Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6890601, 14 pages
https://doi.org/10.1155/2020/6890601

Research Article

GPS Data in Urban Online Car-Hailing: Simulation on Optimization and Prediction in Reducing Void Cruising Distance

Yuxuan Wang,[1] **Jinyu Chen**,[1] **Ning Xu**,[2] **Wenjing Li**,[1] **Qing Yu**,[1,3] **and Xuan Song**[1,4]

[1]*Center for Spatial Information Science, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa Chiba 277-8563, Japan*
[2]*Beijing Key Laboratory of Urban Oil and Gas Distribution Technology, China University of Petroleum-Beijing, Fuxue Road No. 18, Changping District, Beijing 102249, China*
[3]*Key Laboratory of Road and Traffic Engineering of the Ministry of Education, Tongji University, 4800 Cao'an Road, Shanghai 201804, China*
[4]*SUSTech-UTokyo Joint Research Center on Super Smart City, Department of Computer Science and Engineering, Southern University of Science and Technology (SUSTech), Shenzhen, China*

Correspondence should be addressed to Xuan Song; songxuan@csis.u-tokyo.ac.jp

Received 30 July 2020; Revised 16 October 2020; Accepted 3 November 2020; Published 25 November 2020

Academic Editor: Jie YAN

Ride-hailing, as a popular shared-transportation method, has been operated in many areas all over the world. Researchers conducted various researches based on global cases. They argued on whether car-hailing is an effective travel mode for emission reduction and drew different conclusions. The detailed emission performance of the ride-hailing system depends on the cases. Therefore, there is an urgent demand to reduce the overall picking up distance during the dispatch. In this study, we try to satisfy this demand by proposing an optimization method combined with a prediction model to minimize the global void cruising distance when solving the dispatch problem. We use Didi ride-hailing data on one day for simulation and found that our method can reduce the picking up distance by 7.51% compared with the baseline greedy algorithm. The proposed algorithm additionally makes the average waiting time of passengers more than 4 minutes shorter. The statistical results also show that the performance of our method is stable. Almost the metric in all cases can be kept in a low interval. What is more, we did a day-to-day comparison. We found that, despite the different spatial-temporal distribution of orders and drivers on different day conditions, there are little differences in the performance of the method. We also provide temporal analysis on the changing pattern of void cruising distance and quantity of orders on weekdays and weekends. Our findings show that our method can averagely reduce more void cruising distance when ride-hailing is active compared with the traditional greedy algorithm. The result also shows that the method can stably reduce void cruising distance by about 4000 to 5000 m per order across one day. We believe that our findings can improve deeper insight into the mechanism of the ride-hailing system and contribute to further studies.

1. Introduction

Ride-hailing, referring to the activity of calling a vehicle or driver to go to a destination, rises in many metropolises and takes a considerable part of mobility services [1]. The studies on the behavior of ride-hailing mount to a peak in recent years [2]. Some discussed the impact of ride-hailing on urban transportation behavior. Tang et al. [3] did an online app survey on 9762 respondents and found that about 35% of respondents are attracted from traditional taxis; 37% are from public transportation. Some tried to discover how ride-hailing influences urban sustainability. Li et al. [4] employed annual ride-hailing data in the United States from Uber Google researches and found that the usage of ride-hailing can help reduce traffic congestions in big cities. Some put efforts into revealing the carbon footprint of ride-hailing. Sui et al. [5] operated spatial analysis on empirical evidence in Chengdu area. They concluded that ride-hailing saves more energy and emits less GHG on per passenger on kilometer basis during the service compared with the traditional taxi industry.

Among all these studies, some scholars explore the efficiency of novel technologies in the ride-hailing system. Bischoff, Kaddoura [6] proposed an agent-based simulation method to optimize the service area of ride-hailing. Feng et al. [7] built a stylized model of a circular road and compared the average waiting times of passengers under different matching mechanisms of ride-hailing. Calderón and Miller [8] proposed a method for modeling the within-day service provision process of ride-hailing service providers with limited data availability. However, the efficiency of real-time scheduling should be further considered in practical application. Korolko et al. [9] indicated that bipartite matching with time window batching and dynamic pricing can lower waiting time for both riders and drivers as well as capacity utilization, trip throughput, and total welfare. However, they only consider the dispatching in the real-time time window and did not consider the travel demand in the future. What is more, Afeche et al. [10] pointed out that the interference from the service platform to avoid dispatching drivers to the area with low travel demand can be optimal. These two conclusions inspire us with the idea that whether we can predict the distribution of travel demands in the future and whether it can help us optimize the dispatching of the ride-hailing system, especially improving the utility of energy. Among current studies, there is no existing literature quantifying and discovering this improvement. This gap needs to be fulfilled and further instructs the development of the ride-hailing system.

Therefore, there are some gaps in current studies:

(1) Considering the real-time matching of driver and passenger with the combination of optimization method and prediction model to minimize the void cruising distance as well as maximizing the energy utility.

(2) The simulation of the method on real-world data to prove the performance and applicability as well as an analysis of the spatial-temporal pattern of emission behavior of two different dispatch strategies.

Operating such kind of research is not an easy task. Solid and real travel demand data are required as the basis of simulation and assessment. Next, a reliable prediction model is in a dominant position in the whole simulation as imprecise prediction can bring misjudge to dispatching decisions. In addition, the dispatching methods should be designed carefully and the performance and applicability should be ensured.

With the development of urban data mining [11, 12], the simulation method based on historical GPS data allows us to develop and analyze the performance of methods. This paper will overcome these difficulties by employing and combing empirical ride-hailing data from the real world, reliable prediction model, and dispatching simulation with optimization methods. Also, we will give a comprehensive analysis of the behavior of the system on improving the energy utility of ride-hailing. We believe that our work can provide the guideline for future decision-making and development of ride-hailing.

In this study, we simultaneously employ a performance-proved prediction model and optimization-based matching strategy as the methodology and apply it to the simulation based on millions of empirical ride-hailing data from the real world to quantify this improvement. Then, we will also provide a comprehensive spatial-temporal analysis. The contribution of this work can be listed as follows:

(1) Proposing a simulation framework combining the optimization and prediction to improve the efficiency of the ride-hailing system.

(2) A simulation sample on millions of Didi ride-hailing record data to provide persuasive evidence of the utility of methodology.

(3) A comprehensive spatial-temporal pattern analysis and comparison that will be provided.

2. Materials and Methods

2.1. Research Framework. Different dispatch programs cause different energy behaviors. Among different dispatch algorithms, optimization is a hot topic [13]. However, the performance of the optimization algorithm depends heavily on current knowledge. In real cases, time window batch is a common method for processing of matching pool in real-time dispatch [14]. The longer the time window is, the more optimal the solution is, but it is lesser applicable because it will cause longer waiting times for users. It is very hard to balance the performance of the solution and the length of the time window. Therefore, merging prediction in the future is a good choice to enrich the knowledge and improve the performance of optimization. In this study, we propose a new dispatch framework and compare the energy behavior of it with one of the original dispatch plans. We use the following framework in this study (see Figure 1).

