Taxi-Cruising Recommendation via Real-Time Information and Historical Trajectory Data

Tong Wang, *Member, IEEE*, Zhaoxian Shen, Yue Cao, *Member, IEEE*, Xiujuan Xu, and Huiwen Gong

***Abstract*— With the development of GPS technology, location-based information services are becoming more and more diverse. Using the trajectory data generated by GPS can analyze and study the various needs of taxi drivers. Big data technology such as interpreting, manipulating data and extracting nugget of information from data is crucial in Intelligent Transportation System (ITS). In order to enhance cruising efficiency of drivers, this paper proposes a Taxi-cruising Recommendation strategy based on Real-time information and Historical Trajectory data (TR-RHT). Primarily, we construct a Passenger-Demand predict model based on Historical Hotspot (PDHH) to predict the passengers' demand in hotspot area. Then, an Improved Decision Tree (IDT) predicting algorithm is proposed to construct spatio-temporal index to select suitable historical data. Furthermore, we introduce Hotspot Recommendation based on Historical Trajectory (HRHT), wherein it defines cruising event as the process of taxi searching for passengers. The HRHT model performs statistics and analysis on the different states of taxi operation, which can obtain the probability of catching passengers at each hotspot and calculate the travel time between hotspots. The model selects the statistics result of cruising efficiency and driving time between hotspots based on spatio-temporal index, then analyzes the selected result to provide optimal pick-up hotspots for drivers. Experiment results show that TR-RHT can precisely suggest a cruising path to reduce cruising time for drivers.**

***Index Terms*— Taxi trajectory, data analysis, hotspot, recommendation system.**

I. Introduction

In urban city, taxis equipped with GPS generate a great deal of data including taxi location, recording time, taxi state, data validity, etc. The huge amount of data reveals a lot of valuable information such as travel regularities in passengers, cruising regularities and distribution of traffic flow [1]–[3]. With the application of C-RAN and D2D technology [4], [5] in mobile networks, the delay of data collection in smart city management is greatly reduced. However, the matching time among availability of taxi drivers and passengers' demand does not decrease, even with the increase of public vehicles. Big data technology such as interpreting, manipulating data and extracting key information from data has become a key problem facing Intelligent Transportation System (ITS) [6]–[8]. Through the statistical analysis of taxi trajectory data in Beijing, we find average daily travel distance of taxis is 400km and average cruising time of taxis is 8 minutes. It accounts for 40 percent of the time that taxi driver spends on each passenger. In the light of this, it's necessary to recommend potential hotspots [9] to reduce fuel consumption and cruising time of taxis.

In spite of numerous literatures on analyzing historical trajectory data, Kong *et al.* [10]–[12] only consider the division of taxi historical trajectory data according to the distinction between working days and rest days. Since selecting different data in different time range will affect the performance of Taxi-cruising recommendation (actively looking for passengers), the classification of trajectory data is important. Some papers [13], [14] use neural networks to predict passenger demand. Moreira-Matias *et al.* [15] introduce a novel methodology for predicting the spatial distribution of taxi passengers for a short-term time horizon using streaming data. Reference [16] proposes an energy-efficient system based on deep convolutional neural networks (CNN) for detection. Furthermore, even if a simple classification method is adopted to classify the historical data, the classification results are still static. Therefore, in order to select the data suitable for recommendation, the optimal method is to dynamically select appropriate data according to real-time traffic information. To tackle the above problems, we propose a Taxi-cruising Recommendation strategy based on Real-time information and Historical Trajectory data (TR-RHT).

First of all, different from previous works on route recommendation [1], [10], [15], [17], TR-RHT not only recommends the actual driving route to pick up passengers, but also forecasts the passengers' demand. Secondly, TR-RHT considers real-time traffic (e.g., the real-time passengers' demand information), to select optimal hotspots for taxi to cruise passengers (that is which hotspots to recommend for the taxi driver). We evaluate the performance of TR-RHT strategy based on historical GPS trajectory of taxis in Beijing, China, the trajectory includes one-month trajectory of 4000 taxis in Beijing [18]. Results show that TR-RHT can greatly reduce the average cruising time of taxi drivers. The major contributions of this paper are as follows:

1) We consider that mobility of taxis and passengers will suffer from delay, when taxi travels towards hotspot to pick up passenger. Such delay may cause inaccu-

Manuscript received 28 January 2020; revised 11 June 2020 and 14 February 2021; accepted 2 June 2021. Date of publication 16 July 2021; date of current version 2 August 2023. This work was supported by the Fundamental Research Funds for the Central Universities under Grant 3072021CF0813. The Associate Editor for this article was S. Mumtaz. *(Corresponding authors: Yue Cao; Xiujuan Xu.)*

Tong Wang, Zhaoxian Shen, and Huiwen Gong are with the College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China (e-mail: wangtong@hrbeu.edu.cn; shenzhaoxian@hrbeu.edu.cn; gonghuiwen@hrbeu.edu.cn).

Yue Cao is with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China, and also with the Shenzhen Key Laboratory of Data Visualization (Smart City), Research Institute, Beihang University, Shenzhen, Shenzhen 518063, China (e-mail: yue.cao@whu.edu.cn).

Xiujuan Xu is with the School of Software, Dalian University of Technology, Dalian 116620, China (e-mail: xjxu@dlut.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3093207

rate recommendation decision. Here, we introduce a model named Passenger-Demand predict model based on Historical Hotspot (PDHH), to predict the demand of passengers in hotspot based on machine learning algorithm.

2) We consider that heterogeneity of historical trajectory data will have an impact on the accuracy of recommendation strategy. Here, TR-RHT dynamically selects the trajectory data which matches passengers' demand, as an input of PDHH model for recommendation decision making.
3) We construct the Hotspot Recommendation based on Historical Trajectory (HRHT) to compute the successful probability and time of cruising event.

II. Related Work

In recent years, there are many analysis methods and data handling methods on hotspot selection, passenger cruising efficiency (cruising time, cruising probability and income, etc.) and driving route planning. Literature generally considers two aspects of recommendation (taxi-cruising recommendation) strategies for drivers, which are individual recommendation and multi taxi recommendation. Here, individual recommendation strategy aims to maximize the cruising efficiency of a single taxi, multi-taxi recommendation strategy tackles the assignment of multiple taxis within the same area to cruising passengers.

A. Recommendation for Multi Taxi Drivers

In order to achieve a fair route assignment for multiple taxis, Qian *et al.* [1] propose a Sharing Considered Route Assignment Mechanism (SCRAM) and define three evaluation principles: priority principle, decaying principle and sharing principle. SCRAM considers the Expected Driving Cost (EDC) function to evaluate sharing routes. However, this work does not discuss the cruising strategy in combination with passenger hotspots. With the increase in the number of routes, SCRAM suffers from huge computation overhead. Besides, Lei *et al.* [19] propose the cooperation approach via strategic concession game, trying to minimize the individual and total travel time.

B. Recommendation for Individual Taxi

To recommend cruising trajectory or hotspots, it is important to extract hotspots and analyze cruising efficiency of trajectory or hotspots. Normally, the cruising efficiency includes cruising probability, cruising time and revenue.

1) Recommendation for Top-N Area Hotspots: Kong *et al.* [10] and Yang *et al.* [11] propose the Time-Location-Relationship (TLR) model to predict the distribution of passengers in functional region and recommend Top-N area to driver. However, similar to analysis of cruising probability, they recommend hotspots (functional regions) only according to the number of passengers demand, without consider factors as cruising distance, cruising time and revenue.

2) Recommendation for High Cruising Probability Area: Ge *et al.* [20] introduce Longest Common Prefix (LCP) algorithm. The algorithm analyzes cruising probability and time of each pickup point to select optimal driving route. Qian *et al.* [17] is to make a detailed analysis of routes, including cruising probability, income and cruising time. It also introduces Markov Decision Process (MDP) to compute optimal route. Hu *et al.* [21] construct a route recommendation system, in which passengers' hotspots are extracted and the heuristic algorithm is used to construct the hotspots tree. Hotspots tree takes the current location as the root node and connects the locations of all passengers. Correspondingly, it proposes a probabilistic model for estimating the weight of each route. It gives a set of weighted cycle recommendation methods. A route recommendation (recommend routes to hotspot) algorithm based on real-time passengers' demand distribution is mainly studied by Hu [22], and it is recommended for vacant taxi drivers to select routes with short travel time and high probability of carrying passengers. Yang *et al.* [23] propose two route recommendation algorithms: Shortest Expected Cruise Route (SECR) and Adaptive Shortest Expected Cruise Route (ASECR) algorithm. The ASECR algorithm is an extension of SECR, which constantly updates routes and dynamically updates the temporal probability of network.

