used than the HRP. When matrix A represents the neighborhood relationship of the cross, we can solve the problem which the HRP could solve.

4.4. Optimization Objective

We can temporarily determine a set of evaluation criteria $\Phi^* = (\phi_1^*, \phi_2^*)$, but there were better criteria initially. The more valuable the evaluation, the better the action guided by the critic produced by the actor module. Therefore, the closer $Q(S_t, Ac_t)$ approaches the

real state-action value function $V(S_t, A_t)$, the better the evaluation criteria. According to Equation (16) and the actor module, we can obtain the objective function to optimize

$$Q_{target}(S_t, A_t) = \mathbb{E}[R(S_t, A_t) + \gamma Q(S_t + 1, A_t + 1 | \pi^*; \Phi^*)], \quad (19)$$

where $R(S_t, A_t)$ means immediate reward. A_{t+1} is determined on the next state S_{t+1} under the incentive strategy π^* provided by the actor module. The evaluation of next state S_{t+1} and next action A_{t+1} is computed through the evaluation criteria $\Phi^* = (\phi_1^*, \phi_2^*)$.

The loss function between the real gain value and the state-action value is constructed. The critic module is optimized through the backward propagation mechanism. The incentive strategy generated under the excellent evaluation criteria is excellent. The loss function is as follows:

$$Loss_{critic} = MSE(Q(S_t, A_t), Q_{target}(S_t, A_t)), \quad (20)$$

where MSE stands for mean square error. As mentioned above, the higher the evaluation, the better the action. The loss function of the actor module can be expressed as follows:

$$Loss_{actor} = -V(S_t, A_t; \Theta^*). \quad (21)$$

Therefore, the model is divided into four parts: actor, actor–target, critic, and critic–target. The actor–target and critic–target do not have a loss function, and their parameters are regularly duplicated from the actor and critic parts. A relatively smooth soft update method is adopted for the parameters of actor and critic, where τ is a hyperparameter.

$$\bar{\Theta} \leftarrow \tau \Theta + (1 - \tau) \bar{\Theta}, \quad (22)$$

$$\bar{\Phi} \leftarrow \tau \Phi + (1 - \tau) \bar{\Phi}. \quad (23)$$

5. Experiment

5.1. Data Analysis

The data set used for the experiment was obtained from the Nanjing public bicycle company (PBC). It contains data from 20 September to 26 September 2016. We obtained the user trajectories, including origin and destination stations and spatiotemporal information. After deleting the data inconsistent with the time and the blank site, there were 980,036 valid data pieces and 1089 actual stations. Each travel record included the user ID, travel start (end) time, travel start (end) station, station name, station classification, station longitude and latitude, and other information. The bicycle system has 34,729 bicycles (about 35,000), 227,835 users, 41,672 pile positions, and the average loading rate of 0.83.

In Figure 4, we show the spatial location of bicycle stations. Stations are divided into four types according to their colors: small margin type (SD), dispatching demand type (DB), dynamic balance type (SM), and key protection type (KG). The operating company determines the division standard according to the frequency and flow difference of vehicle borrowing and returning at the station. As seen in the figure, the collection of different types of stations is overlapping and cannot be completely separated according to geographical location. In the ellipse, the distribution of stations near Zhongshan Road is enlarged. It can be seen from the ellipse that there are different sizes of bubbles. The bubble size indicates the maximum number of bikes stored at the station. This information implies that nodes in the bicycle network have strong heterogeneity. In addition, bicycle stations are mostly distributed along the traffic trunk road, and the bicycle journey trajectory (such as the black dotted line in the ellipse) is closely related to the traffic route. Using the time distance between bicycle stations is more reasonable than the straight distance between stations. Since the cycling speed of bicycles is relatively stable, we use the average cycling time between stations to represent the distance between stations to avoid obtaining a large number of calculations of track length between stations.

