Optimizing Taxi Driver Profit Efficiency: A Spatial Network-Based Markov Decision Process Approach

Xun Zhou, Huigui Rong, Chang Yang, Qun Zhang, Amin Vahedian Khezerlou, Hui Zheng, Zubair Shafiq, and Alex X. Liu

Abstract—Taxi services play an important role in the public transportation system of large cities. Improving taxi business efficiency is an important societal problem. Most of the recent analytical approaches on this topic only considered how to maximize the pickup chance, energy efficiency, or profit for the immediate next trip when recommending seeking routes, therefore may not be optimal for the overall profit over an extended period of time due to ignoring the destination choice of potential passengers. To tackle this issue, we propose a novel Spatial Network-based Markov Decision Process (SN-MDP) with a rolling horizon configuration to recommend better driving directions. Given a set of historical taxi records and the current status (e.g., road segment and time) of a vacant taxi, we find the best move for this taxi to maximize the profit in the near future. We propose statistical models to estimate the necessary time-variant parameters of SN-MDP from data to avoid competition between drivers. In addition, we take into account fuel cost to assess profit, rather than only income. A case study and several experimental evaluations on a real taxi dataset from a major city in China show that our proposed approach improves the profit efficiency by up to 13.7 percent and outperforms baseline methods in all the time slots.

Index Terms—Markov decision process, spatial network, profit efficiency, driving recommendation

1 INTRODUCTION

Taxi services are playing an important role in the public transportation system in modern cities. Taxi drivers are a big social group in many major cities in the world. Improving taxi business efficiency helps increase taxi drivers’ income and thus contributes to the development of the urban economy. Higher efficiency in taxi operations means less driving time and cruising distance needed, and thus leading to lower gas emissions and fuel consumption. With the rapid development of GPS devices and location-based services, detailed public transportation records have been collected in many cities. These “big” datasets provide a chance to transform the current business model of taxi services. Through analyzing these datasets, one could learn optimal driving routes, demand and supply distributions, etc., and provide suggestions to drivers on how to improve their income.

Recent research [2], [3], [4], [5], [6], [7] have focused on developing recommendation systems for taxi drivers. These recent works usually learn knowledge about passenger demand distribution from data and recommend the best route or location for a driver to optimize one or more of the following measures: the profit margins for the next trip [3], [4], the chance of finding the next passenger [2], or energy consumption before finding the next passenger [7].

The above work, although effective in improving the proposed measures, still have some limitations. First of all, some of the above work (e.g., [2], [4], [8]) simply aggregated the historical data over the entire study period and ignored the temporal variation of the passenger distribution. This may lead to inaccuracy in the recommendations. More importantly, all of the existing work focus on optimizing the measures for the immediate next trip. However, they do not consider the impact of the driver’s future pick up opportunities on the overall profit. For example, a taxi may find it very easy to find a passenger in a residential area A in the morning rush hours, where many people want to go to business area B for work. However, there will be very few passengers who need a taxi at this time in B. Although the taxi can find the first passenger easily, the driver may lose more time in the next round of seeking. In other words, the optimization goal for a taxi driver should be the total profit in a time window rather than only for the next trip. Greedy strategies might not always give the best solution.

Our recent work [1] proposed a Grid-based Markov Decision Process Approach. Given a set of historical pick up and drop off records, and the current status (e.g., location) of a vacant taxi, the Grid-based Markov Decision Process Approach aims to find the best move for this taxi to maximize the profit in an extended time window in the near future.
We propose a spatial network-based MDP model, with parameters learned from the data. Case study and experiment results showed that this approach effectively improved the profit efficiency of inexperienced drivers by up to 15 percent and outperformed a baseline by up to 8.4 percent.