3) Recommendation for High Revenue Route: From the perspective of optimization on taxi income, previous work [2] analyzes the drivers who have high revenue efficiency in cruising customers. The Markov Decision Process (MDP) is applied to simulate the process of taxi cruising passengers. However, this article does not consider and cluster of passenger hotspots instead of utilize grid. From the comprehensive perspective of drivers and passengers, Yuan *et al.* [24] propose a recommendation system for taxi drivers and passengers. The recommendation system provides drivers with locations and routes where they are more likely to pick up passengers quickly (on the route or in these locations). The system provides drivers with the location of high probability to wait for taxis. However, this work does not introduce real-time information in online recommendation phase.

4) Recommendation for Other Aspects: Different from analyzing cruising efficiency of cruising trajectory or hotspots, Zhang *et al.* [25] propose a recommendation system based on spatio-temporal clustering for taxi drivers to provide nearby pickup points. The real-time recommendation of that paper is to search suitable pick-up place using online map query method. It is an effective solution to the problem of real-time performance on the road. Also, Huang *et al.* [26] propose Time-Dependent Vehicle Routing Problem with Path Flexibility (TDVRP-PF) model to solve traffic congestion problem. Shen *et al.* [27] propose an improved DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm to cluster hotspot. This work introduces trajectory clustering algorithm based on hotspots to recommend routes by analyzing short-term GPS sensor data. Dai *et al.* [12] recommend personalized route for individual taxis. In order to dynamically select subsets of trajectories instead of simply distinguish working days and weekdays, this work introduces

TABLE I
SYMBOL LIST

Symbol	Explanation
$point_{ij}$	Taxi GPS log
$PickupPoint$	Pickup point
$Hotspots$	A collection of passenger hotspots
h_i	The i^{th} hotspot
$P(h_i, d_j)$	Passengers' demand in 24 hours of a day
$M(h_i)$	Passenger's demand in all days
$Train$	Train data of PDHH model
$labels$	Labels of PDHH model
$point_s$	Starting position of taxi
$point_d$	Destination of taxi
t_k	The k_{th} time interval of a day
n_{ik}	The number of passengers
m_{jk}	The number of taxis
$probability_{jk}$	Cruising probability of taxi
r_{jk}	Average revenue of hotspots

real-time traffic information and proposes predicting model based on historical data. This work also introduces the online and offline processes to reduce the computing time required for the recommendation phase.

III. SYSTEM MODEL

In this section, we analyze trajectory data to tackle the three problems as follows: (1) predicting passengers' demand in hotspot area; (2) calculating expected cruising time of taxi travel to destination in hotspot area; (3) evaluating cruising probability of taxi travel to destination in hotspot area.

Primarily, we list symbols to describe commonly used variables as shown in Table I. Secondly, basic definitions are given below:

A. Basic Definitions

Each GPS point is recorded as a strip of trajectory data. We give detailed definitions as below:

Definition 1 (GPS Log): A strip of GPS data records taxi ID, time, longitude, latitude and business state. That is:

$$point_{i,j} = \{taxi_i, time_j, longitude_{i,j}, latitude_{i,j}, state_{i,j}\} \tag{1}$$

where $State_{i,j}$ indicates whether $taxi_i$ has found a passenger. $State_{i,j} = 1$ means the taxi is carrying a passenger to the destination. $State_{i,j} = 0$ means the taxi is carrying next passenger.

Definition 2 (Pickup Point): Pickup point denotes the position that a taxi driver finds a passenger. That is:

$$PickupPoint_{ij} = \{taxi_i, time_j, longitude_{ij}, latitude_{ij}\} \tag{2}$$

Definition 3 (Hotspots): It is defined as cluster centers of pickup points defined in Definition 2. That is:

$$Hotspots = \{h_1, h_2, \ldots, h_n\} \tag{3}$$

where h_i is defined as $\{longitude_i, latitudei\}$.

Definition 4 (Passengers' Demand): Given hotspot h_i, we filter the pickup point in h_i of one day. Further, we capture the passenger's demand within 24 hours of one day at the pickup point. Therefore, we can get a vector where its dimension is 24 to indicate the passenger demand in h_i. That is:

$$P(h_i, d_j) = \{p_{ij}^1, p_{ij}^2, \ldots, p_{ij}^k\}^{\mathrm{T}} \tag{4}$$

where d_j indicates the date of this vector. P_{ij}^k is the number of passengers at the k^{th} hour in d_j.

Definition 5 (Passenger Matrix): We collect all passengers' demand information of h_i. That is:

$$\begin{aligned} M(h_i) &= \{P(h_i, d_1), P(h_i, d_2) \ldots, P(h_i, d_n)\} \\ &= \begin{bmatrix} p_{i1}^1, & p_{i1}^2, & \ldots, & p_{i1}^k \\ p_{i2}^1, & p_{i2}^2, & \ldots, & p_{i2}^k \\ \ldots & & & \\ p_{in}^1, & p_{in}^2, & \ldots, & p_{in}^k \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{5}$$

Definition 6 (Taxis Travel Event): Taxis travel event refers to the driving process of a taxi from starting position $point_s$ to destination $point_d$. That is:

$$\begin{aligned} trip(point_s, point_d) = \{&time_s, time_d, location_s, \\ &location_d, state_{sd}\} \end{aligned} \tag{6}$$

where $time_s$ and $time_d$ are the starting time and ending time of the driving event, $location_s$ and $location_d$ are the starting and destination position, $state_{sd}$ represents the cruising state of taxis during the event. If $state_{sd}$ is 0, it represents the taxi is occupied. If not, it means the taxi is unoccupied. If $point_s$ is the passenger pickup point and $point_d$ is the passenger drop-off point, the event means passenger-carrying event. If P_s is the passenger drop-off point, P_d is the passenger pickup point, the event means passenger-seeking event.

Definition 7 (Hotspot Event): In order to measure the cruising efficiency of each hotspot, it is required to map the location of starting point and destination point of driving event into its located hotspot area h_s and h_d. That is:

$$\begin{aligned} route(h_s, h_d) = \{&time_s, time_d, location_s, location_d, \\ &state_{sd}\} \end{aligned} \tag{7}$$

Definition 8 (Minimum Driving Time Between Two Hotspots): Given as the minimum driving time between hotspot h_s and hotspot h_d in time period t_k. That is:

$$t(h_i, h_j, t_k) = t_{ijk}^{driving} \tag{8}$$

Definition 9 (Driver's Revenue): For the convenience of expression, this paper assumes that driver's revenue depends on driver's driving distance. Thus, driving distance can be expressed by driving velocity and driving time. That is:

$$r = time \times v \times price \tag{9}$$

where $time$ is the driving time of a taxi in the process of carrying passenger, $price$ is the fare for taxi driving distance per kilometre, v is the driving velocity of taxi. Since the driving velocity of the taxi is not marked in historical

data, we assume that driving velocity is constant. Therefore, the income of the taxi driver depends on his passenger mileage.

Definition 10 (The Cruising Efficiency of Hotspot): The cruising efficiency of a hotspot consists of the average cruising time at a hotspot, the number of passengers in the hotspot, the number of taxis in the hotspot and the average cruising revenue of the hotspot. That is:

$$E(h_j, t_k) = \{t_{jk}^{waiting}, n_{jk}, m_{jk}, r_{jk}\} \tag{10}$$

where h_j represents the j^{th} hotspot, t_k is the k^{th} time interval of the day. $t_{jk}^{waiting}$ expresses the average time taxis spent on seeking passenger at hotspot h_i in the time interval t_k. Meanwhile, n_{jk} m_{jk} and r_{jk} represent the number of passengers appeared, the number of vacant taxis appeared and drivers' average revenue in the time interval at hotspot h_j, respectively.

Definition 11 (Recommended Hotspot Performance): The recommended hotspot performance consists of the time required for a taxi to travel to a hotspot, the cruising probability of the recommended hotspot and the revenue of carrying passengers. That is:

$$performance(h_i, h_j, t_k) = \{t_{ijk}^{cruising}, probability_{jk}, r_{jk}\} \tag{11}$$

where $t_{ijk}^{cruising}$ indicates the cruising time required for a taxi to find a passenger at hotspot h_j from the area h_i where the starting point is located, $probability_{jk}$ means the cruising probability at hotspot h_j, r_{jk} is the revenue of carrying passengers at hotspot h_j. That is:

$$t_{ijk}^{cruising} = t_{ijk}^{driving} + t_{jk}^{waiting} \tag{12}$$

$$probability_{jk} = \frac{n_{jk}}{m_{jk}} \tag{13}$$

$$r_{jk} = \alpha * t_{jk}^{taking} \tag{14}$$

where t_{jk}^{taking} is the driving time of a taxi in the process of carrying passenger, α is a constant and represents the coefficient.

B. Problem Definitions

To solve the problem of hotspot recommendation, TR-RHT needs to calculate the probability and expected time of cruising a passenger on a given path. Problem definitions are given below.

Problem Definition 1 (Prediction of Passengers' Demand): In order to dynamically select the historical data in real-time and recommend the selected historical data, the passengers' demand is predicted based on the offline historical data and online taxis passenger-carrying information.