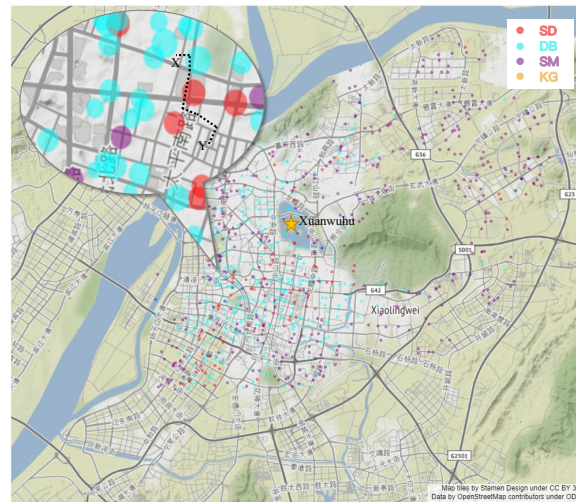


Figure 4. The scatter diagram of bicycle stations in Nanjing. This figure is the distribution map of four types of stations with Xuanwuhu as the center of the map. As shown in the legend in the upper right corner of the figure, bicycle stations include small margin type (SD), dispatching demand type (DB), dynamic balance type (SM), and key protection type (KG). The ellipse shows an enlarged version of the site distribution map near Zhongshan Road street. The black dotted line between station X and station Y represents the travel path of a bicycle.

Next, we look at the temporal characteristics of the bike's borrowing or returning behavior data in Figure 5. Figure 5a,b are the borrowing frequency diagrams of bicycle stations (not shown because the return frequency diagrams are similar). It can be seen from the figure that there are two peaks in the morning (7:00–9:00) and evening (17:00–19:00) on weekdays except Thursday. This phenomenon is called the tide phenomenon. We check the weather conditions on Thursday and find that it rained in the afternoon, reducing people's bicycle travel behavior. On weekends, the tide phenomenon is not obvious. This case shows that people's travel demand is greatly affected by commuting. The direct reason for the BRP is the imbalance between user demand and supply of bicycle stations. The changing trend of the net flow of bicycle stations over time is shown in Figure 5c, and the changing trend of the difference between inflow and outflow is shown in Figure 5d. The peak period in the morning and evening is still the period with the largest fluctuation in the net flow. In addition, the imbalance of working days is much more serious than that of weekends. Figure 5a–d shows the overall performance of all sites, meaning that most nodes have this characteristic performance.

Figures 4 and 5 show the spatial and temporal characteristics of bicycle networks, respectively. Processing spatiotemporal data is one of the important features of our model, which is also its original intention. Our model is based on long-term interests, so we can prepare in advance in the face of *tidal phenomena* to alleviate the imbalance between supply and demand in the bicycle environment.

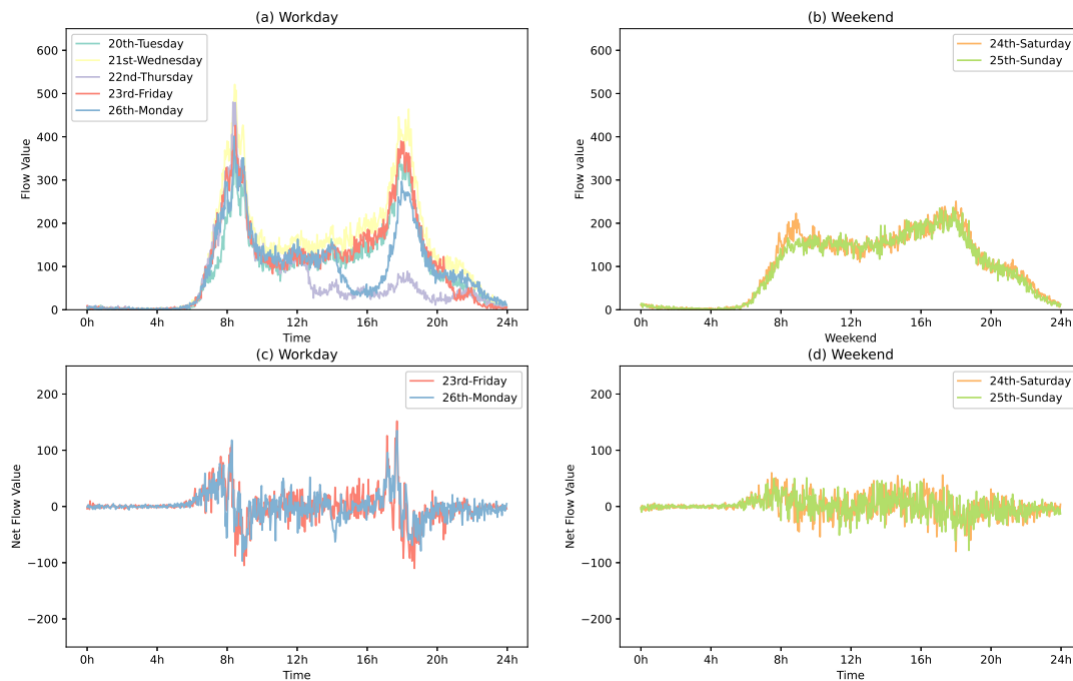


Figure 5. The curves of bike flow and net flow. (a) The flow curve of bicycle stations on weekdays, with obvious tide phenomenon. (b) The graph shows the flow curve at the weekend, and the tide phenomenon is not obvious. In (c,d), the net flow curve of bicycles is presented on two days each. The absolute value of the net flow in (c) is very large in the morning and evening, which is also one of the reasons for the imbalance of bicycles.