This paper is a significant extension of our prior paper [1]. In this paper we substantially improve the approach in our prior work by generalizing the MDP approach to spatial networks and proposing a novel way of learning time-variant parameters from historical taxi data for the MDP model. In the new model, we incorporate information on the recent behavior of the passengers and other taxis to adjust the parameters, therefore avoiding the completing of taxi drivers. We conduct extensive experiments and simulations using real data. Specifically, the major contributions of this paper are summarized as follows:

- We propose a spatial network-based MDP model with rolling-horizon for taxi driving direction recommendation.
- We incorporate fuel cost into the model and give recommendations based on profit (i.e., revenue - cost).
- We propose a new statistical model to learn time-variant MDP parameters (e.g., pick up probability, destination probability) from historical taxi records.
- We dynamically adjust the parameters to account for the recent behaviors of the passengers and taxis and effectively avoid recommending drivers to the same roads.

The rest of this paper is organized as follows: Section 2 introduces the background and the data we are using in this analysis, with a few pre-processing steps. Section 3 presents our analysis to quantify taxi business efficiency and identify successful and unsuccessful drivers. Section 4 discusses our proposed Spatial Network-based MDP model and a dynamic programming algorithm to solve the MDP. Section 5 presents evaluation results on various parameters. Section 6 presents the proposed Spatial Network-based MDP model and a dynamic programming algorithm to solve the MDP. Section 7 presents a detailed discussion of related work. Finally we conclude the paper in Section 7.

2 DATA AND ANALYSIS SETTINGS

The dataset used in this study contains taxi operation records for one year from the capital city of a central province in China. There are approximately 19 million taxi trip records (53,000 per day), where each record has the latitude-longitude coordinates and timestamps of the pick-up and drop-off events, along with total traveled distance and the fare of the corresponding trip. The data contains the records of about 1,400 taxi cabs.

2.1 Background

To ensure the highest utilization of the taxi cabs, taxi companies usually assign two drivers for each taxi cab: a daytime driver driving between 6 AM to 5 PM, and a nighttime driver, working between 5 PM to 6 AM the next day. A small number of taxi cabs may switch drivers slightly earlier or later, according the traffic conditions. In this paper, we assume that all the trips that started between 6 AM and 5 PM are operated by the day-time drivers, while the other trips are operated by the night-time drivers. A trip count distribution over the 24 hours of day is presented in Fig. 1. As can be observed, 4-5 PM has a lower number of trips compared to other day-time time slots. This is because many taxis switch drivers during this hour and do not operate for the entire hour. The average annual revenue of daytime drivers and nighttime drivers are 140,000 Yuan and 120,000 Yuan, respectively. Due to the limited space of this paper, we focus on the day-time drivers’ driving strategies. However, the proposed methods can easily be applied on night-time drivers’ records, too.

2.2 Analysis Settings and Data Summary

In this dataset, there are a number of pick-up/drop-off events falling outside of the metropolitan area of the city. This might be resulted from (1) errors of GPS devices on the taxi, and (2) rare trips to/from suburban areas. Our analysis focuses on the majority of the trips. Therefore, we use a 25.6 km x 25.6 km bounding box to define the study area. GPS records outside this area are filtered out. The map of the study area from Baidu Map [9] is shown in Fig. 2.

In this analysis, we select 821 main roads in the city, which are obtained from Baidu Map. These roads are edges of the road network, and there are 1,633 intersections on the map, i.e., the vertices of the road network. Our road network is an undirected graph, and the weight of each edge is the length. Distribution of road length shown as Fig. 3. Over 95 percent of roads have length below 1,500 meters.

Besides, we have matched pick-up/drop-off points to the selected roads. Each pick-up or drop-off point will be matched to one road which is nearest to it. The distance between the point and road should be less than 200 meters.

As can be observed, 4-5 PM has a lower number of trips compared to other day-time time slots. This is because many taxis switch drivers during this hour and do not operate for the entire hour. The average annual revenue of daytime drivers and nighttime drivers are 140,000 Yuan and 120,000 Yuan, respectively. Due to the limited space of this paper, we focus on the day-time drivers’ driving strategies. However, the proposed methods can easily be applied on night-time drivers’ records, too.