Problem Definition 2 (Probability of Cruising a Passenger in Hotspot h_i): For a given hotspot h_i, we need to estimate the success probability $P(h_i)$ of getting a passenger at a certain time.

Problem Definition 3 (Cruising Time in Hotspot h_i): For a given hotspot h_i, it is necessary to estimate the cruising time of finding a passenger at a certain time.

Problem Definition 4 (Revenue in Hotspot h_i): For a given hotspot h_i, we need to calculate the average revenue in the future.

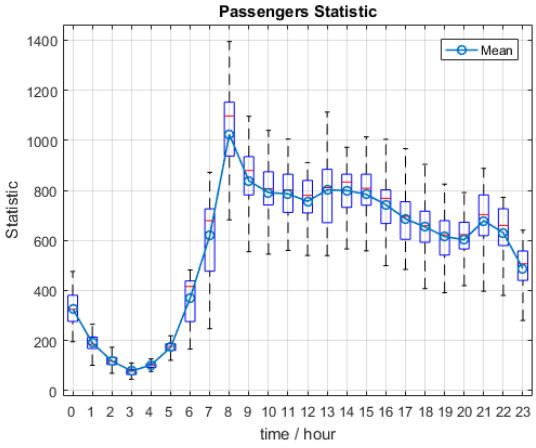


Fig. 1. Distribution of capacity of passengers based on every hour during one month data.

C. System Model

In this part, we introduce details of our proposed PDHH model. Fig. 1 shows the statistic of hotspot demand in one month according to each hour. The x-axis coordinates from 0 to 23 represent each hour. The y-axis represents the number of passengers. For a fixed value on x-axis, there are 5 values correspond to y-axis. They are maximum value, Q3 (3st quartile), average, Q1 (1st quartile), minimum value from top to bottom respectively. From Fig. 1, we find that the capacity of hotspot changes regularly as time goes on in one day. The number of passengers decreases gradually between 2 a.m. and 4 a.m. in the morning because most people are resting. The number of passengers increases sharply from 5 a.m. to 9 a.m. in the morning because most people are commuting.

We also find that the distribution of the number of passengers has a striking difference in one month. For example, the maximum value at 8 a.m. is 1395 and the minimum value is 538. Therefore, it is inaccurate to simply utilize the average passengers' demand to express the hot spot heat. In addition, we also find that if there are more passengers between 10 a.m. and 12 a.m., the number of passengers from 1 p.m. to 3 p.m. may be relatively high in the afternoon.

Fig. 2 illustrates the statistic of passengers in one-month data. The x-axis coordinates from 1 to 31 represent one-month data. The y-axis represents the number of passengers. From Fig. 2, we find the curve changes periodically. The change period of curve is roughly one week. Therefore, we consider that travelling is related to weather, air index, weekdays, weekends and so on. These factors are included in historical trajectory, although they don't record in historical data. Further, our prediction model is valuable. We can find the lines are basically coincident between 2 a.m. and 4 a.m. in the morning. During that time, passengers' travel usually cannot be affected by weather or other factors. For example,if someone catch the train in the morning, his travel will not be restricted by above factors. Thus, it is meaningless to find similar days that own same factors by predicting in that day.

As mentioned in Definition 5, we divide the pickup point data into 24 parts based on 24 hours in each day. Therefore,

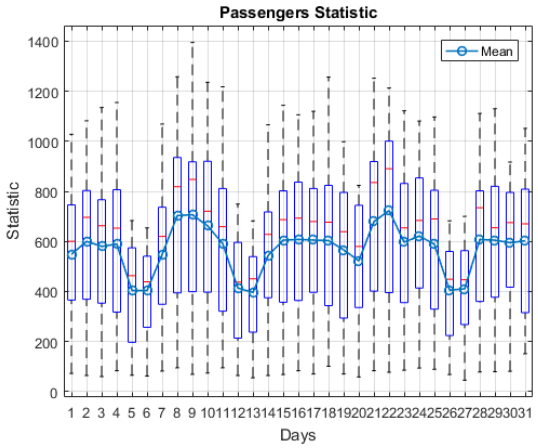


Fig. 2. Distribution of capacity of passengers based on every day.

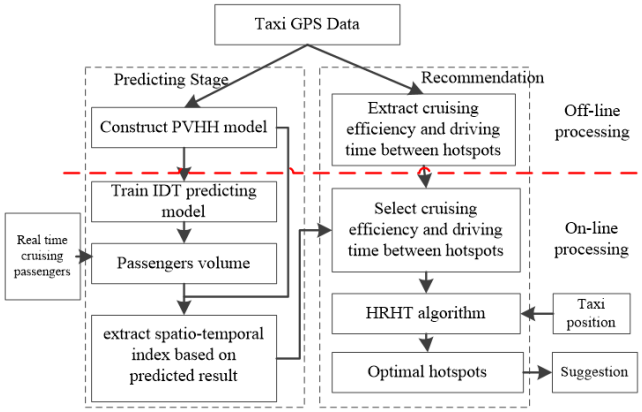


Fig. 3. Overview of TR-RHT strategy.

we can get a matrix $M(h_i)$ which contains hotspot information. We predict passengers' demand in $hotspot_i$ based on matrix $M(h_i)$. In this method, we can predict the passengers' demand without considering the impact of weekdays and weekends. The goal of PDHH model is to find a historical day that is similar to today.

Suppose that we need to predict passengers demand at m o'clock by utilizing the past k hours data, we extract the $m-k$ columns to $m-1$ columns data in $M(h_i)$ as features, and m columns data as labels. That is:

$$original\ featcures\ (h_i, m) = \begin{bmatrix} p_{i1}^{m-k}, & p_{i1}^{m-k+1}, & \ldots, & p_{i1}^{m-1} \\ p_{i2}^{m-k}, & p_{i2}^{m-k+1}, & \ldots, & p_{i2}^{m-1} \\ \ldots & & & \\ p_{in}^{m-k}, & p_{in}^{m-k+1}, & \ldots, & p_{in}^{m-1} \end{bmatrix}^{\mathrm{T}} \tag{15}$$

And

$$originallabels(h_i, m) = [p_{i1}^{m}, p_{i2}^{m}, \ldots, p_{in}^{m}] \tag{16}$$

As the time is more closed to m o'clock, passengers' demand is more correlative to the value at m o'clock. Hence, we need weight $train(h_i, m)$ to enhance predicting accuracy. That is:

$$original\ features(h_i, m, w) = \begin{bmatrix} p_{i1}^{m-k} \times w_{m-k}, p_{i1}^{m-k+1} \times w_{m-k+1}, \ldots, p_{i1}^{m-1} \times w_{m-1} \\ p_{i2}^{m-k} \times w_{m-k}, p_{i2}^{m-k+1} \times w_{m-k+1}, \ldots, p_{i2}^{m-1} \times w_{m-1} \\ \ldots \\ p_{in}^{m-k} \times w_{m-k}, p_{in}^{m-k+1} \times w_{m-k+1}, \ldots, p_{in}^{m-1} \times w_{m-1} \end{bmatrix}^{\mathrm{T}} \tag{17}$$

Although we have extracted pickup points in predicted hour as labels, these labels are continuous. We need to quantify these labels.

$$labels(h_i, m) = [q_{i1}, q_{i2}, \ldots, q_{in}] \tag{18}$$

Therefore, we can collect features and labels as training data. That is:

$$trainset(h_i, m, w) = \{features(h_i, m, w), labels(h_i, m)\} \tag{19}$$

In our experiment, we treat $train$ as training data and $labels$ as labels. Next, we use machine learning algorithm to train the prediction model based on these two matrices. Finally, we can predict the passengers' demand based on our trained machine learning model.

IV. Service Recommendation

In the process of recommending hotspots, the impact of different historical data on the recommendation is crucial. Therefore, it is not convincing to use fixed historical data to analyze and recommend hotspots. At present, the division of historical data mostly focuses on distinguishing working days and non-working days, so that the data is still fixed from this perspective. In order to dynamically select the historical data in real-time, this paper applies prediction result of passengers' demand to select historical data, based on offline historical data and online taxis passenger' demand. Then we recommend the passenger hotspot according to the location of taxis released. The following part focuses on the taxi hotspot recommendation framework, and the process is shown in Fig. 3.

In Fig. 3, there are two processes: online processing and offline processing. Processing above the red dotted line is taxis historical data offline processing process, while that under the red is the online recommendation process. The processing of historical data is divided into two aspects, the left side of box represents passengers' demand prediction process, and the right side represents the passenger hotspot recommendation process.

In the term of passengers' demand prediction, we firstly construct passengers' demand prediction model PDHH based on taxis historical trajectory data. Secondly, we implement IDT prediction algorithm, where the IDT prediction model is trained according to the previous processing result. Then, the passengers' information that is feedback from the real-time traffic data is input to the IDT prediction model. Finally,

we integrate the PDHH model and the prediction result of passengers' demand, and screen out the time-space index of the historical data closet to the predicted passengers' demand.