5.2. Experimental Setup

Using the method of deep reinforcement learning to solve the problem of bicycle scheduling rebalancing, we adopted three existing baselines:

- DDPG [28]: Deep Deterministic Policy Gradient algorithm. It can solve the small problem with low dimensional space.
- HRA [19]: Hybrid Reward Architecture decomposes the whole reward function into many components, which only depends on a subset of all features to solve a high-dimensional representation using a deep network.
- HRP [9]: Hierarchical Reinforcement Pricing Algorithm is a deep reinforcement learning algorithm that builds upon DDPG and uses a divide-and-conquer structure with an embedded localized to capture spatial and temporal dependencies.

BRIS is used to rebalance bicycles based on greed. Theoretically, as long as the temptation is big enough, all users can be fully inspired to complete all rebalancing work. It shows that our method has feasible solutions. Then, considering the BRP from the user's perspective, we adopt the reinforcement learning method to reduce the cost of bicycle rebalancing. Bike Gym should provide strong negative feedback if the purpose is violated. Therefore, detection is added in the *step function* according to Equation (5). If the detection is unqualified, Bike Gym provides a maximum negative value to the export reward. If unqualified detection is in the process of initializing the memory pool in the Bike Gym, the memory pool is cleared. In the process of resetting, the threshold range of the incentive strategy (action) is narrowed until the detection is qualified. This work can force the reduction in costs and optimize the model in the right direction from the beginning.

The memory pool is a space for storing interactive experience bars, and the number of experience bars stored is limited. It has a queue structure. The first experience bar is discarded when the storage is full, and a new experience is added. It ensures that the latest experience in the memory pool is always stored for a period to improve the STRPA model. The position of each experience bar is $(S_{t-K}, \dots, S_t, A_t, R_t, S_{t+1}, D)$, where $D = 1$

indicates the termination phase; otherwise, it is not terminated. It is worth mentioning that our memory pool is different from the buffer we temporarily built in initializing actions (as shown in Section 3.4).

In the general training process of reinforcement learning, a small exploration probability of ε is set to randomly generate action values and the probability $1 - \varepsilon$ is used to determine action values generated by the action module. That setting is implemented to increase the possibility of exploration.

Therefore, at the beginning, the action module randomly generates actions without criterion, and the critic module randomly evaluates without a target. After many experiments, the action module and the critic module are placed on the right track. But this must require a lot of useless work for our large state and action spaces. Firstly, the critic module is used to load the experience bars in batch $B = 256$ from the memory pool that Bike Gym effectively initialized to improve the evaluation criteria of the critic module. Then, the method of exploring actions is changed. Probability ε_1 to randomly generate actions and probability of ε_2 to acquire actions by neighborhood search algorithm, as well as the probability to determine actions by action module $1 - \varepsilon_1 - \varepsilon_2$, are employed. This method can be used through ε_1 to expand the scope of action exploration and use ε_2 to determine local effective actions. This improves the training speed. However, it is highly likely that the model cannot find the optimal solution, and it exports a suboptimal solution.

Our model must divide time discretely, and its time interval is set to $\Delta t = 10$ min. When GRU extracts time features in the action module, the number of whole-time stages is set to $K = 6$. Therefore we extract the time information characteristics of the past hour. In the critic model, we must import the state and action vectors. When the two are not at the same quantity level, the accuracy of the state action value function is compromised. Therefore, we normalize the input data through the maximum and minimum and define the reverse operation. In addition, the relationship matrix of nodes may be transformed to an unweighted matrix by threshold or a weighted matrix through the maximum and minimum. The normalization of data is beneficial to the stability of the model and the reduction in training difficulty.

5.3. Model Component

Compared with the traditional model DDPG [28], our model introduces great changes in both the actor module and the critic module. The change in different models causes the accuracy of the model to be improved. Therefore, this paper designs two contrast models: STRPA-A and STRPA-C. The structure of the actor module remains unchanged in STRPA-A, while the approximation formula of the state-action value function in the critic module is as follows:

$$Q(S, Ac) = \sum_{i=1}^N V(S_i, Ac_i), \quad (24)$$

where $V(S_i, AC_i)$ is the state-action value function of node v_i . The left of Equation (24) is the state-action value function of the whole network.