Fig. 1. Average number of trips in the city during a day.

Fig. 2. A map of the study area.
outlier and discard it. As a result, we matched 89.15 percent drop off points, and 92.27 percent pick up points onto the 821 roads. Fig. 4 shows the heat maps of total number of pick up numbers in four one-hour time slots over the entire year. Pick up number shown in Fig. 4 is in log scale.

3 Problem Formulation

In this section we present our analysis to quantify the success of a driver and identify the more (less) successful drivers. Then we identify the most important factors for drivers to improve their business. Based on this analysis we formulate the problem as an optimal decision problem.

3.1 A Measure of Success for Drivers

We first calculate the total business time of each taxi cab to estimate the drivers’ time commitment. The total business time (denoted as \(T_{bus}\)) is the sum of two parts, the total operating time \(T_{drive}\) and the total seeking time \(T_{seek}\) as shown in Equation (1). The total operating time \(T_{drive}\) is the the total time the taxi is carrying a passenger. Calculating total seeking time is not a simple task, because the gap between two consecutive trips might not entirely used by the driver for seeking passengers. Fig. 6 shows the distribution of the length of all the time gaps between trips. As can be observed, 90 percent of the gaps are shorter than 25 minutes, so, we thus use 25 minutes as the threshold, to determine whether a time gap should be considered a seeking time or not

\[
T_{bus} = T_{drive} + T_{seek}.
\]  

3.2 A Driving Efficiency Measure for Drivers

As noted previously, the total business time includes driving time (occupied taxi) and seeking time (vacant taxi). A taxi driver is earning income when driving with passenger but has no earnings while seeking passengers. The profit efficiency of a driver depends on (1) how much money a driver can make per time unit, i.e., driving efficiency, and (2) how quickly the driver can find the next passenger (seeking efficiency). Taking slower routes and running into traffic congestions will lower their driving efficiency which will

\[
E_{profit} = \frac{M - C \times T_{bus}}{T_{bus}} = \frac{M - C \times (T_{drive} + T_{seek})}{T_{drive} + T_{seek}},
\]
result in lower profit efficiency. First we define driving efficiency $E_{drive}$ as follows:

$$E_{drive} = \frac{M - C \times (T_{drive} + T_{seek})}{T_{drive}}.$$  

We also compare the average seeking time of the top 10 percent and bottom 10 percent drivers. Fig. 7c shows the average seeking time (the time gap between consecutive trips that are less than 25 minutes) of the top (yellow bar) and bottom drivers (red bar) in each time slot. Results show that top drivers on average spend 25 to 35 percent less time to seek passengers.

3.3 Optimal Passenger Seeking Strategy

Taxi data has previously been used in the literature to calculate the probability of finding a passenger in each region or along each road segment. Based on this probability and the current location of a driver, one can choose the route or location with the highest probability to find the next passenger.

However, where passengers want to go is also a key issue. Drivers do not choose the destination once they pick up a passenger, because it is illegal in most cities in China to refuse service to any passenger. Taking this issue into consideration, we believe that the driver's decision on where to find the next passenger is very important, not only because it determines the seeking time, but also it determines location of the next seeking trips, which will an impact on the overall profit efficiency. To this end, we formulate the best taxi seeking strategy as follows:

Given.

- The historical passenger trips and seeking trips
- The current status of a vacant taxi
- The underlying spatial network

Find. The next movement for the driver.

Objective. Maximize the total expected profit for this taxi in the next $T_{seek}$ minutes.

Constraint. The underlying spatial network is undirected. Each passenger only waits for a taxi for a certain period of time before disappearing.

The last constraint is added because usually passengers will change their location or try other transportation modes after waiting for too long without being picked up. We discuss the impact of this maximum waiting time of passengers in the Evaluation section.