The construction of the PDHH model is completed with offline manner, while the real-time prediction of the passengers' demand is completed with online manner, which can greatly reduce the workload of the online process. In the process of predicting passenger hotspot, the cruising efficiency and the driving time required between the hotspots are extracted from the historical trajectory of the taxis. Secondly, we select the cruising efficiency and the driving time required between hotspots of the corresponding time according to the input time-space index. Furthermore, when the driver sends a request for a passenger, HRHT is used to select the passenger hotspot and return the result to drivers. As it can be seen from Fig. 3, it only takes one step from the driver, to make a cruise request to the HRHT with a returned recommendation result. This will greatly reduce the amount of computing and support high concurrent access.

TABLE II
GPS ORIGINAL INFORMATION

id	state	timestamp	latitude	longitude	valid
195839	0	1451518112	39.96019	116.4902	1

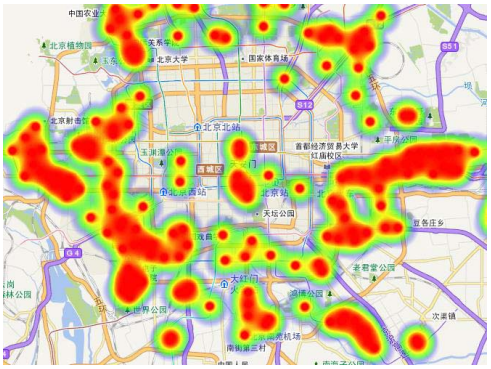


Fig. 4. Distribution of hotspot.

A. Data Preprocessing

Our trajectory data is generated by over 12,000 taxis in one month, in Beijing City [18]. For each day, there are 12G data records with 30 million points. The information contained in the taxi GPS data is shown in the table below. Firstly, we need to clean the original data, such as detecting invalid fields and duplicate data. Secondly, we extract pickup points of passengers by sorting GPS logs by time, and filter the points that state jump from 0 to 1. Furthermore, we need to find the areas that the most drivers intend to wait for a passenger or cruise a passenger. Consequently, we cluster these pickup points to find hotspot shown in Fig. 4. We find 274 hotspots by analyzing 360G historical trajectory data that was recorded one month in Beijing City. Further, we extract pickup points near to hotspot. At last, we partition these data in to 24 parts according to each hour in one day.

The specific methods of data preprocessing are as follows:

1) Obtain historical trajectory data of Beijing taxis.
2) Data cleaning: delete useless records for this paper, delete duplicate data, etc.
3) Load the data processed in step 2. Map the data into an object whose main key is the taxi number and other fields are values.
4) Collect elements with the same primary key into a collection.
5) Sort the value of each element of the data object according to the time field.
6) The place where the passenger loading status changes from scratch is the taxi pickup point.
7) Extract the latitude and longitude information in the passenger point data. Use DBSCAN algorithm to cluster them.

B. Improvement Decision Tree Algorithm

Due to the correlation of the passengers' demand, the latest data in forecasted information is more valuable for the passengers' demand in future, while the obsolete data has smaller reference value. Therefore, IDT algorithm firstly uses the information gain theory in decision trees to analyse the impact of data at different moments on predicted value.

In the process of predicting the passengers' demand of hotspots, the k value and w vector in formula (17) have a significant influence on the prediction result. The information theory of Decision Tree algorithm [28] can effectively analyze the degree of influence of passengers' demand on prediction results in different time period to further set up k and w. Since the passengers' demand data is a continuous value, and the C4.5 algorithm (a kind of decision tree algorithm) as a decision tree algorithm can process the continuity of data [28]. Here, IDT uses the C4.5 algorithm to analyze the data at different moments.

Firstly, we compute the information gain value for each attribute in the training sample. That is:

$$Entropy(A) = -\sum_{i=1}^{c} p_i \log_2 p_i \tag{20}$$

where p_i is the probability of the occurrence of different values in attribute A, and c represents the number of attributes.

Further, we calculate the influence of each attribute occurred on the predicted target, namely the information gain:

$$Gain(T, A) = Entropy(T) - \sum_{i=1}^{c} p_i Entropy(T|A = a_i) \tag{21}$$

Finally, we calculate the information gain rate for each attribute:

$$Gain\ Ratio(T, A) = \frac{Gain(T, A)}{Split\ Information(T, A)} \tag{22}$$

Thereby, we have:

$$Split\ Information(T, A) = -\sum_{i=1}^{c} \frac{T_i}{T} \log_2 \frac{T_i}{T} \tag{23}$$

We collect all the information gain rate of attributes into the Attribute List, that is:

$$Attribute\ List = \{a_1, a_2, \ldots a_k\} \tag{24}$$

Attribute List can effectively reflect the extent to which the number of passengers affect prediction result at different moments in historical data, and accurately set the k value in the formula (17) by deleting the attribute with less information gain in the attribute list, then weighting the selected first k hours of data to increase the accuracy of prediction.

The IDT algorithms use the Adaptive Boosting (AdaBoost) method to construct a single-layer decision tree [28]. The AdaBoost method enhances the classification by integrating multiple weak classifiers into one strong classifier. The classification idea is to set the sample weight value for each sample data, and update the weight value of each sample according to the classification error rate. Throughout continuously updating the sample weight value, the sample weight of correct classification is decreased, and the sample weight of incorrect classification is increased. Since passengers' demand prediction needs to apply recent data to update the prediction model regularly in actual application, AdaBoost method can effectively solve this problem.

IDT algorithm is shown in below:

Algorithm 1 IDT Algorithm

1: Input: Import $trainset(h_i, m, w)$ as trainset T
2: k: train number
3: Output: IDT model
4: Start:
5: set every weight in $trainset(h_i, m, w)\ as\ \frac{1}{|T|}$
6: **for** $(i = 1; i < k; i++)$ **do**
7: extract samples in T as T_i train single-level decision tree model as M_i
8: compute error ε of M_i
9: **if** $\varepsilon > 0.5$ **then**
10: return to step 6
11: **end if**
12: **for** $T_{ij}\ in\ T_i$ **do**
13: update weights in T_{ij} according to predicting results
14: **end for**
15: **end for**
16: return all train models
17: End

First, we set an initial weight value for each sample in the training data set, to obtain a weight value vector D. Further, we train a single-layer decision tree model and calculate the error rate of each model:

$$\varepsilon = \frac{num_d}{num_T} \tag{25}$$

where num_d is the number of incorrect samples, num_T is the number of training samples. Further, we calculate the weight value of decision tree α:

$$\alpha = \frac{1}{2}\ln(\frac{1-\varepsilon}{\varepsilon}) \tag{26}$$

With the calculation on weight value of decision tree α, we update the weight D_i of each set of data based on the predicted result:

When the predicted result is correct, we have:

$$D_i^{t+1} = \frac{D_i^{(t)} e^{-\alpha}}{sum(D)} \tag{27}$$

When the predicted result is wrong, we have:

$$D_i^{t+1} = \frac{D_i^{(t)} e^{\alpha}}{sum(D)} \tag{28}$$

where $sum(D)$ means the sum of all the weights. With this, the sample weight value of classification error is increased through updating the sample weight value.

We get the prediction function by updating the weights k times, as the following shows:

$$f(x) = \sum_{i=1}^{k} \alpha_k DT_k(x) \tag{29}$$

where k is the number of classifiers, and $DT_k(x)$ represents the k^{th} decision tree classification result.

C. Trajectory Recommendation Based on Online Traffic Flow

According to Definition 11, recommended passenger hotspots are supposed to have smaller cruising time $t_{ijk}^{cruising}$, higher probability of cruising passenger $probability_{jk}$ and higher revenue of carrying passenger r_{jk}. Since these three factors belong to different dimensions, it is difficult to select optimal hotspot. Each factor needs to be quantified. In the case of passenger hotspot recommendation, we should quantify the hotspots that are close to the above influencing factors as long as possible, and separate the passenger hotspots which are far from the influencing factors.

As shown in the Fig. 5, the horizontal axis is passenger hotspot sorted based on cruising time, and the longitudinal axis is the average cruising time of each hotspot. Points in same rectangular box in Fig. 5 represent the hotspots that have same quantitative value in this article. In order to achieve the quantitative goal in Fig. 5 (means the points within each rectangular box are as centralized as possible), we thus minimize the variance of cruising time for points in each rectangular box.

Taking the quantitative taxi cruising time as an example, we assume the location of a taxi is h_i, time is the k^{th} period, and the quantitative length of cruising time is x, and there are l hotspots in total.