The research framework used in this part is shown in Figure 1. The Didi apps in users' (both passenger and driver) phones collect users' GPS data. The data source is from Chengdu, China, where the ride-hailing service has been operated for over 6 years. According to the report made by Didi Media Research Institute [15], the times of ride-hailing services are beyond the local taxi services and the ride-hailing system serves over 1.4 million times one day. Thus, the dataset is suitable for study. After receiving these data, we distract the position and timestamp of the appearance of the driver as well as the origin and destination of the passenger. These data also include the timestamps. We added one more step, which is to preprocess the OD data of orders into the desired format of the prediction model. The detailed steps will be elaborated in the methodology part. The proposed methodology can be divided into two parts: the prediction part and the dispatching part. The prediction part is mainly responsible for predicting the distribution of travel demand in the future based on a deep-learning method. The input of this deep-learning model requires historical observation and corresponding metadata; the output is the predicted spatial distribution of travel demand. The dispatching part focuses on optimizing the dispatch under the consideration of minimizing void cruising distance proportion based on the predicted distribution of travel demand in the future. To

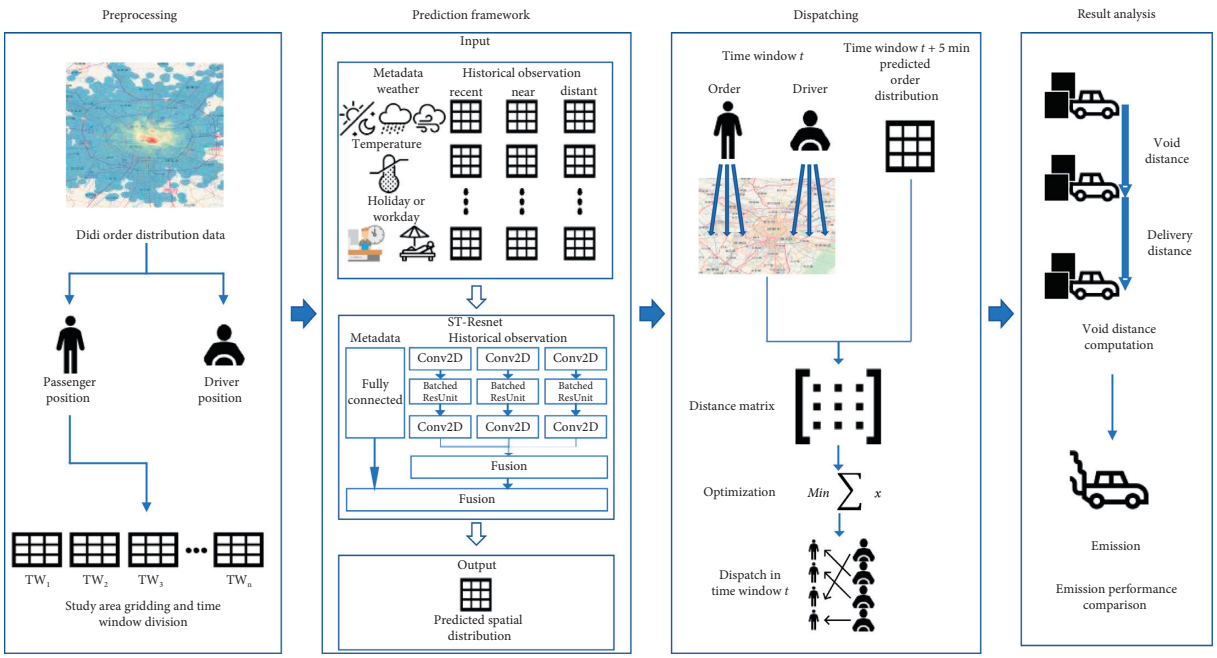


Figure 1: The framework used in this study.

better combine the two parts, we adopt the time window division method. The order and driver will be divided by serial time windows as input [16]. For each time window, the prediction and optimization method will be separately operated once to decide the assignment of the driver to order. To better show the utility of the proposed algorithm, we will use the greedy algorithm operated in the time window and dispatch in the original dataset as a baseline and compare the performance in result analysis.

The assumptions made in this part are shown as follows:

(1) The cost of each passenger and driver to adopt the ride-hailing will be considered. If a passenger waits for a driver to take the order longer than 15 minutes [17], the order will be canceled.

(2) The rejection of order out of the driver's personal issue is not allowed [18].

(3) The participation of the driver is in long term, which means that no driver will quit the system until the simulation is over [6].

(4) During idle time, the driver will park their cars nearby the drop-off location [19].

Finally, we give the metric to compute efficiency. As reviewed in literature reviews, a high percentage of void cruising distance traveled in ride-hailing operation was observed and reducing void cruising distance is an urgent effort. Thus, we choose the void cruising distance proportion as the main metric to measure energy behavior. This definition of void cruising distance proportion is defined as

$$P_v = \frac{d_v}{d_v + d_d}, \tag{1}$$

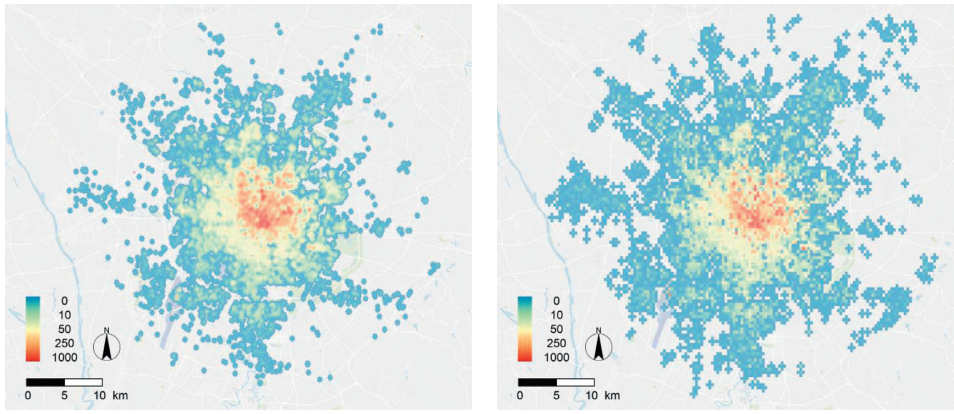
where P_v is the void cruising distance proportion; d_v is void cruising distance; d_d is the delivery distance.

As it can be seen from the definition, the lower void cruising distance proportion means lower invalid energy usage proportion in operation. In result analysis, we will compare the overall void cruising distance proportion between the former dispatch strategy and our proposed dispatch methodology. We will also provide a spatial-temporal analysis of the dispatch result. In work by Korolko et al. [9], they also took the waiting time of passengers into consideration. Thus, in this study, we will also consider the waiting time of passengers and the cancelation rate of orders. The definition of the waiting time is defined as

$$T_{\text{wait}} = T_{\text{match}} + T_{\text{pitch}}, \tag{2}$$

where T_{wait} is the waiting time of a passenger; T_{match} is the period from the time when the passenger places his or her order to the time when the order is dispatched successfully to a specified driver. It is calculated according to the length of the dispatch algorithm time window, which will be elaborated in Optimization in Dispatch section; T_{picth} is the time between a driver driving to a pick-up location and a user boarding. It is calculated according to the spatial distances of a specified driver to pick up the passenger.

2.2. Case Study. The raw dataset we use in this study is the ride-hailing record data from Didi cooperation. The dataset is collected by users' mobile phones and includes the information of order ID, driver ID, start time, end time, start location, and end location. The dataset describes the city of Chengdu, a super city with high traffic volume, emission, and energy consumption [20]. The date range of the dataset is from November 2, 2016, to November 30, 2016, without November 10, 2016. There is also an obvious data vacancy on November 8. We plot the heatmap of the spatial distribution of the dataset on one day as examples (see Figure 2).



(a) (b)

Figure 2: Thermal map of order OD data on one day in Chengdu. (a) Origins of all orders. (b) Destinations of all orders.

The study area covers the main areas of Chengdu city and includes some suburban areas. From an overall view, the travel demand is mainly distributed in the city center of Chengdu and the concentration decays with the radical extension.

2.3. Prediction Model. Travel demand prediction is currently a hot research topic in the field of computer science [21, 22]. The mainstream of methods is deep learning. Many researchers developed various deep-learning neural networks that concern this problem. In recent years, there exist a lot of achievements, like convolutional LSTM neural network [23]. The prediction model we use in this study is the ST-Resnet [24]. This is a deep-learning neural network based on the residual unit.