In STRPA-C, the structural design of the critic module in STRPA is inherited, while in the actor module, the incentive strategy is formulated according to the following formula:

$$Ac_t^i = \Theta \times \tilde{X}^i + b \quad (\forall v_i \in V). \quad (25)$$

It is worth noting that each node is decomposed without relation to its neighbor nodes. The current state of a node determines the current policy of the node. At the same time, the formula also ignores the heterogeneity of nodes. In these two replacement models, we separate the node independently. It eliminates the impact of matrix A .

5.4. Adjacency Matrix

The action module and the critic module in Section 4 both influence the relationship matrix A . Different matrices represent different relationships between nodes. In this paper,

we set two kinds of matrix relations. The nodes in the bicycle network are attached with relevant information on the geographical location, and the distance between the nodes can be obtained. The spatial topology between the nodes can be obtained by setting the threshold according to the distance. The matrix defined according to the proximity relationship between nodes is called the neighborhood matrix (NM) in this paper. In addition, adjacent nodes are not necessarily closely related. The matrix is obtained according to the close relationship between the nodes. In this paper, we call it the pattern matrix (PM). According to whether the thresholding method is used, these two matrices can also be divided into the unweighted neighbor adjacency matrix (NAu), the weighted neighbor adjacency matrix (NAw), the weighted pattern adjacency matrix (PAu), and the weighted pattern adjacency matrix (PAw). Four adjacency matrixes are brought into the model to test the effect of the model.

6. Visualization

In Section 5.2, we set three baselines. Compared with DDPG, the structure in other models in the critic module is established based on the hierarchical principle. Although, theoretically speaking, DDPG can complete the task of reinforcement learning. However, because the number of nodes is too large during training, as shown by the gray dashed line in Figure 6, the loss value of DDPG cannot drop for a long time, and the model cannot reach a stable state. In the structure of the actor module and the critic module, the loss value of other models fluctuates and decreases with the increase in training rounds and finally tends to be stable. To better show the effect of the models, we can display the performance of these models in Table 1. Key performance includes Q loss, A loss, MAE, and RMSE. Q loss represents the loss value of the criticism module of the different models trained. The loss represents the loss value of the participant module of the different models trained. MAE and RMSE represent the average absolute error and root mean square error between the reward predicted by the training model and the reward replayed by the environment. We provide the maximum, minimum, and average values of the key performance of the model.

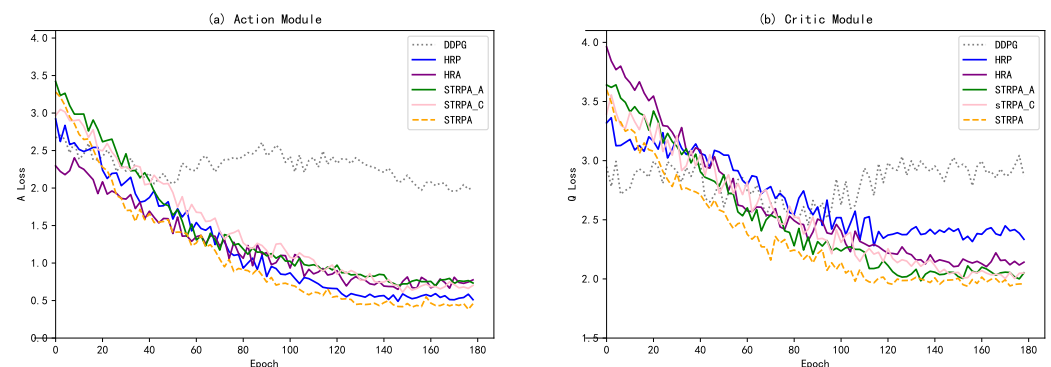


Figure 6. These are the line charts of action module loss and critic module loss from six different models. (a) The action module of STRPA has better convergence speed than others. (b) The critic module of STRPA has smaller loss value than others.