Firstly, TR-RHT sorts the cruising time that taxi needs to spend among l hotspots into hotspot cruising time vector, with ascending order:

$$T^{cruising}(h_i, k) = \{t_{i1k}^{cruising}, t_{i2k}^{cruising}, \ldots, t_{ilk}^{cruising}\} \tag{30}$$

Then, the hotspot cruising time vector $T^{cruising}$ is divided into k sub-vectors, resulting in:

$$T^{cruising}(h_i, k, x) = \{T_1^{cruising}, T_2^{cruising}, \ldots, T_k^{cruising}\} \tag{31}$$

The average variance of these k sub-vectors is:

$$Var(T^{cruising}(h_i,k,x)) = \sum_{i=1}^{k}(\frac{|T_i^{cruising}|}{|T^{cruising}(h_i,k} \times Var(T_i^{cruising})) \quad (32)$$

where $Var(T_i^{cruising})$ represents the variance of the i^{th} cruising time sub-vector. The minimum $Var(T^{cruising}(h_i,k,x))$ is the optimal segmentation method. The methods of quantifying the probability of cruising passenger and the revenue of carrying passenger are same as the method mentioned above. As a result, the cruising time, the probability of cruising passenger and the revenue of carrying passenger h_i to each passenger hotspot h_j can be quantified as:

$$parameter(h_i,h_j,k)=\{k-1-q_{ijk}^{time}, q_{ijk}^{probability}, q_{ijk}^{revenue}\} \quad 1 \le j \le l \quad (33)$$

In the function (32), q_{ij}^{time}, $q_{ij}^{probability}$, $q_{ij}^{revenue}$ are the quantitative results of cruising time, the probability of cruising passenger and the revenue of carrying passengers respectively. Each value of them is between 0 and $k-1$. Since the cruising time should be minimized, and the probability of finding passengers and the revenue of carrying passenger should be maximized, we utilize $k-1-q_{ij}^{time}$ quantitative results to replace the cruising time. Then results can still be guaranteed $k-1-q_{ij}^{time}$ is between 0 and $k-1$, and the selected target is changed to search the hotspot with larger value of $k-1-q_{ij}^{time}$, $q_{ij}^{probability}$ and $q_{ij}^{revenue}$.

If there is a hotspot that three quantified values are all larger than the hotspot h_j, the hotspot h_j should not be used as a target recommended to the driver. As shown in Fig. 6, there are three passenger hotspots h_1, h_2 and h_3, and the values of each hotspot on different axis are the quantitative results of the cruising time, probability of cruising passengers and the revenue of carrying passengers. It can be seen from Fig. 6 that the value of hotspot h_1 on three axes are all greater than the value of hotspot h_2. This means that the taxis drives to hotspot h_1 for cruising passenger are superior to hotspot h_2. Therefore, hotspot h_2 should be removed from the pre-recommended hotspots. Moreover, compared with hotspot h_3, hotspot h_1 has smaller probability of finding passengers, but has advantage in cruising time and the revenue of carrying passengers. This means there is a competition between hotspot h_1 and hotspot h_3. For this reason, hotspot h_1 and hotspot h_3 should be recommended to drivers.

To sum up, we define the hotspots closer to the original point as internal hotspots and the hotspots far from original point as edge hotspots. The purpose of finding optimal hotspots is to screen out the "edge" hotspots, rather than recommending "internal" hotspots. In order to find the "edge" hotspot quickly, this paper calculates a weight for each hotspot, which is the sum of cruising time, probability of finding passengers and the revenue of carrying passengers.

$$weight(h_i,h_j,k) = k-1-q_{ijk}^{time}+q_{ijk}^{probability}+q_{ijk}^{revenue} \quad (34)$$

Further, we sort the weight of passenger hotspot in descending order, and record the sorted result as:

$$S(h_i,k) = (weight_1, weight_2, \ldots, weight_i) \quad (35)$$

The sorted result $S(h_i,k)$ has the following properties:

Property 1: Since $weight_1$ is the maximum weight, we can obtain the "edge" node of the hotspot it belongs to: If h_a has the maximum weight and it belongs to "internal" hotspot, there will be a h_b whose cruising time, probability of cruising passengers and revenue of carrying passenger are all higher than it. Therefore, the weight of h_b $(weight(h_i,h_b,k))$ is higher than the weight of h_a $(weight(h_i,h_a,k))$. This is conflict with the assumption that h_a has the maximum weight. For this reason, h_a is an "edge" hotspot.

Property 2: If the cruising time, probability of finding passengers and the revenue of carrying passenger of the hotspot $weight_2$ located are all smaller than the hotspot $weight_1$ located, then the hotspot $weight_2$ located is an "edge" hotspot; Otherwise, the hotspot $weight_2$ located is an "edge" hotspot. Proved as follow: If the hotspot has the weight of $weight_2$ is an "internal" hotspot, then there is a hotspot h_b whose cruising time, probability of finding passengers and the revenue of carrying passenger of hotspot are all larger than the hotspot $weight_2$ located, therefore, the weight of h_b is larger than $weight_2$. However, there is only $weight_1$ has higher weight than $weight_2$, which conflicts with the assumption.

Algorithm 2 HRHT Algorithm

```
Input: Parameter of every hotspots
Output: Hotspots set R
Start:
  compute weights of Parameters, and sort to S(h_i, k)
  create hotspots set R
  add first hotspot in S(h_i, k) to R, and delete it from S(h_i, ryk)
for each weight_i in S(h_i, k) do
  for each weight_i in R do
    if every element in weight_i is less than weight_i then
      delete weight_i from S(h_i, k), return to step 8
    end if
  end for
  if S(h_i, k) contains weight_i then
    add weight_i to R, and delete it from S(h_i, k)
  end if
end for
End
```

Furthermore, we judge each element sequentially in sequence $S(h_i,k)$, and filter the "edge" hotspots, and then recommend the filtered "edge" hotspot to the drivers, the filtering algorithm is shown above. First of all, we calculate the weight of each passenger hotspot $S(h_i,k)$, and sort it with ascending order. Secondly, add the first passenger hotspot in the sorted result into the pre-recommended passenger hotspot collection R. Finally, according to Property 2, we discuss whether the remaining passenger hotspots are "edge" hotspots in order.

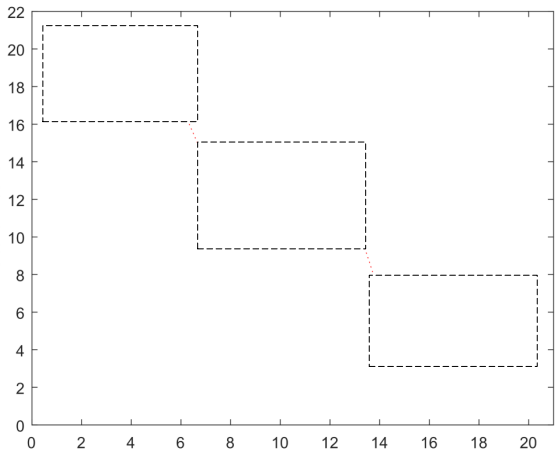


Fig. 5. Quantitative schematic map of cruising time.

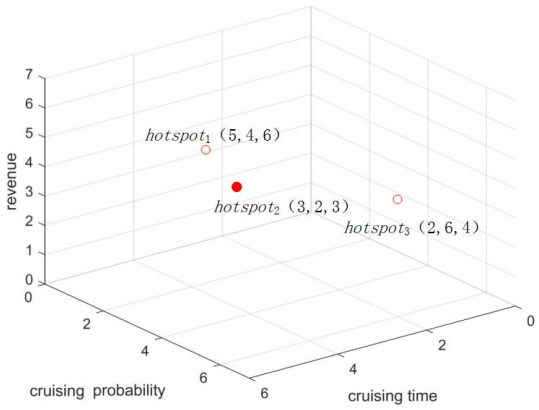
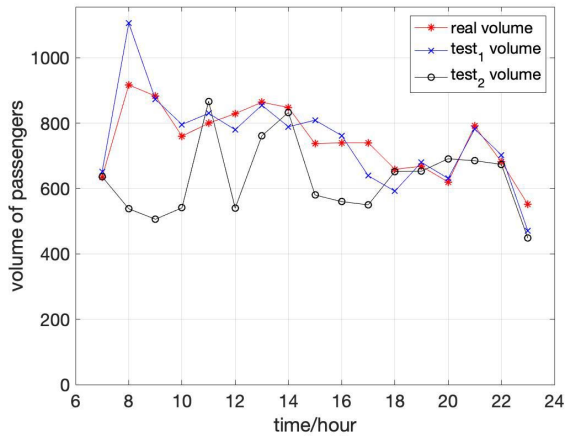
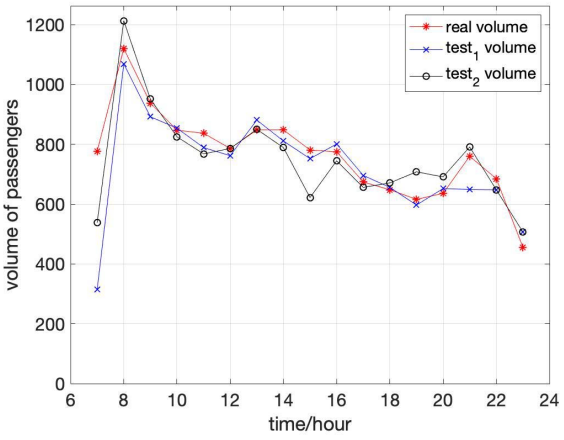


Fig. 6. Example of screening passenger hotspots.