The input of this prediction model is divided into two parts, which are separately historical observation and metadata. The output is the spatial distribution in the future. The desired input format of historical observation is a matrix in essence. The preprocessing is needed to convert spatial data to the matrix. Firstly, we extract the order data in each time window. Next, we apply the regional grid method [25]. The concept is to convert the spatial distribution data to image-like data, which is in the form of matrixes. We divided the study area with a grid size of 40×40. The spatial size of each cell is $3690\,\mathrm{m} \times 3690\,\mathrm{m}$. If we set a larger grid size, although the number of predicted orders may be more accurate, more errors will be introduced to the position of predicted orders and affect the accuracy of the position of predicted orders. Next, this will affect the result of optimization. However, a small grid size is not necessarily good. A small grid size, which means more cells, will make many cells zero. So, matrixes will be very sparse. Too sparse input and output will corrupt the performance of the prediction model. An example is that the gird with low quantity may be probably predicted as zero and the total quantity of orders will be far from the ground truth. We did some experiments on the grid size and finally found the size of 40×40 as the determined one.

Then, we count the number of orders in the area of each cell in each time window. The quantity of orders is the element of each matrix. Finally, we can get a sequence of matrixes abstracted from spatiotemporal data. We choose 5 minutes as the time length of prediction because the time scale of prediction with minute magnitude can help provide enough future knowledge for the decision. However, a long time scale may keep some drivers waiting a long time for the next order. From preprocessing of the data in one month, we can get totally 8352 matrixes, out of the concern of stability of the training process and shortening the training time. In the prediction model, the input of observation is divided into three sequences, separately recent, near, and distant: recent: a sequence of continuous matrixes of historical observation closely before the time window we want to predict; near: a sequence of continuous matrixes of historical observation that is one day before near; distant: a sequence of continuous matrixes of historical observation that is one week before near. We choose the length of the sequence to be 6. Thus, if we use $x_n, n \in [2022, 8351]$ to represent the output, the input can be represented as

$$X \begin{cases} \{x_i | i \in [n-6, n-5, n-4, n-3, n-2, n-1]\}, \\ \{x_j | j \in [n-288-6, n-288-5, n-288-4, n-288-3, n-288-2, n-288-1]\}, \\ \{x_z | z \in [n-2016-6, n-2016-5, n-2016-4, n-2016-3, n-2016-2, n-2016-1]\}. \end{cases} \tag{3}$$

The input dimension of historical observation is $40 \times 40 \times 6 \times$ batch size.

There are three individual input channels separately for recent, near, and distant. We will introduce the structure of

one channel as all three channels are the same. The first layer is a 2D convolutional layer with a kernel size of 3×3 and 64 filters that extract the feature of the input sequence to the matrix of size $40 \times 40 \times 64$. Then, the following part is a sequence of residual units. The job of each residual unit is to deeper analyze the features. The longer the sequence is, the deeper mechanism can be extracted.

After an iteration in the sequence, the result will go through the final 2D convolutional layer and fuse together. The method of fusion is to add the matrixes from three channels together to one. Then, this one matrix will be added with the reshaped output feature of metadata. Finally, the summed-up feature will be handled by a Tanh function and turned out to be the output of the prediction result.

During the training process, we use the former 20% of the dataset as the test set and the latter 80% as the training set. The optimizer for the gradient descent is Adam [26], which has shown a better performance among all the optimizers.

Another part of the input is the metadata. Generally speaking, metadata includes all the information that can have an impact on the spatial distribution of order. In the original paper, the author used the weather data and date information as the metadata. For compliance with the original model, in this study, we marked the hour that the time window is located in one day, the day in one week, and the week in one year of each time window, the mark of whether the day is a holiday or not (holiday is marked as 1; workday is marked as 0) as the date information. Meanwhile, we also use the weather and temperature data as the weather data. The table of detailed weather data is shown as follows (see Table 1).

The temperature data will be rescaled to [−1, 1] by

$$\frac{\text{tem} - \text{tem}_{\min}}{\text{tem}_{\max} - \text{tem}_{\min}} \times \left(t_{\max} - t_{\min}\right) + t_{\min}, \tag{4}$$

where tem is the temperature we want to rescale; $\text{tem}_{\min}$ is the minimal observed temperature; $\text{tem}_{\max}$ is the maximal observed temperature; $t_{\max}$ is the upper bound of the rescaled interval; $t_{\max}$ is the lower bound of the rescaled interval.

The date information and weather information will be separately turned to numerical data by a one-hot encoding method [27]. Then, two one-hot encoded data will be concatenated together as one matrix. The feature of metadata information will be extracted by a fully connected layer with a Relu function shown as

$$\text{Relu}(x) = \begin{cases} 0, & x < 0, \\ x, & x \geq 0. \end{cases} \tag{5}$$

The output feature will be then reshaped to the size of $40 \times 40 \times 6$ as the same size of historical observation.

By far, we have completed the introduction of the prediction model. In the result analysis, we will illustrate the accuracy of the prediction model.

2.4. Optimization in Dispatch. Optimization is a classical mathematical method used in many research fields including the ride-hailing [28]. Generally speaking, the concept of optimization is to optimize the objective function and find a global solution.

In the dynamic ride-hailing dispatch problem, a widely used processing method is the time window division [29] (see Figure 3).

The process of time window division can be treated as a group of timeline. The end of each time window is also the beginning of the last one. Suppose that the length of the time window is l, the number of the time window is N, and the start time of simulation is 0. During the period of time window $n, n \in [1, 2, \ldots, N]$, the orders and available drivers given between time $(n-1)l$ and nl will be collected in the matching pool. Then, at the time nl, the simulation algorithm will be operated to give the matching result.

Baseline algorithm, greedy algorithm: the greedy algorithm is a classical algorithm used in many real-time dispatch studies of pick-up and delivery problems [18, 30]. In principle, the algorithm will iterate over every travel demand and find the closest driver who can pick up the order or follow the rule of first-come-first-serve [31]. Generally speaking, the algorithm only considers the optimized solution for each single object. Although this method is easy to implement and manage, it is naturally uncoordinated and tends to prioritize immediate passenger satisfaction over the global supply utilization. In the result and analysis part, we will illustrate the performance of the greedy algorithm.

In this study, we propose a dispatch strategy that integrates both optimization and prediction. Different from the strategy introduced before, the core of the algorithm is that when we consider the dispatch problem in time window n, the predicted distribution of orders in the next 5 minutes will also be taken into account.

At each time of execution, in addition to the orders that really exist in the current time window, the algorithm will add the orders predicted in the next 5 minutes to the matching pool. Then, the optimization algorithm will decide which and how orders in the current time window will be dispatched.

However, this does not mean that we simultaneously complete the dispatch problem in both the current time window n and the next 5 minutes. The dispatching problem in each individual time window is supposed to be solved independently. The prediction on the spatial distribution of orders in the future serves the purpose of enriching the knowledge in the current optimization problem.

Let us consider a dispatch problem with a low dimensionality of 2×2. In Figure 4, D refers to the driver; P refers to the passenger. The dashed object in the figure means that it is predicted; the solid one means it is existing. The distance marked near the arrow is the probable picking up distance. Without the preknowledge of the possible existence of passenger 2, it is obvious that driver 2 will be dispatched to passenger 1 and driver 1 to passenger 2, while if we can predict the appearance of passenger 2, it will be the opposite case. The global picking up distance will be reduced from 1200 m to 800 m. In the simulation, we will quantity this benefit. This also explains why we do not set the time scale of prediction too long. In case there is one predicted order, which is closer than any other order to the driver, the dispatch algorithm may keep the driver waiting until the

Table 1: Weather data used in the prediction model.