It can be seen from Figure 6 that the loss value of DDPG model fluctuates with the training rounds. In contrast, the loss value of HRA, HRP, STRPA, STRPA-A, and STRPA-C fluctuates with the increase in training rounds, indicating that the model can be optimized and tend toward stability. From the values of the ordinates in Figure 6a,b, it can be seen that the loss value of the critic network is higher than the actor loss value. This is because the loss value in the critic network is related to the parameters in the critical network and the parameters in the actor network. The physical meaning of actor loss represents the average expected value of the value of the state action of the node. When the number of nodes is fixed and the bicycle environment is unchanged, it can be seen that the lower the expectation of the state action value of the node, the lower budget for rebalancing the

bicycle. Therefore, the rebalancing cost of the bike is reduced. The physical meaning of the loss function of the critic module represents the difference between the state value function of the node and the true gain returned. The lower the difference, the better the effect of the simulated real return value of the critical network. In Figure 6, STRPA demonstrates its outstanding capabilities in both the actor module and the critic module.

Table 1. The performance of five models. *Q Loss* indicates the loss value from the critic module; *A Loss* indicates the loss value from the actor module; MAE indicates mean absolute error between prediction and targets; RMSE indicates mean root mean squared error between prediction and targets.

Model	Keywords	Max	Min	Mean
DDPG	Q Loss	4.93	0.89	3.19
	A Loss	0.58	−2.61	−1.24
	MAE	703.83	120.30	398.27
	RMSE	77.23	34.68	58.12
HRA	Q Loss	4.12	1.70	2.59
	A Loss	1.13	0.67	0.84
	MAE	161.49	87.88	115.93
	RMSE	39.95	25.82	30.69
HRP	Q Loss	8.07	3.56	4.72
	A Loss	2.52	1.91	2.05
	MAE	151.24	54.05	69.65
	RMSE	38.89	18.34	28.02
STRPA	Q Loss	3.21	1.21	1.81
	A Loss	2.17	1.59	1.81
	MAE	165.15	28.43	60.29
	RMSE	39.02	16.87	30.64
STRPA-A	Q Loss	5.33	1.93	3.27
	A Loss	1.77	1.14	1.36
	MAE	188.64	51.73	70.53
	RMSE	48.34	25.40	30.71

We can see that models STRPA, STRPA-A, and STRPA-C can become stable after training and can be used to formulate a pricing strategy (action) for node detours in the bicycle network. To better compare the performance of the model, we used the same matrix and the model performance in different periods under the same bicycle environment. The sub-chain of three periods is extracted in the Markov chain. These three time periods include morning peak, afternoon peak, and evening peak, and are defined as the morning stage (M), the noon stage (N), and the evening stage (E).

The bar graph in Figure 7 represents the MAE and RMSE mean values of the three models in three periods under the unauthorized neighborhood adjacency matrix. From the perspective of MAE and RMSE of the model, the STRPA model performs best, followed by STRPA-A. In terms of time stage, the best result is at night. Because bicycle travel is more susceptible to commuting, the effect of its node detour strategy at night is more easily compromised, thus achieving good results.

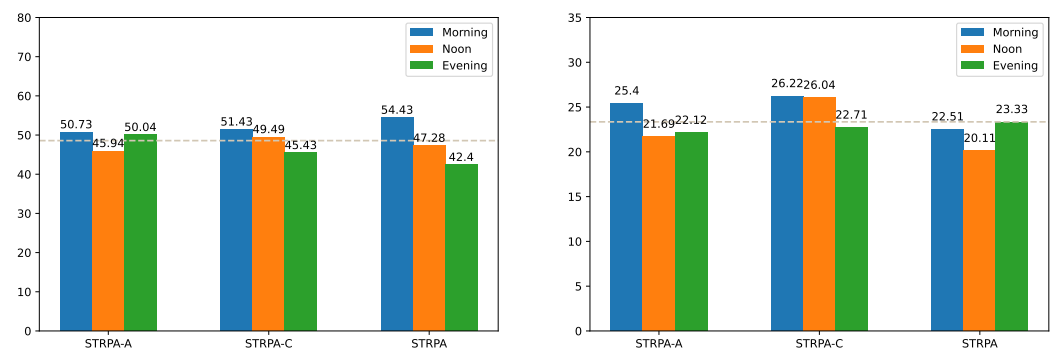


Figure 7. The MAE and RMSE of STGRA with different components. The MAE values of STRPA, STRPA-A, STRPA-C in different time on left subfigure. The RMSE values of STRPA, STRPA-A, STRPA-C in different times on right subfigure. The dashed line represents the mean value of all results.