(a)



(b)

Fig. 7. Result of predicting model by IDT. (a) weekend. (b) weekday.

V. Performance Evaluation

A. Dataset Description and Process

The GPS trajectory data set is collected by over 12,000 taxis running in Beijing in December 2015. There are 30 million records with the size of 12G data for each day. Offline preprocessing and frequent trajectories of this paper are constructed on Spark platforms. The latitude range of trajectory data is between 39.26 and 41.03. The longitude range of trajectory data is between 115.25 and 117.3. The sampling interval of GPS data is 30 seconds. Our large amount of original GPS data is stored on the Hadoop Distributed File System (HDFS). Data preprocessing and analysis of this paper are constructed on Spark platforms.

B. Compared Methods

The TR-RHT strategy is composed of two algorithms: IDT predicting algorithm and HRHT recommending algorithm. IDT algorithm as a classification method is compared with KNN algorithm and SVM algorithm. In order to compare the errors of these three prediction algorithms, we introduce the following three metrics MSE, MAE, and $RMSE$, to evaluate our proposed model. These three evaluation functions are as follows:

- *MSE:* It can measure the performance of the model by computing the mean squared error of the model on the test set as follows:

$$MSE = \frac{1}{m}\sum_{i=1}^{i=m}(y_i^{real} - y_i^{predict})^2 \tag{36}$$

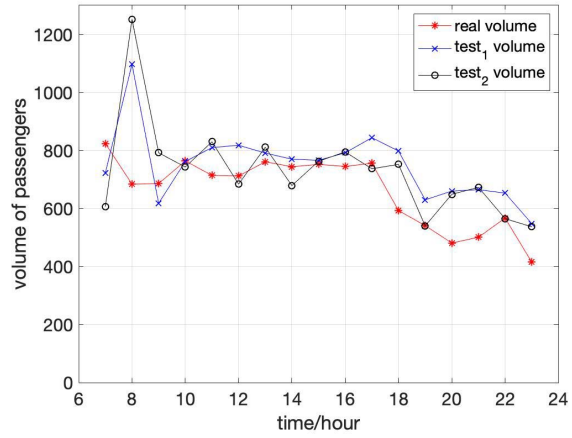
- *MAE:* Its measurement method is similar to MSE. It computes the mean absolute error as follows:

$$MAE = \frac{1}{m}\sum_{i=1}^{i=m}\left|y_i^{real} - y_i^{predict}\right| \tag{37}$$

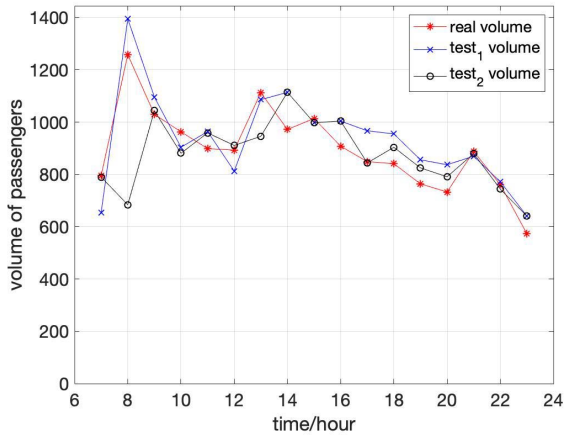
- *RMSE:* It computes the square root of MSE as follows:

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{i=m}(y_i^{real} - y_i^{predict})^2} \tag{38}$$

The HRHT algorithm is compared with two methods: SCRAM [1], LCP [20]. This paper compares the improvement of HRHT algorithm on cruising time, cruising probability and revenue.

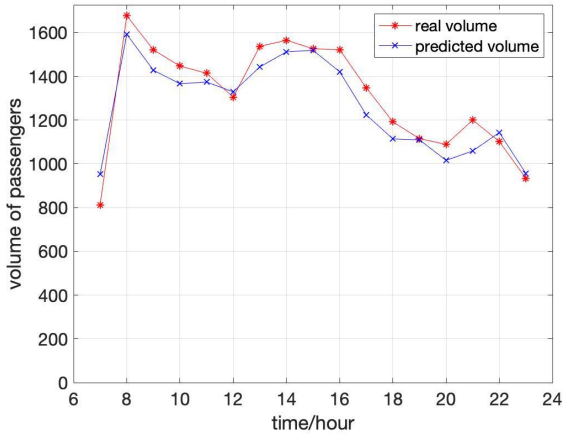


(a)

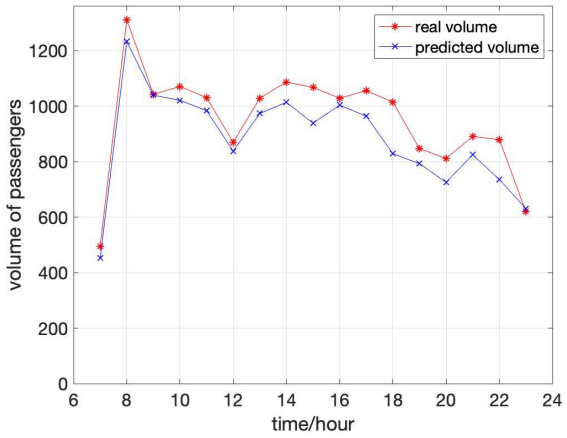


(b)

Fig. 8. Result of predicting model by SVM. (a) weekend. (b) weekday.



(a)



(b)

Fig. 9. Result of predicting model by KNN. (a) weekend. (b) weekday.

C. Experimental Results Analysis

1) Performance of Predicting Model: In this part, we predict the capacity of passengers in the given time and hotspot to enhance the accuracy of recommendation. For a given hotspot, we require the data should be related to passengers' demand in past 6 hours, to infer the passengers' demand in forthcoming hour. Since the data in the early morning has a lot of statistical noise, we only predict the passengers load capacity after 7 o'clock. We compare our proposed model IDT with the other two methods KNN and SVM according to three given evaluation indexes.

In this article, we first construct a prediction algorithm model, and then randomly group the 30-day historical trajectory data in Beijing. We use 80% of the data as input data to train the model, and 20% of the data as predictive data (real-time traffic data) to test the accuracy of model prediction.

In Fig. 7 and Fig. 8, we respectively predict twice in each picture by IDT and SVM, $test_1$ data represents the result that distinguishes weekdays and weekends, $test_2$ data represents the result without distinguishing weekdays and weekends. Because SVM algorithm is binary classification, it needs adaptation in our application scenario and costs a long time in our experiment. From Fig. 8, we can find some time it will not accurately predict the result. Fig. 7 illustrates that our proposed IDT algorithm is capable of predicting model.

Fig. 9 shows the actual capacity and predicted capacity of passengers in every hour within one day. Since the KNN algorithm is looking for the closest distance (Euclidean distance) from today, we don't distinguish weekdays and weekends here. In these pictures, we can find the result computed by KNN algorithm basically coincides with original data. However, KNN algorithm is not a machine learning algorithm that needs to be trained, it needs to load metadata in each prediction. Fig. 10 further shows the different performance evaluations in MAE, MSE and RMSE.

We find the mean value of IDT, SVM and KNN algorithm is respectively 127.1, 243.8, 20.0 and in Fig. 10(a). The mean value of IDT, SVM and KNN algorithm is respectively 10.3, 14.4 and 4.2 in Fig. 10(b). The mean value of IDT, SVM and KNN algorithm is respectively 61.8, 114.8 and 33.7 in Fig. 10(c). Although we ignore the difference between weekdays and weekends, our prediction model can still accurately predict the passengers' demand on weekdays and weekends.