Weather	The highest temperature of the day	The lowest temperature of the day
Partly cloudy, cloudy, sunny, and little rainy	[9, 22]	[4, 13]

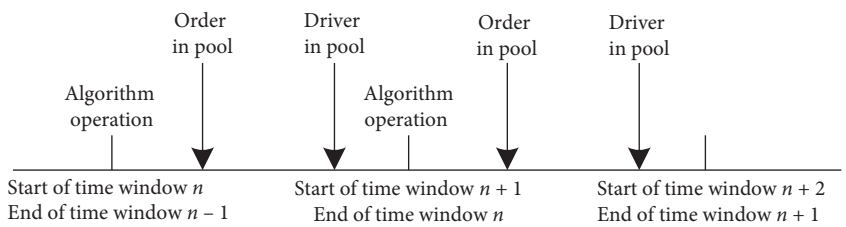


Figure 3: Time window division.

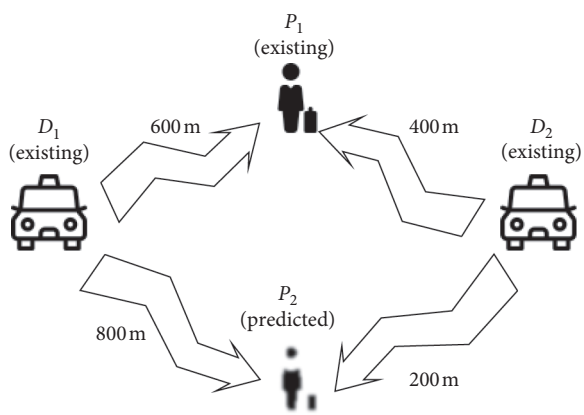


Figure 4: Sample of the benefit of prediction.

order appears. The longest possible waiting time is the time scale of prediction.

The target function of the optimization algorithm is

$$\min\left(f = \sum_i \sum_j S_{i,j} D_{i,j}\right), \tag{6}$$

subject to

$$\sum_j S_{i,j} \leq 1, \tag{7}$$

$$\sum_i S_{i,j} \leq 1, \tag{8}$$

where $S_{i,j}$ is the decision variable that decides whether driver i picks up passenger j or not; $D_{i,j}$ is the distance of driver i to pick up passenger j.

Constraint (7) aims to ensure that one driver can be maximally assigned with one order; constraint (8) aims to assure that one order can be maximally assigned with one driver. Here, we can choose to impose one more constraint:

$$\sum_i \sum_j S_{i,j} = \max(I, J). \tag{9}$$

This constraint can serve the purpose of trying best to satisfy orders in time window n with current available drivers. The difference is that, without the constraint, a part of drivers will not be dispatched to the order in the optimized solution because there may be an order that is much closer to him, while with the constraint, if there is no other candidate driver for the order, the driver will be dispatched in the current time window. In this study, we will also compare the performance of the algorithm with and without the constraint.

From the target function, we can observe that the problem is an ILP (integer linear programming) problem. The basic method to solve ILP is Simplex algorithm [32]. Its basic concept is to firstly construct an initial solution, which is a feasible and finite solution. If the initial solution is not the globally optimal one, then the algorithm will introduce nonbase variables to replace a base variable for a better solution. The iteration is repeated until the globally optimal one is found. Here, we will explain the whole process of solving the optimization algorithm (see Figure 5).

In the first step, we construct the distance matrix of each pair of driver and order. The driver list only contains the available drivers in the current time window n; the order list contains orders in both current time window n and future time window $n + 1$. The column of the matrix refers to the list of drivers and the row refers to the list of orders. The element $D_{i,j}$ means the distance between driver i and order j. Because we can only predict the number of orders in each cell, we lack information on the exact spatial distribution of orders in each cell. Therefore, it is hard to compute the exact distance between predicted orders and existing drivers. To solve this problem, we furtherly divide each cell of the grid into a 10×10 grid (see Figure 6).

Each cell of a larger grid can be described by a 10×10 grid. Then, we do statistics on the historical spatial distribution of orders in each 10×10 grid. Based on the statistic result, we can estimate where the orders may be located if there are orders predicted in the larger cell. We assume the location of predicted orders at the spatial center of the cell of 10×10 grid. After that, we can compute the distance between the driver and predicted orders.

The following part in the process optimization solution is more like a greedy algorithm, where we pick out the pair of order and driver in the ascending order of distance to construct an initial solution of matching. In the next step, we

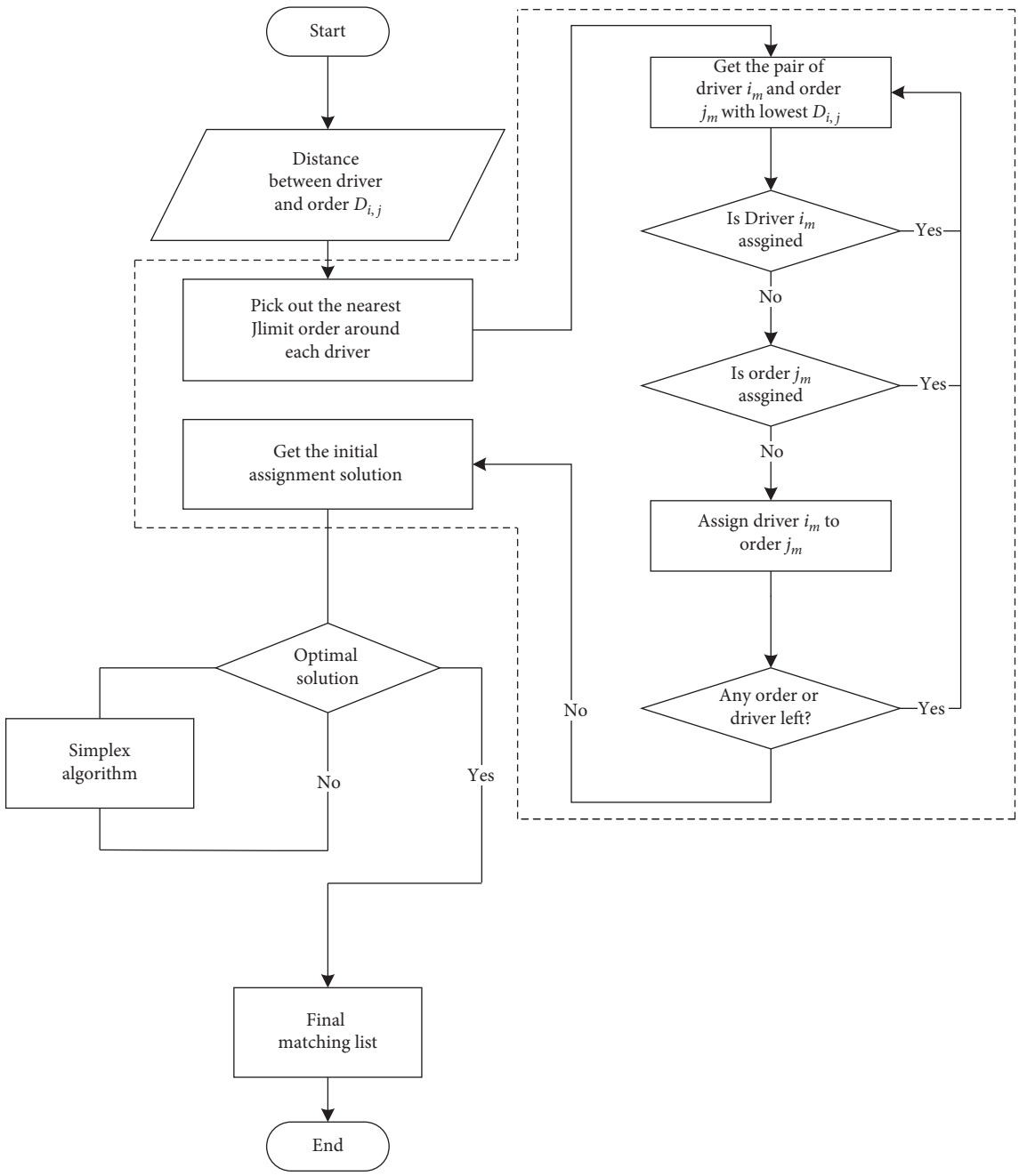


Figure 5: Flowchart of the optimization algorithm.

improve this solution by applying the simplex algorithm until we get the optimal solution. In our proposed method, one dispatch process would be finished within several seconds. This CPU compute time is efficient enough for the practical application.