Graph convolution network can mainly aggregate the information of the neighborhood nodes through the adjacency matrix, so the different adjacency matrices result in different impacts of certain models. It can be seen from Section 5.4 that there are four kinds of matrices, including NAu , NAw , PAu , and PAw . The STRPA model with four matrices is selected to explore the influence of the matrix. It can be seen from the three charts that the performance of the model with different matrices is excellent, respectively, in different periods. In general, the performance of the unweighted matrix is better than that of the weighted matrix, and the performance of the neighborhood adjacency matrix is better than that of the pattern adjacency matrix. The unweighted matrix is more accessible to obtain than the weighted matrix, and it can minimize the difficulty of the model. The reason for the better performance of the neighborhood matrix is that the node relationship assumed to influence detour decisions is more likely the neighbor relationship constructed by the geospatial structure than the patterned relationship constructed by the similarities between node characteristics. From this, we can see that the construction of our matrix should be closer to the physical meaning of the model.

In Figures 8a,b and 9, we show the reward curves of three models in the early, middle, and late periods. It can be seen that the STRPA model is the best among the three models in terms of the stable performance of reward and the fluctuation of the curve, and we can see that the performance of the three models in the same period has little difference in reward. On the contrary, the performance difference of the same model in different periods is a little larger. This is due to the degree of internal imbalance of the bicycle network, which is independent of the model itself.

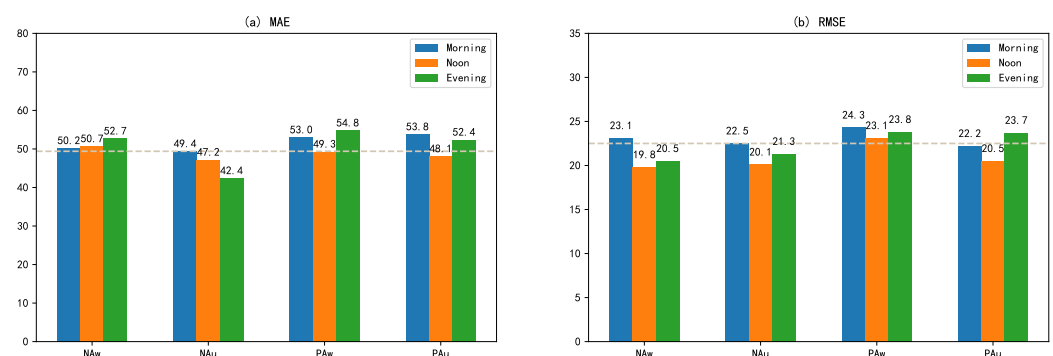


Figure 8. The curve diagram of loss from STRPA using different adjacency matrices. Matrices represent different node relationships during the different time. (a) The MAE values in different adjacency matrices; the performance of matrix NAu is best. (b) The RMSE value in different adjacency matrices; the performance of matrix NAu is best.

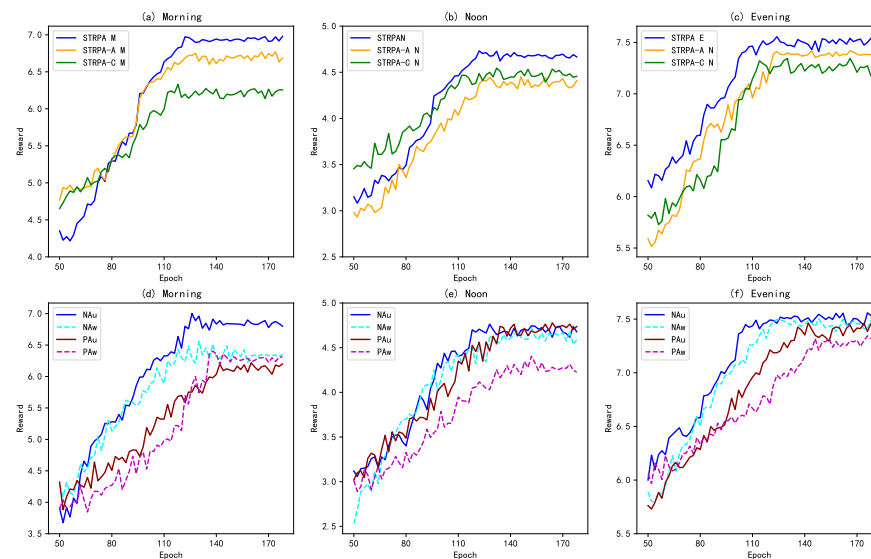


Figure 9. Curve diagrams of the reward given by environment in different models and in different matrices. Figure (a–c) show the rewards from STRPA, STRPA-A, STRPA-C during the morning, noon, and evening, respectively. STRPA performs the best, as shown in three subfigures. Figure (d–f) show the rewards from STRPA with different matrices during the morning, noon, and evening, respectively. STRPA performs the best with matrix *NAu*, as shown in Figure 9.