Fig. 11 shows the empirical CDF of predicting error. It can be clearly seen from the Fig. 11 that the best algorithm is KNN, followed by IDT, and finally SVM. Although KNN

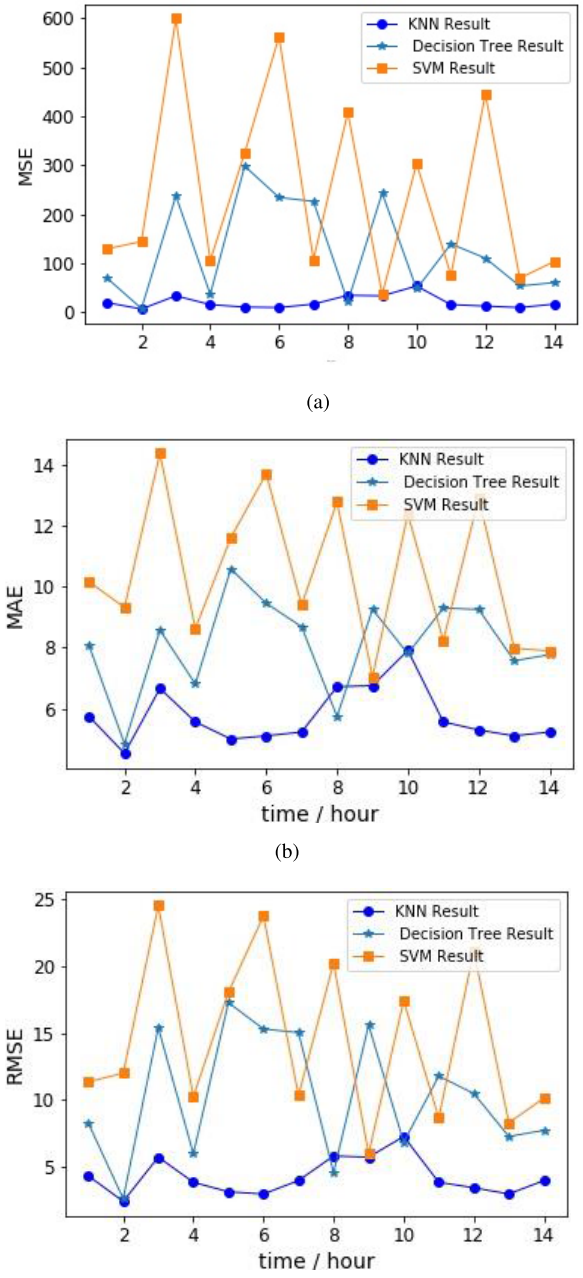


Fig. 10. Error Analysis (a) MSE (b) MAE (c) RMSE.

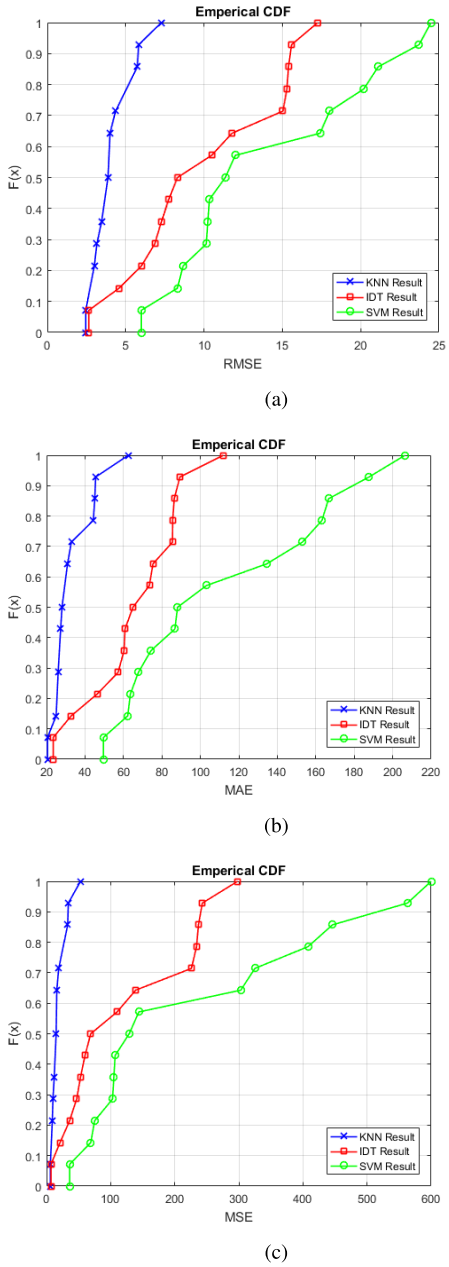


Fig. 11. Emperical CDF of predicting error (a) RMSE (b) MAE (c) MSE.

performs very well in the prediction process, KNN needs to load metadata during the training process. As the amount of data increases, it may not be feasible in practical applications. Binary classification characteristics of SVM leads to increased computation time when training the model. However, SVM has indispensable merit is to avoid the problems encountered when we utilize KNN algorithm. Obviously, our proposed IDT algorithm performs well in this model.

From the above experimental results: The KNN algorithm mentioned in this article needs to reload historical data per each prediction, therefore it can't meet real-time requirements in practical applications due to computation time cost. Although the SVM algorithm can train the model, it is not suitable for prediction of passenger capacity. When the taxi history data is updated regularly, the model needs to be retrained. Among the historical taxi data, data with a short interval is more meaningful for prediction, while data with a longer interval is less meaningful for the prediction result. The SVM algorithm is difficult to implement and predict data from this perspective. The IDT algorithm proposed in this paper can solve these two problems.

2) Experimental Results of Recommendation System: This section evaluates the performance of HRHT recommendation in three factors: strategy on cruising time, probability of cruising a passenger and the revenue of carrying passenger. Three options are generated as HRHT recommendation, LCP recommendation and SCRAM recommendation by applying a comparison method [20].

Fig. 12 shows the performance of cruising probability in HRHT recommendation strategy. Three polylines represent

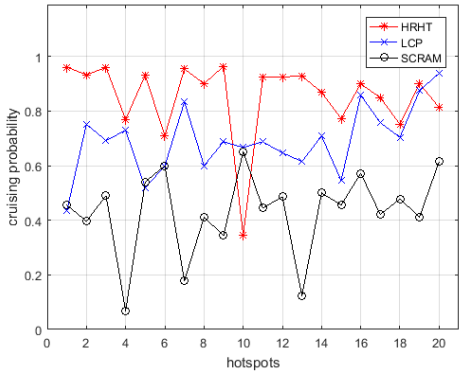


Fig. 12. Probability of cruising passengers in the recommended hotspots.

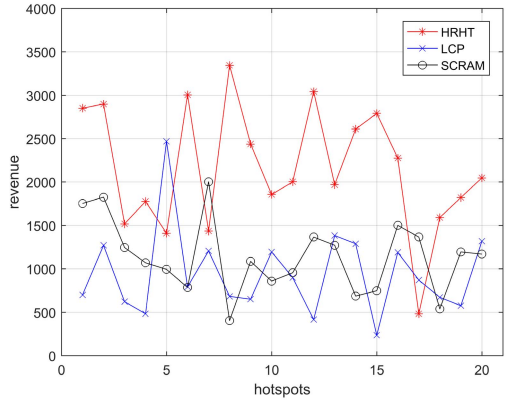


Fig. 13. The revenue of carrying passenger in the recommended hotspot.

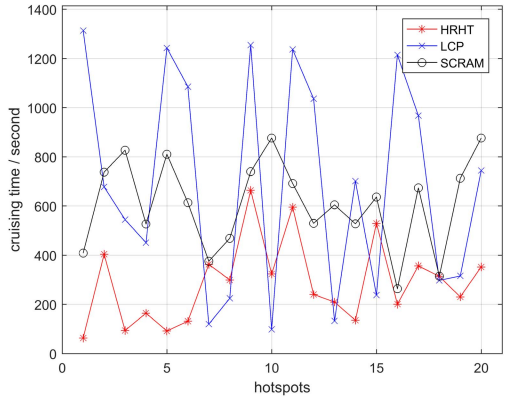


Fig. 14. The revenue of carrying passenger in the recommended hotspot.

HRHT recommendation, LCP recommendation and SCRAM recommendation, respectively. According to Fig. 12, the average cruising probability of the HRHT recommendation proposed in this paper is 0.85; the average cruising probability of the LCP recommendations 0.69; the average cruising probability of the SCRAM recommendation is 0.43.

The performance of the HRHT recommendation strategy on the revenue of carrying passenger is shown in Fig. 13, and three polylines represent HRHT recommendation, LCP recommendation and SCRAM recommendation, respectively. According to Fig. 13: the average revenue of the HRHT recommendation proposed in this paper is the fare of 35-minutes driving; the average revenue of the LCP recommendation is the fare of 15-minutes driving; the average revenue of the SCRAM recommendation is the fare of 19-minutes driving.

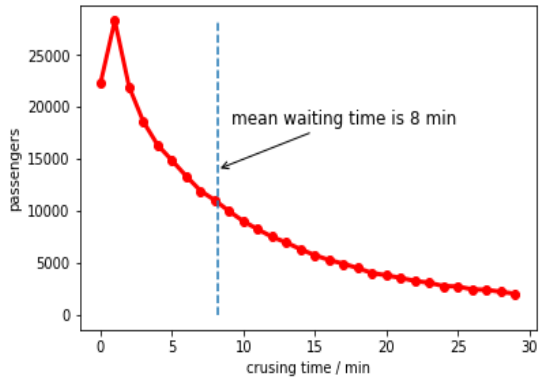


Fig. 15. Distribution of real cruising time.

Fig. 14 shows that the performance of the cruising time of HRHT recommendation strategy, and three polylines represent HRHT recommendation, LCP recommendation and SCRAM recommendation, respectively. According to Fig. 14, the average cruising time of the HRHT recommendation strategy proposed in this paper is 4.3 minutes; the average cruising time of LCP recommendation is 11 minutes; the average cruising time of the SCRAM recommendation strategy is 10.1 minutes.