3. Results and Discussion

3.1. Prediction Model Verification. An important parameter of ST-Resnet is the number of residual units used in the model. In the original paper, the author indicated that the more residual the units in the neural network, the deeper the neural network is, the accurate the prediction is. However, more residual units mean more memory usage during training and slower training. To find a balance between the accuracy and computation resources, we choose the number of residual units to be 21. There are a total of over 4.4 million trainable parameters in the model, which is quite a large quantity.

In the field of computer science, multiple forms of losses are used to evaluate the performance of the prediction model like *mse* (mean squared error) and *mae* (mean absolute error). However, these losses are usually used to compare the performance among different prediction models and hardly

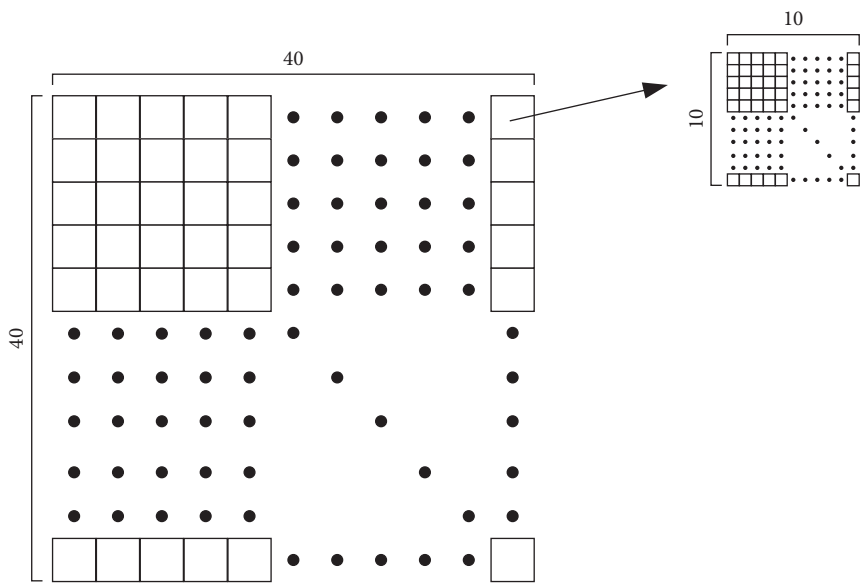


Figure 6: Cell division.

give a direct impression on the accuracy. In this study, we adopt the mape (mean absolute percentage error) as the metric of accuracy, which can directly show the differences.

After completing the training, we compute the mape of the test set to be 0.01169, which means that for each cell of the grid, the difference between prediction result and ground truth is about 1.169%. In addition, we also visualize the prediction result and the ground truth of two samples (see Figure 7).

The figures on the left side are the ground truth matrixes of the spatial distribution of order distribution; the figure in the middle is the corresponding prediction result; the figure on the right is the heatmap of the difference between ground truth and prediction result, which is computed by

$$A_{\text{diff}} = \left|A_{\text{GT}} - A_{\text{PR}}\right|, \tag{10}$$

where A_{diff} is the difference matrix; A_{GT} is the ground truth; A_{PR} is the prediction result.

We can see from the figure that there is little difference between the ground truth and prediction result. The prediction is relatively accurate and enough to put into the simulation.

3.2. Comparison of the Performance of Different Dispatch Strategies. Here, we will separately introduce the performance of different dispatch strategies. To provide a clearer pattern of performance comparison, we start from the difference between the greedy algorithm and pure optimization without prediction. We randomly selected one day in the dataset and operated the simulation. The performance of the two algorithms is shown in Table 2.

From Table 2, we can see that the greedy algorithm shows a poorer performance. Though the proportion of canceled order of greedy algorithm is not much different compared with the proposed methods, the average waiting time of passengers of the greedy algorithm (over 7 minutes) is twice as long as the proposed methods. This is mainly because the baseline algorithm does not provide priority to the orders that have been waiting long enough. In the real-world application, for commercial purposes, the ride-hailing dispatch platform may provide priority to the orders that have been waiting for a long time. Besides, when operating the dispatch, the Didi dispatch platform will provide each order to several candidate drivers to raise the chances that the order will be taken. Thus, the waiting time should be shorter in the real application. What is more, we did statistics on the void cruising distance proportion in the original record data. We found that the average value is 29.55%. This indicates a lever principle between the void cruising distance and the satisfaction of orders in former dispatch strategies. In the real application, the Didi company tried to assign the orders to more candidate drivers and caused more pick-up distance. In the simulation, the average waiting time of passengers in the optimization method is only 1/2 times of the greedy algorithm. This shows that the optimization algorithm can more easily and quickly answer the travel demand from customers. What is more, from the perspective of void cruising distance proportion, the optimization method shows a better performance, which is lesser than 1/2 times of the greedy algorithm. The overall result shows that the proposed algorithm surpasses the traditional greedy algorithm. We also operated a probability density statistic on the metrics of each order in the simulation (see Figure 8).

From left to right, the figures show the result of the separately greedy algorithm and optimization method without optional constraint and with optional constraint. The upper figures show the waiting time of passengers and the lower ones show the void cruising distance proportion. The x-axis is the value of the metric and the y-axis can be treated as the "probability." The larger the y value is, the higher probability is. The integration of the result of the greedy algorithm is small mainly because of the high