7. Conclusions

In this paper, a red envelope with money is used to inspire users to complete the task of bicycle rebalancing from a long-term perspective. The difficulty of this problem lies in determining the incentive tasks for each station. To better balance the incentive tasks, we consider the gain of the incentive strategy from a long-term perspective. We use the Markov decision theory to solve this problem. In this paper, we mainly perform two tasks. In Section 3, we simulate a general bicycle environment system—Bike Gym. In Section 4, the model framework STRPA is established to find the optimal incentive strategy. Based on the built Bike Gym, we verify the Nanjing public bicycle data set well. The model can extract spatiotemporal information, tolerate the heterogeneity of nodes, and can be applied to the problem of multiple node relationships. Inspired by this, the model can be applied to other traffic data sets such as rail transit, public electric vehicles, and shared cars.

Author Contributions: Conceptualization, B.P.; methodology, B.P.; software, B.P.; validation, B.P., L.T. and Y.P.; formal analysis, B.P.; investigation, B.P.; resources, B.P.; data curation, B.P.; writing—original draft preparation, B.P.; writing—review and editing, B.P.; visualization, B.P.; supervision, B.P.; project administration, B.P.; funding acquisition, L.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Key R&D Program of China (Grant No. 2020YFA0608601), National Natural Science Foundation of China (Grant No. 72174091 a) and Major programs of the National Social Science Foundation of China (Grant No. 22&ZD136).

Data Availability Statement: The data are unavailable due to privacy or ethical restrictions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shaheen, S.A.; Guzman, S.; Zhang, H. Bikesharing in Europe, the Americas, and Asia. *Transp. Res. Rec. J. Transp. Res. Board* **2010**, *2143*, 159–167. [CrossRef]
2. Beijing Transport Institute. Beijing Transport Annual Report. 2021. Available online: <https://www.bjtrc.org.cn> (accessed on 10 March 2021).
3. Liu, J.; Sun, L.; Chen, W.; Xiong, H. Rebalancing Bike Sharing Systems: A Multi-source Data Smart Optimization. In Proceedings of the 22nd ACM SIGKDD International Conference, San Francisco, CA, USA, 13–17 August 2016.