In summary, the HRHT recommendation strategy can take into account the taxis cruising time, the probability of cruising passengers and the revenue of carrying passenger in three factors. And the average cruising time of recommended hotspot has been reduced to 4.3 minutes, which has a great improvement on driver's cruising efficiency by comparing with the average cruising time of 8 minutes shown in Fig. 15.

VI. Conclusion

This paper describes a strategy for taxi drivers to cruise passengers. Specifically, PDHH model mines the passengers' demand information in hotspots to predict the heat of passenger hotspots. The experiment results show that KNN algorithm predicts the best result under the premise of ignoring time cost in experiment. The reason for long running time of KNN algorithm is that KNN needs to load metadata. Conversely, SVM does not show good results in forecasting passenger number. A partial explanation for this may lie in the fact that the SVM algorithm is binary classification. The two results, however, complement each other well, for each emphasizes a different aspect of recommendation. On the other hand, the proposed HRHT algorithm reduces the time cost of taxis by half. The experiments achieve HRHT recommendation strategy which considers the probability of cruising a passenger, the revenue of carrying passenger and cruising time three factors. These three factors can equalize the cruising efficiency of passenger hotspots. The average cruising time of HRHT recommendation strategy is 4.3 minutes, which will save half of the cruising time of taxis from the perspective of the entire city.

References

[1] S. Qian, J. Cao, F. L. Mouël, I. Sahel, and M. Li, "SCRAM: A sharing considered route assignment mechanism for fair taxi route recommendations," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 955–964.

[2] H. Rong, X. Zhou, C. Yang, Z. Shafiq, and A. Liu, "The rich and the poor: A Markov decision process approach to optimizing taxi driver revenue efficiency," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2016, pp. 2329–2334.
[3] W. Ma and Z. S. Qian, "Statistical inference of probabilistic origin-destination demand using day-to-day traffic data," *Transp. Res. C, Emerg. Technol.*, vol. 88, pp. 227–256, Mar. 2018.
[4] K. M. S. Huq, S. Mumtaz, J. Rodriguez, P. Marques, B. Okyere, and V. Frascolla, "Enhanced C-RAN using D2D network," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 100–107, Mar. 2017.
[5] S. Mumtaz, H. Lundqvist, K. M. S. Huq, J. Rodriguez, and A. Radwan, "Smart direct-LTE communication: An energy saving perspective," *Ad Hoc Netw.*, vol. 13, pp. 296–311, Feb. 2014.
[6] J. Engelbrecht, M. J. Booysen, G.-J. van Rooyen, and F. J. Bruwer, "Survey of smartphone-based sensing in vehicles for intelligent transportation system applications," *IET Intell. Transp. Syst.*, vol. 9, no. 10, pp. 924–935, Dec. 2015.
[7] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 1, pp. 383–398, Jan. 2019.
[8] M. Z. Khan, S. Harous, S. U. Hassan, M. U. G. Khan, R. Iqbal, and S. Mumtaz, "Deep unified model for face recognition based on convolution neural network and edge computing," *IEEE Access*, vol. 7, pp. 72622–72633, 2019.
[9] Y. Dong, S. Qian, K. Zhang, and Y. Zhai, "A novel passenger hotspots searching algorithm for taxis in urban area," in *Proc. 18th IEEE/ACIS Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distrib. Comput. (SNPD)*, Jun. 2017, pp. 175–180.
[10] X. Kong, F. Xia, J. Wang, A. Rahim, and S. K. Das, "Time-location-relationship combined service recommendation based on taxi trajectory data," *IEEE Trans. Ind. Informat.*, vol. 13, no. 3, pp. 1202–1212, Jun. 2017.
[11] Q. Yang, Z. Gao, X. Kong, A. Rahim, J. Wang, and F. Xia, "Taxi operation optimization based on big traffic data," in *Proc. IEEE 12th Int. Conf. Ubiquitous Intell. Comput., IEEE 12th Int. Conf. Autonomic Trusted Comput., IEEE 15th Int, Conf. Scalable Comput. Commun. Associated Workshops (UIC-ATC-ScalCom)*, Aug. 2015, pp. 127–134.
[12] J. Dai, B. Yang, C. Guo, and Z. Ding, "Personalized route recommendation using big trajectory data," in *Proc. IEEE 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 543–554.
[13] J. Xu, R. Rahmatizadeh, L. Boloni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2572–2581, Aug. 2018.
[14] Z. Huang, J. Tang, G. Shan, J. Ni, Y. Chen, and C. Wang, "An efficient passenger-hunting recommendation framework with multitask deep learning," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7713–7721, Oct. 2019.
[15] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.
[16] S. Khan, K. Muhammad, S. Mumtaz, S. W. Baik, and V. H. C. de Albuquerque, "Energy-efficient deep CNN for smoke detection in foggy IoT environment," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9237–9245, Dec. 2019.
[17] S. Qian, Y. Zhu, and M. Li, "Smart recommendation by mining large-scale GPS traces," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2012, pp. 3267–3272.
[18] X. Xu, J. Y. Zhou, Y. Liu, Z. Z. Xu, and X. W. Zha, "Taxi-RS: Taxi-hunting recommendation system based on taxi GPS data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1716–1727, Aug. 2015.
[19] T. Lei, S. Wang, J. Li, and F. Yang, "A cooperative route choice approach via virtual vehicle in IoV," *Veh. Commun.*, vol. 9, pp. 281–287, Jul. 2017.
[20] Y. Ge, H. Xiong, A. Tuzhilin, K. Xiao, M. Gruteser, and M. Pazzani, "An energy-efficient mobile recommender system," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2010, pp. 899–908.
[21] H. Hu, Z. Wu, B. Mao, Y. Zhuang, J. Cao, and J. Pan, "Pick-up tree based route recommendation from taxi trajectories," in *Proc. Int. Conf. Web-Age Inf. Manage.* Berlin, Germany: Springer, 2012, pp. 471–483.
[22] H. Hu, Z. Wu, M. Bo, Z. Yi, and J. Pan, "Pick-up tree based route recommendation from taxi trajectories," in *Proc. Int. Conf. Web-Age Inf. Manage.*, 2012, pp. 471–483.
[23] W. Yang, X. Wang, S. M. Rahimi, and J. Luo, "Recommending profitable taxi travel routes based on big taxi trajectories data," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2015, pp. 370–382.
[24] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.
[25] M. Zhang, J. Liu, Y. Liu, Z. Hu, and L. Yi, "Recommending pick-up points for taxi-drivers based on spatio-temporal clustering," in *Proc. 2nd Int. Conf. Cloud Green Comput.*, Nov. 2012, pp. 67–72.
[26] Y. Huang, L. Zhao, T. Van Woensel, and J.-P. Gross, "Time-dependent vehicle routing problem with path flexibility," *Transp. Res. B, Methodol.*, vol. 95, pp. 169–195, Jan. 2017.
[27] Y. Shen, L. Zhao, and J. Fan, "Analysis and visualization for hot spot based route recommendation using short-dated taxi GPS traces," *Information*, vol. 6, no. 2, pp. 134–151, Apr. 2015.
[28] P. Harrington, *Machine Learning in Action*. Shelter Island, NY, USA: Manning Publications Co., 2012.

Tong Wang (Member, IEEE) received the Ph.D. degree in computer application from Harbin Engineering University, Harbin, Heilongjiang, China, in 2006. He is currently a Professor with the Information and Communication Engineering College, Harbin Engineering University. His research interests include wireless networks, vehicular *ad hoc* networks, and the Internet of Things.

Zhaoxian Shen received the M.S. degree in information and communication engineering from Harbin Engineering University, Harbin, Heilongjiang, China, in 2019. His current research interests include big data mining and ITS.

Yue Cao (Member, IEEE) received the Ph.D. degree from the Institute for Communication Systems (ICS); formerly known as Centre for Communication Systems Research, University of Surrey, Guildford, U.K., in 2013. He was a Research Fellow with the University of Surrey and an Academic Faculty with Northumbria University, Lancaster University, U.K., and Beihang University, China. He is currently a Professor with the School of Cyber Science and Engineering, Wuhan University, China. His research interests focus on intelligent transport systems, including e-mobility, V2X, and edge computing.

Xiujuan Xu received the Ph.D. degree from the College of Computer Science and Technology, Jilin University, Jilin, Changchun, China, in 2008. She is currently an Associate Professor with the School of Software, Dalian University of Technology. In 2016, she worked as a Visiting Scholar with the School of Computer Science, The University of Adelaide. She is the author of more than 50 publications. Her research interests include data mining, intelligent transportation systems, recommendation systems, and social network analysis.

Huiwen Gong received the B.E. degree in communication engineering from Southwest Minzu University, Chengdu, Sichuan, China, in 2019. She is currently pursuing the M.S. degree in electronic and communication engineering with Harbin Engineering University, Harbin, Heilongjiang, China. Her current research interests include traffic analysis and data mining.