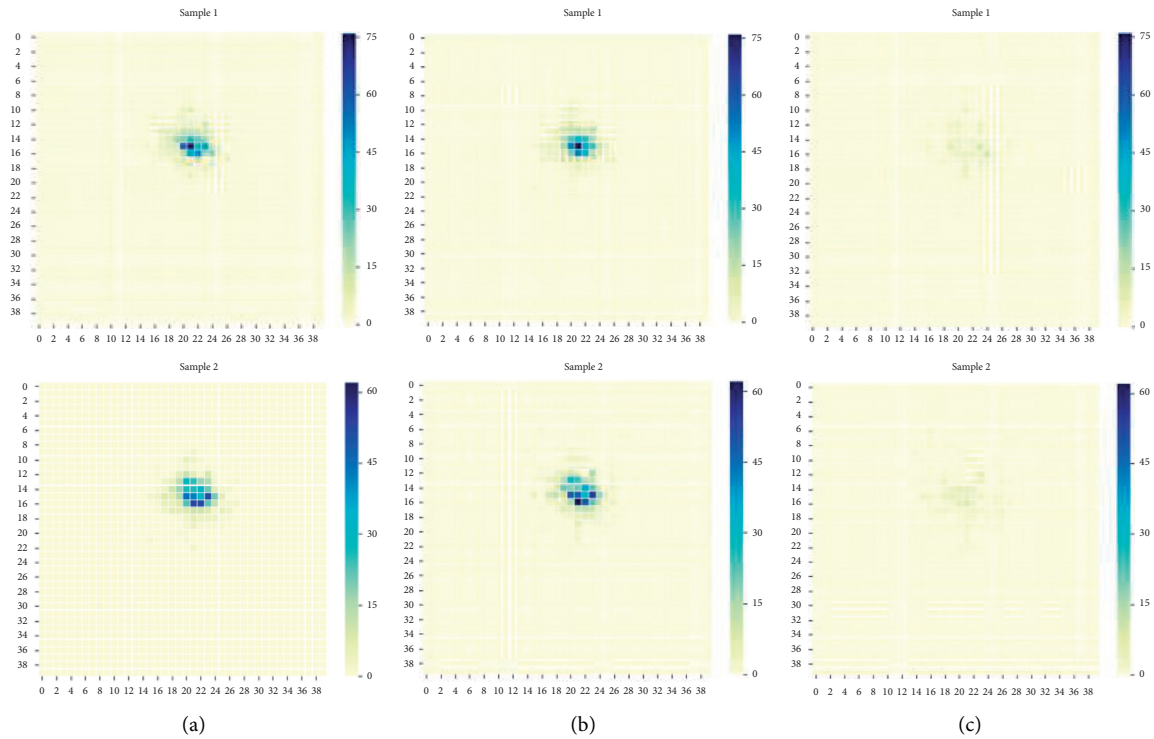


Figure 7: Visualization of prediction result. (a) Ground truth matrix. (b) Predicted matrix. (c) The difference between ground truth and prediction.

Table 2: Comparison of the greedy algorithm and proposed method by metric.

	Statistic	Simulation		
Metric	Original dataset	Baseline greedy algorithm	Proposed method without optional constraint (s)	Proposed method with optional constraint
AWT		466.15	219.23	217.02
AVCDP	29.55%	11.32	2.66	2.57
PCO	0.00%	1.61	0.00149	0.00

AWT: average waiting time of passenger; AVCDP: average void cruising distance proportion; PCO: proportion of canceled orders.

cancelation proportion. We can have a clear vision that the proposed algorithm can effectively suppress the waiting time and void distance proportion of most cases into a low interval. The waiting time in all of the cases is under 1000 seconds, which is about 16 minutes; the highest void cruising distance proportion is about 25%, while on the other hand, there are some extreme cases in the greedy algorithm. Some have been waiting for over 2500 seconds. In a small part of cases, the picking up distance is near half of the delivery distance. This furtherly proves the stability of the performance of the proposed algorithm.

If we do a transverse comparison between the optimization method with and without the optional constraint, we can find that the optimization algorithm with optional constraint performs better than that without constraint. A smaller proportion of canceled orders and shorter waiting times is natural. We also notice that it accidentally brings a lesser void cruising distance proportion. To better understand the mechanism behind this, we conduct an experiment under the perfect prediction, which means that there is no error in the prediction result. The result of metrics is separate, the average waiting time of passenger is 215.11 s; the average void cruising distance proportion is 2.50%; the proportion of canceled orders is 0.00%. What may cause the difference between the result of the optimization algorithm with and without optional constraint is probably the uncertainty of distance in the prediction. In the process of solving the optimization problem, we compute the distance between each order and driver in both the current and predicted time window and construct a distance matrix, because we lack the information of the exact spatial distribution of orders in each cell. This causes many uncertainties in the simulation and affects the performance of optimization. Since currently, most of the prediction of travel demand is mainly based on the area gridding method, we recommend more to pay main efforts on optimizing the current dispatch problem. The prediction result aims to provide guidance on which group of drivers are better choices of dispatching at present. We can also judge from the result that there is still a space of 0.07% of void cruising

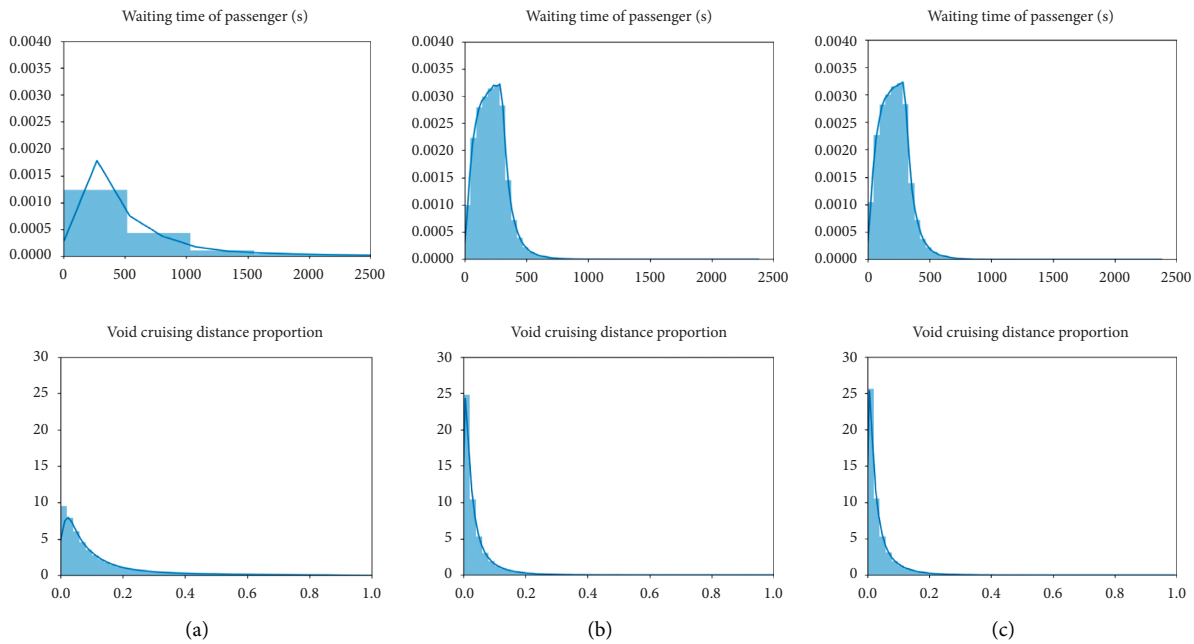


Figure 8: The probability density result of metrics. (a) Greedy algorithm. (b) Proposed method without optional constraint. (c) Proposed method with optional constraint.

distance proportion. This also indirectly proves the accuracy of the prediction model.

We further test the performance of different dispatch strategies in the region with high travel demand (Table 3). We operated a statistic on the metrics of the grids with more than 3000 orders. The area of the high travel demand region accounts for 7.38% of the total area, while the order's amount of the high travel demand region accounts for 96.68% of the total orders. The result in the high travel demand region is similar to the result of the whole research area.

In the next subsection, we will simulate the dataset on different days with different metadata to test the sensitivity of performance.

3.3. Comparison of Performance under Different Travel Distributions. As mentioned in the methodology part, we use the weather and mark of holiday and workday as the input of metadata. Therefore, these two factors can have their own impact on the spatial and temporal distribution of travel demand in the study area. In this section, we will discuss and analyze the sensitivity of performance.

The random day we have chosen in the previous subsection is November 25, which is a cloudy workday. We also choose data in the other three days for test and comparison; the attribute and simulation results are shown in Table 4.

In the table, there is only a little difference among the performances on different days, especially on workdays. All the values of metrics are nearly the same. We found a higher waiting time and void cruising distance proportion on weekend. We also operated simulation on other holidays (see extended Table 1 in Supplementary Materials) and concluded that it is not related to the holiday. Some holidays also show lower values of metrics. The overall performance of the method among different days is stable despite different daily conditions.

3.4. Temporal Analysis of Simulation Results. In this subsection, we will give an analysis of the temporal change pattern of void cruising distance. Generally speaking, the average void cruising distance becomes low when the number of orders rises because of high density (see extended Figure 1 in Supplementary Materials). We plot the temporal change of order numbers and reduced average void cruising distance compared with the greedy algorithm and original dataset (see Figure 9).