4 A SPATIAL-NETWORK BASED MARKOV DECISION PROCESS APPROACH

In our previous paper [1] we proposed a grid-based Markov Decision Process approach. For details of the previous approach please refer to our previous article [1]. In this paper, we generalize our model to spatial networks and model the passenger seeking strategy of a taxi as a Spatial Network Based Markov Decision Process (SN-MDP). Solving this SN-MDP model will give us the best seeking strategy for a taxi at each different state. We further discuss a rolling-horizon method to update the best driving recommendation for drivers. Table 1 shows all the parameters we use in the MDP model.

4.1 System States

It is obvious that the best seeking strategy of a taxi is dependent on the current location and the current time. In our model, each state of a vacant taxi is described by three parameters: location of the taxi (i.e., road segment ID) $l \in L = \{1, 2, \ldots, 821\}$, time the recommendations are given for $t \in T = \{1, 2, \ldots, 60\}$, and the road from which the taxi arrived at road $l$, denoted as $d \in D$.

The "arriving direction" is needed for each state to ensure that the taxi does not fall into a infinite loop among a small number of road or staying in the same road forever. In reality, a taxi may not be able to make frequent U-turns or stop on certain roads due to traffic conditions or regulations. Also, traversing the same road twice within a short time is unlikely to improve the chance of finding a passenger. To this end, we require that a taxi cannot cruise the same road twice within a short period of time. Based on where the taxi came to the current road segment, its next move may be 

---

<table>
<thead>
<tr>
<th>Variable</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$l, L$</td>
<td>The current road ID and the road segment set</td>
</tr>
<tr>
<td>$t_{seec}$</td>
<td>The time of driving direction recommendation</td>
</tr>
<tr>
<td>$T_c$</td>
<td>Length of time window for profit optimization</td>
</tr>
<tr>
<td>$t$</td>
<td>Time after $t_{seec}$ (minute)</td>
</tr>
<tr>
<td>$d, D$</td>
<td>Incoming road index to the current road and the set of all the possible incoming roads</td>
</tr>
<tr>
<td>$s$</td>
<td>A state of the MDP model, $s = (l, t, d)$</td>
</tr>
<tr>
<td>$S$</td>
<td>The collection of all the states, $s \in S$</td>
</tr>
<tr>
<td>$a$</td>
<td>An action taken by a vacant taxi</td>
</tr>
<tr>
<td>$A$</td>
<td>The set of all the possible actions, $a \in A$</td>
</tr>
<tr>
<td>$\pi_a(s)$</td>
<td>The allowed actions for state $s$</td>
</tr>
<tr>
<td>$\pi_a(s)$</td>
<td>The optimal action for state $s$</td>
</tr>
<tr>
<td>$t_{seek}(j)$</td>
<td>The time needed for a taxi to cruise road $j$</td>
</tr>
<tr>
<td>$t_{drive}(i, j)$</td>
<td>The time needed to drive from road $i$ to $j$</td>
</tr>
<tr>
<td>$P_{taxi}(j,k)$</td>
<td>The probability a passenger picked up in road $j$ wishes to go to road $k$</td>
</tr>
<tr>
<td>$P_{pass}(j)$</td>
<td>The probability that a passenger can be found in road $j$ during the vacant cruise</td>
</tr>
<tr>
<td>$r(i, j)$</td>
<td>The expected reward (trip fare) from road $i$ to $j$</td>
</tr>
<tr>
<td>$E_{drive}$</td>
<td>Driving efficiency (Yuan per minute while occupied)</td>
</tr>
</tbody>
</table>
in the middle of a road while cruising. Specifically, if the taxi came to the current road \(l\) at time \(t\) from road \(i\), then \(A_{\text{allowed}}(l, t, d) = \{k | \text{Road}_l \cap \text{Road}_k \neq \emptyset \} \) and \(\text{Road}_l \cap \text{Road}_k = \emptyset \). Fig. 8 illustrates an example. In (a), there are 6 different incoming directions for road 0. In (b) there are 6 possible actions to take when the taxi is in road 0 (without considering the incoming direction constraint). (c) shows unallowable actions (red arrows) taken by a taxi at road 0, assuming the incoming direction (blue arrow) was road 5. (d) shows the allowed actions, where a taxi coming to the current state from left (5) can only go to the road (1, 2, 3).