4. Singla, A.; Santoni, M.; Bartók, G.; Mukerji, P.; Meenen, M.; Krause, A. Incentivizing Users for Balancing Bike Sharing Systems. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015; AAAI Press: Palo Alto, CA, USA, 2015; Volume 29. [\[CrossRef\]](#)
5. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*, 2nd ed.; A Bradford Book: Cambridge, MA, USA, 2018.
6. Lin, S.; Kernighan, B.W. An Effective Heuristic Algorithm for the Traveling-Salesman Problem. *Oper. Res.* **1973**, *21*, 498–516. [\[CrossRef\]](#)
7. Fricker, C.; Gast, N. Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity. *EURO J. Transp. Logist.* **2016**, *5*, 261–291. [\[CrossRef\]](#)
8. Chemla, D.; Meunier, F.; Pradeau, T.; Calvo, R.W.; Yahiaoui, H. Self-service bike sharing systems: simulation, repositioning, pricing. *Dyn. Routing* **2013**. Available online: <https://api.semanticscholar.org/CorpusID:5954769> (accessed on 12 August 2023).
9. Pan, L.; Cai, Q.; Fang, Z.; Tang, P.; Huang, L. A Deep Reinforcement Learning Framework for Rebalancing Dockless Bike Sharing Systems. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019. Volume 33, pp. 1393–1400.
10. Ghosh, S.; Trick, M.; Varakantham, P. Robust Repositioning to Counter Unpredictable Demand in Bike Sharing Systems. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI’16, New York, NY, USA, 9–15 July 2016; pp. 3096–3102.
11. Ballakur, A.A.; Arya, A. Empirical Evaluation of Gated Recurrent Neural Network Architectures in Aviation Delay Prediction. In Proceedings of the 2020 5th International Conference on Computing, Communication and Security (ICCCS), Patna, India, 14–16 October 2020; pp. 1–7. [\[CrossRef\]](#)
12. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [\[CrossRef\]](#) [\[PubMed\]](#)
13. Bruna, J.; Zaremba, W.; Szlam, A.; Lecun, Y. Spectral Networks and Locally Connected Networks on Graphs. *arXiv* **2013**, arXiv:1312.6203.
14. Sandryhaila, A.; Moura, J. Discrete Signal Processing on Graphs. *IEEE Trans. Signal Process.* **2013**, *61*, 1644–1656. [\[CrossRef\]](#)
15. Lin, L.; He, Z.; Peeta, S. Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach. *Transp. Res. Part C Emerg. Technol.* **2018**, *97*, 258–276. [\[CrossRef\]](#)
16. Ai, Y.; Li, Z.; Gan, M.; Zhang, Y.; Yu, D.; Chen, W.; Ju, Y. A deep learning approach on short-term spatiotemporal distribution forecasting of dockless bike-sharing system. *Neural Comput. Appl.* **2019**, *31*, 1665–1677. [\[CrossRef\]](#)
17. Li, Y.; Zhu, Z.; Kong, D.; Xu, M.; Zhao, Y. Learning Heterogeneous Spatial-Temporal Representation for Bike-Sharing Demand Prediction. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, AAAI’19, Honolulu, HI, USA, 29–31 January 2019. [\[CrossRef\]](#)
18. Volodymyr, M.; Koray, K.; David, S.; Rusu, A.A.; Joel, V.; Bellemare, M.G.; Alex, G.; Martin, R.; Fidjeland, A.K.; Georg, O. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533.
19. Seijen, H.V.; Fatemi, M.; Romoff, J.; Laroché, R.; Barnes, T.; Tsang, J. Hybrid Reward Architecture for Reinforcement Learning. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
20. Dantzig, G.B.; Ramser, J.H. The Truck Dispatching Problem. *Manag. Sci.* **1959**, *6*, 80–91. [\[CrossRef\]](#)
21. Szeto, W.Y.; Liu, Y.; Ho, S.C. Chemical reaction optimization for solving a static bike repositioning problem. *Transp. Res. Part D Transp. Environ.* **2016**, *47*, 104–135. [\[CrossRef\]](#)
22. Szeto, W.Y.; Shui, C.S. Exact loading and unloading strategies for the static multi-vehicle bike repositioning problem. *Transp. Res. Part B Methodol.* **2018**, *109*, 176–211. [\[CrossRef\]](#)
23. Wang, Y.; Szeto, W. Static green repositioning in bike sharing systems with broken bikes. *Transp. Res. Part D Transp. Environ.* **2018**, *65*, 438–457. [\[CrossRef\]](#)
24. Liu, Y.; Szeto, W.; Ho, S. A static free-floating bike repositioning problem with multiple heterogeneous vehicles, multiple depots, and multiple visits. *Transp. Res. Part C Emerg. Technol.* **2018**, *92*, 208–242. [\[CrossRef\]](#)
25. Caggiani, L.; Camporeale, R.; Ottomanelli, M.; Szeto, W.Y. A modeling framework for the dynamic management of free-floating bike-sharing systems. *Transp. Res. Part C Emerg. Technol.* **2018**, *87*, 159–182. [\[CrossRef\]](#)
26. Cheng, Y.; Wang, J.; Wang, Y. A user-based bike rebalancing strategy for free-floating bike sharing systems: A bidding model. *Transp. Res. Part E Logist. Transp. Rev.* **2021**, *154*, 102438. [\[CrossRef\]](#)
27. Li, Y.; Liu, Y. The static bike rebalancing problem with optimal user incentives. *Transp. Res. Part E Logist. Transp. Rev.* **2021**, *146*, 102216.
28. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
29. Duan, Y.; Wu, J. Optimizing Rebalance Scheme for Dock-Less Bike Sharing Systems with Adaptive User Incentive. In Proceedings of the 2019 20th IEEE International Conference on Mobile Data Management (MDM), Hong Kong, China, 10–13 June 2019.
30. Savić, M.; Ivanović, M.; Jain, L.C., Introduction to Complex Networks. In *Complex Networks in Software, Knowledge, and Social Systems*; Springer International Publishing: Cham, Switzerland, 2019; pp. 3–16. [\[CrossRef\]](#)

31. Boccaletti, S.; Bianconi, G.; Criado, R.; Del Genio, C.; Gómez-Gardeñes, J.; Romance, M.; Sendiña-Nadal, I.; Wang, Z.; Zanin, M. The structure and dynamics of multilayer networks. *Phys. Rep.* **2014**, *544*, 1–122. [[CrossRef](#)] [[PubMed](#)]
32. Sutton, R.S.; McAllester, D.; Singh, S.; Mansour, Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99, Denver, CO, USA, 29 November–4 December 1999; MIT Press: Cambridge, MA, USA, 1999; pp. 1057–1063.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.