Figure 9 shows the temporal change of averagely reduced void cruising distance. The result shows that, in simulation, the averagely reduced void cruising distance shares a similar pattern with average void cruising distance. The averagely reduced void cruising distance remains at a low level when the quantity of orders is high and high when quantity is low. From 9 am to 9 pm, the ride-hailing activity becomes active suddenly. Orders appear in the study area with high density. During this period, the pick-up distance of each order can be reduced by about 1000 to 1500 m on average. From 3 am to about 7 am, the travels in ride-hailing are at a low level. The sparse travels in the urban area can cause high pick-up distance and thus cause a higher reduction. The effectiveness of the algorithm shows the best performance during this period. In Figures 9(c) and 9(d), the reduced void cruising distance is relatively stable across one day. The original dispatch contains many other factors that affect the performance; for example, drivers can choose whether to pick up or reject the assignment and the platform will provide the order for as many candidate drivers as possible to assure the

Table 3: Comparison of the greedy algorithm and proposed method by metric in high travel demand region.

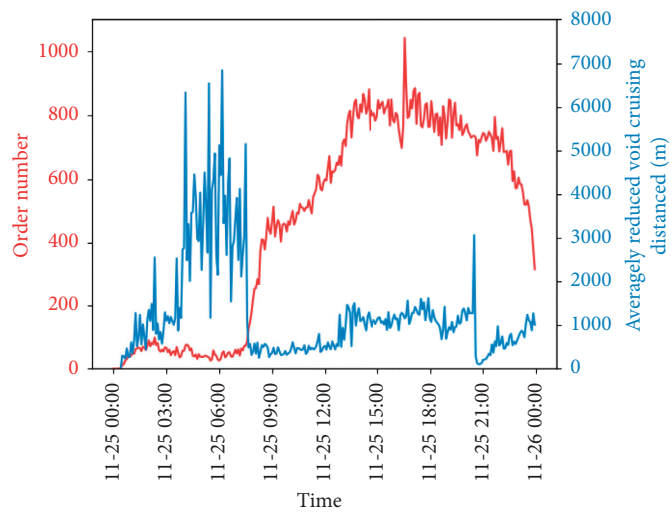
Statistic Metric	Original dataset (%)	Baseline greedy algorithm (s)	Proposed method with optional constraint (s)
AWT		447.47	215.41
AVCDP	29.34	11.24	3.73
PCO	0.00	1.64	0.00

AWT: average waiting time of passenger; AVCDP: average void cruising distance proportion; PCO: proportion of canceled orders.

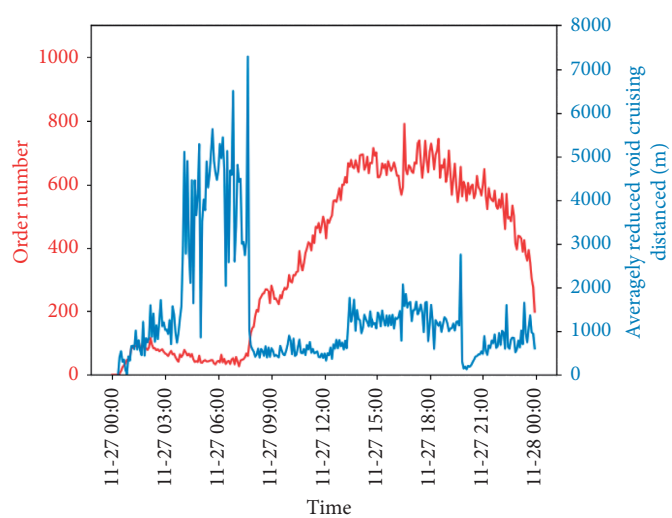
Table 4: The metadata and simulation result on different days.

Date attribute Weather	November 14 Sunny	November 16 Rainy	November 25 Cloudy	November 27 Cloudy
Holiday or workday	Workday	Workday	Workday	Holiday
AWT	217.42 s	217.64 s	217.02 s	225.23 s
AVCDP	2.56%	2.54%	2.57%	2.77%
PCO	0.00%	0.00%	0.00%	0.00%

AWT: average waiting time of passenger; AVCDP: average void cruising distance proportion; PCO: proportion of canceled orders.



(a)



(b)

Figure 9: Continued.

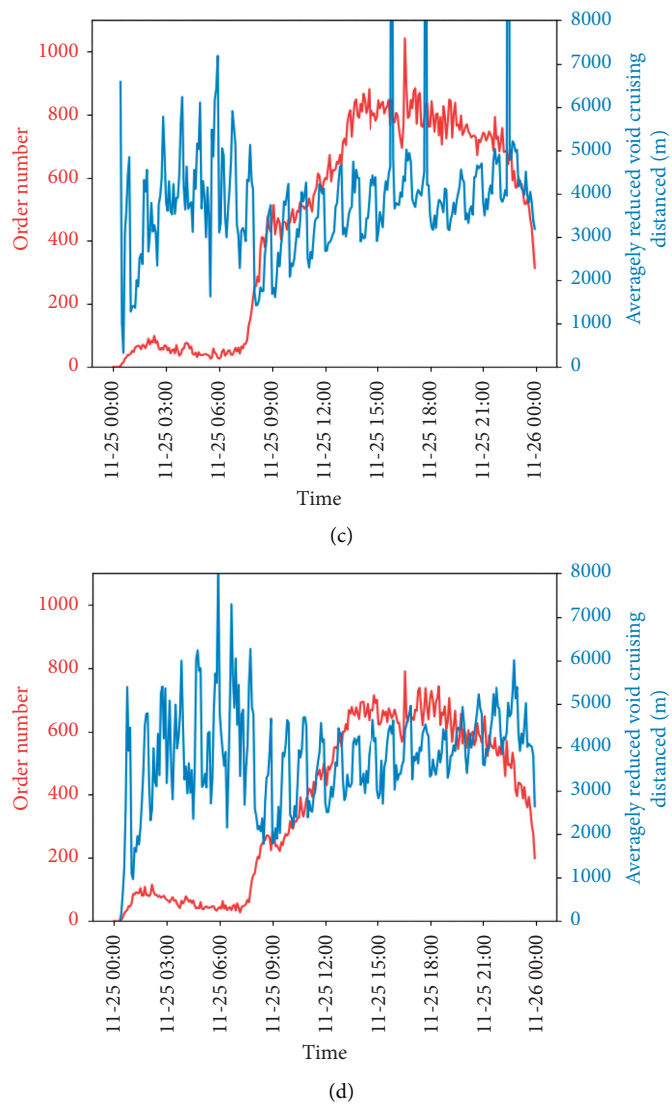


Figure 9: The temporal change pattern of the order number and averagely reduced void cruising distance compared with (a) greedy algorithm on November 25; (b) greedy algorithm on November 25; (c) original dispatch on 25; (d) original dispatch on 27.

acceptance. We can conclude that there is a lot of potential in the reduction of void cruising distance in the dispatch process. The platform can try to propose some policies to maximally ensure the optimal assignment if necessary.

4. Conclusions

Under the background of energy-saving and emission reduction, many studies focus on traffic emissions [33]. It is pointed out that void cruising for the next passenger caused a lot of unnecessary exhaust emission [34].