4.3 State Transition and Objective Function

Assuming the current state of the taxi is \(s = (i, t, d)\). An action \(a\) is taken to move the taxi from road \(i\) to its adjacent road \(j\). As a result, the taxi will cruise to the next destination \(j\) and cruise the entire road \(j\) in \(t_{\text{seek}}(j)\) minutes. There will be two possible consequences.

1. The taxi did not find any passenger after \(t_{\text{seek}}(j)\) minutes in \(j\). Then the taxi must leave the current road and move to the next one. Assume the taxi took action \(a = \text{Road}_i \rightarrow \text{Road}_j\). Then the taxi will end up in the next state \(s' = (j, t + t_{\text{seek}}(j), \text{Road}_j)\).

2. The taxi successfully finds a passenger in road \(j\) after cruising the road for \(t_{\text{seek}}(j)\) minutes. In this case, the passenger may choose to go to one of the road as destination (denoted as \(k\)) at a probability \(p_{\text{dest}}(j, k)\). We will discuss how to obtain this probability later. Upon finishing this trip, the taxi will arrive at location \(k\). We use \(t_{\text{drive}}(j, k)\) to represent the total time needed to travel from \(j\) to \(k\). The driver will receive a fare of \(r(j, k)\) Yuan, where \(r(j, k)\) represents the expected fare between road \(j\) and \(k\). The taxi will start seeking from \(k\) again. The state of the taxi is thus transitioned to \(s' = (k, t + t_{\text{seek}}(j) + t_{\text{drive}}(j, k), 0)\).

In the above process, an important parameter is the probability that the taxi can find a passenger during the cruising of road \(j\), denoted as \(P_{\text{find}}(j)\). We will discuss how we estimate this parameter in the next section.

To summarize, a vacant taxi in any state \(s_0 = (i, t, d), s_0 \in S\) may take one of the possible actions \(a \in A_{\text{allowed}}(s_0)\) from road \(i\) to cruise to a connected road \(j\). With the probability \(1 - P_{\text{find}}(j)\) the taxi will transition to state \(s_1 = (j, t + t_{\text{seek}}(j), \text{Road}_j)\) with no reward. With the probability of \(P_{\text{find}}(j) \times P_{\text{dest}}(j, k), (k = 1, 2, \ldots, |L|)\), the taxi will end up in the next state \(s_2 = (k, t + t_{\text{seek}}(j) + t_{\text{drive}}(j, k), 0)\) and receive a reward of \(r(j, k)\) Yuan. The state transition diagram of the proposed MDP model is shown in Fig. 9.

4.2 Actions

In our model, each vacant taxi at one of the states has several possible actions to choose. Each action \((a)\) is to move from the current road to one of the roads connected to the current one, or stay in the current road. Similar to \(D\), we also use IDs \((1 - N(l))\) to index the actions. Fig. 8b illustrates the mapping of the actions. The arrow directions are the opposite compared to the incoming directions \((d)\).

In order to accurately capture the mobility pattern of taxis, we also include the following constraints. First of all, a taxi may not enter a road marked as invalid, as discussed in Section 4.1. Second, we also need to prevent a taxi from cruising the same road or a small number of roads in a loop. Fig. 8c illustrate such a scenario. To this end, we require that in a state \(s\), a taxi may only choose actions from a subset of \(A_{\text{allowed}}\), denoted as the allowed action set \(A_{\text{allowed}}(s)\). For a taxi that just dropped off a passenger, the arriving direction is 0 (no direction). The taxi may choose any action. After the taxi traversed to the end of the current road, it must turn to the next road. Also we do not allow taxis to make U-turns that in a state \(s\), the taxi may only choose actions from a subset of \(A_{\text{allowed}}\).