The main idea of this study is to propose a dispatch method based on both prediction and optimization methods to improve the efficiency of the ride-hailing system. We use the same dataset for simulation. We firstly preprocess the data into the desired input format of historical observation needed by the prediction model and collect the metadata for additional input. Next, we adopt the ST-Resnet as the deep-learning neural network for prediction and successfully train the prediction model. The rescaled MAPE is rather little and enough for simulation of the dispatch algorithm. Then, we introduce the dispatch algorithm based on the optimization method. We state the target function to optimize and the constraint including the optional one which can impose a full satisfaction of orders. We use the greedy algorithm which is used widely in the current real-time dispatch system as the baseline and compare the performances. We find that the proposed method outperformances the baseline and shows a good stability of performance on the evaluation metrics, which proves a great potential in real-time application. In addition, we also find that the algorithm shows better performance with the optional constraint. This is mainly because of the uncertainties in the location of predicted orders. Thus, we suggest imposing the constraint that maximizes the number of served orders in solving the current dispatch problem. By far we have successfully answered and filled research gaps mentioned in the literature review.

There are certainly some limitations in this work. For example, we mainly use the statistical result to estimate the location of predicted orders in the future. Adopting a smaller cell can help get a more accurate location. However, it will inevitably make the spatial distribution matrix sparser, thus making the model hard to be trained. In the future, if a better prediction method like GCN (graphical convolutional network) can be developed better, we can improve this limitation by adopting them.

Data Availability

The Didi ride-hailing record data used to support the findings of this study have been deposited in the GAIA Open Dataset repository https://outreach.didichuxing.com/research/opendata/en/#.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Supplementary Materials

Extended Table 1: simulation result on holidays. Extended Figure 1: temporal change of order number and average void cruising distance. (a) November 25; (b) November 27. (*Supplementary Materials*)

References

[1] P. S. Lavieri and C. R. Bhat, "Investigating objective and subjective factors influencing the adoption, frequency, and characteristics of ride-hailing trips," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 100–125, 2019.

[2] A. Tirachini, "Ride-hailing, travel behaviour and sustainable mobility: an international review," *Transportation*, vol. 47, pp. 1–37, 2019.

[3] B.-J. Tang, X.-Y. Li, B. Yu, and Y.-M. Wei, "How app-based ride-hailing services influence travel behavior: an empirical study from China," *International Journal of Sustainable Transportation*, vol. 14, pp. 1–15, 2019.

[4] Z. Li, Y. Hong, and Z. Zhang, "An empirical analysis of on-demand ride sharing and traffic congestion," in *Proceedings of the International Conference on Information Systems*, Dublin, Ireland, 2016.

[5] Y. Sui, H. Zhang, X. Song et al., "GPS data in urban online ride-hailing: a comparative analysis on fuel consumption and emissions," *Journal of Cleaner Production*, vol. 227, pp. 495–505, 2019.

[6] J. Bischoff, I. Kaddoura, M. Maciejewski, and K. Nagel, "Simulation-based optimization of service areas for pooled ride-hailing operators," *Procedia Computer Science*, vol. 130, pp. 816–823, 2018.

[7] G. Feng, G. Kong, and Z. Wang, "We are on the way: analysis of on-demand ride-hailing systems," Research Paper, 2017.

[8] F. Calderón and E. J. Miller, "Modelling within-day ride-hailing service provision with limited data," *Transportmetrica B: Transport Dynamics*, vol. 67, no. 8, pp. 1–24, 2020.

[9] N. Korolko, D. Woodard, C. Yan, and H. Zhu, "Dynamic pricing and matching in ride-hailing platforms," Research Paper, 2018.

[10] P. Afeche, Z. Liu, and C. Maglaras, "Ride-hailing networks with strategic drivers: the impact of platform control capabilities on performance," pp. 18-19, 2018, Columbia Business School Research Paper.

[11] H. Zhang, X. Song, T. Xia et al., "Battery electric vehicles in Japan: human mobile behavior based adoption potential analysis and policy target response," *Applied Energy*, vol. 220, pp. 527–535, 2018.

[12] H. Zhang, X. Song, Y. Long et al., "Mobile phone GPS data in urban bicycle-sharing: layout optimization and emissions reduction analysis," *Applied Energy*, vol. 242, pp. 138–147, 2019.

[13] T. Sühr, A. J. Biega, M. Zehlike, K. P. Gummadi, and A. Chakraborty, "Two-sided fairness for repeated matchings in two-sided markets: a case study of a ride-hailing platform," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3082–3092, Anchorage, AK, USA, 2019.

[14] C. Yan, H. Zhu, N. Korolko, and D. Woodard, "Dynamic pricing and matching in ride-hailing platforms," *Naval Research Logistics (NRL)*, vol. 67, 2019.

[15] D. M. R. Institute, "Report on big data of intellegent travel in chengdu," 2016.

[16] L. Zheng, L. Chen, and J. Ye, "Order dispatch in price-aware ridesharing," *Proceedings of the VLDB Endowment*, vol. 11, no. 8, pp. 853–865, 2018.

[17] M. M. Vazifeh, P. Santi, G. Resta, S. H. Strogatz, and C. Ratti, "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, no. 7706, p. 534, 2018.

[18] A. M. Arslan, N. Agatz, L. Kroon, and R. Zuidwijk, "Crowdsourced delivery—a dynamic pickup and delivery problem with ad hoc drivers," *Transportation Science*, vol. 53, pp. 222–235, 2018.

[19] T. ICfEa, "Online ride-hailing network and fuel consumption: a driver's perspective," *Innovation Center for Energy and Transportation*, vol. 20, 2017.

[20] K. Shi, B. Di, K. Zhang, C. Feng, and L. Svirchev, "Detrended cross-correlation analysis of urban traffic congestion and NO_2 concentrations in Chengdu," *Transportation Research Part D: Transport and Environment*, vol. 61, pp. 165–173, 2018.

[21] D. Wang, Y. Yang, and S. Ning, "Deepstcl: a deep spatio-temporal convlstm for travel demand prediction," in *Proceedings of the 2018 international Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, Rio de Janeiro, Brazil, 2018.

[22] H. Yao, F. Wu, J. Ke et al., "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, Toronto, Canada, 2018.

[23] K. F. Chu, A. Y. Lam, and V. O. Li, "Travel demand prediction using deep multi-scale convolutional LSTM network," in *Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1402–1407, IEEE, Maui, HI, USA, 2018.

[24] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, CA, USA, 2017.

[25] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[26] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, http://arxiv.org/abs/14126980.

[27] M. Cassel and F. Lima, "Evaluating one-hot encoding finite state machines for SEU reliability in SRAM-based FPGAs," in *Proceedings of the 12th IEEE International On-Line Testing Symposium (IOLTS'06)*, p. 6, IEEE, Lake of Como, Italy, 2006.

[28] L. Al-Kanj, J. Nascimento, and W. B. Powell, "Approximate dynamic programming for planning a ride-hailing system using autonomous fleets of electric vehicles," *European Journal of Operational Research*, vol. 284, no. 3, pp. 1088–1106, 2020.

[29] X. Yu and S. Shen, "An integrated decomposition and approximate dynamic programming approach for on-demand ride pooling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, 2019.

[30] Y. Khazbak, J. Fan, S. Zhu, and G. Cao, "Preserving location privacy in ride-hailing service," in *Proceedings of the 2018 IEEE Conference on Communications and Network Security (CNS)*, pp. 1–9, IEEE, Beijing, China, 2018.

[31] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: a queueing-theoretical perspective," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 186–203, 2016.

[32] V. Klee and G. J. Minty, "How good is the simplex algorithm," *Inequalities*, vol. 3, pp. 159–175, 1972.

[33] Z. Huang, F. Cao, C. Jin, Z. Yu, and R. Huang, "Carbon emission flow from self-driving tours and its spatial relationship with scenic spots - a traffic-related big data method," *Journal of Cleaner Production*, vol. 142, pp. 946–955, 2017.

[34] X. Luo, L. Dong, Y. Dou et al., "Analysis on spatial-temporal features of taxis' emissions from big data informed travel patterns: a case of Shanghai, China," *Journal of Cleaner Production*, vol. 142, pp. 926–935, 2017.