Formally, a state in our MDP model can be represented as \(s = (i, t, d)\). The maximum number of states in our problem setting is \(|L| \times |T| \times |D| = 821 \times 60 \times 7 = 344820\), where 7 is the maximum number of adjacent roads in the underlying spatial network. However, some of the states are invalid because the road numbers are not continuous.
circle represents a state with the three parameters listed beside it. Each road network represents several possible actions with only allowed actions highlighted. Arrows show state transitions. Squares represent branching of state transition. Due to space limit, this diagram just illustrates a small portion of MDP with one seeking action.

**Rolling Horizon.** In our previous work, the objective of the proposed grid-based MDP model was to maximize the total revenue within the remaining time of the current one-hour time window (e.g., 7-8 AM). A limitation of this model is that the parameters (e.g., seeking time, passenger pick up probability) are assumed to be static within each hour, which is an unrealistic assumption. Using fixed time slots also makes it hard to give recommendations we are close to the end of the current time slot.

To address this limitation, we employ a rolling horizon scheme. We assume that the parameters of the MDP change over time and can be updated constantly. The objective of the SN-MDP model is to maximize the total expected reward in the next $T_w$ minutes for every state. By default, $T_w$ is set to 60. For each time step $t_{move}$, the MDP treats states with $t = t_{move} + T_w$ as terminal states. For example, the terminal state for time windows beginning at time $t = 7:20$ AM would be those with time $= 8:20$ AM (as opposed to 8:00 AM in our prior work). Once the system reaches these states, no more actions can be taken.

The new MDP model can be viewed as a series of MDPs with fixed time window lengths. At each time step $t_{move}$, we solve an MDP model to find the best actions according to information available at this point. For every $a$ in this MDP, the $V^*(s,a)$ function represents the maximum expected profit in the remaining time before $t_{move} + T_w$. If action $a$ is taken at state $s$, $V(s)$ is the maximum expected profit for state $s$. Formally the objective can be expressed as follows:

$$V^*(s,a) = (1 - P_{find}(l)) \times V(l_t, t + t_{seek}(l_t), l) + \sum_{k=1}^{L} P_{find}(l_k) \times P_{dest}(l_k, k) \times (r(j, k) - C \times (t_{seek}(j) + t_{drive}(j, k)) + V(k, t + t_{seek}(j) + t_{drive}(j, k), 0),$$

where $s = (l, t, d)$ is a state and $a$ is an action that moves the taxi from road $l$ to road $l_t$, $C$ is the fuel cost per unit time. The optimal policy ($\pi$) based on parameters obtained at time $t_{move}$ is defined as follows:

$$\pi(s) = \arg \max_a \{V^*(s,a)\} \quad (5)$$

$$V(s) = V^*(s, \pi(s)). \quad (6)$$

The above model is updated for every time step $t_{move}$. The final optimal policy for the rolling-horizon MDP is the latest policy for each state.

### 4.4 Learning SN-MDP Parameters

Now we discuss how to decide the necessary parameters for the SN-MDP model. All the parameters are learned from our historical passenger trips and seeking trips.

**Learning the Passenger Pickup Probability $P_{find}$.** To calculate $P_{find}$, we use the following basic idea: total number of pick-ups divided by total number of pick-up attempts. Total number of pick-up attempts is the number of taxis that entered this road segment while being vacant. Formally

$$P_{find}(l) = \frac{n_{find}(l)}{n_{find}(l) + n_{pass}(l)} \quad (7)$$

Since the data only contains the origin and destination of each trip, we have to estimate the route taken by each taxi when seeking for a passenger. For each seeking trip, we use the API provided by Baidu Map to calculate the shortest path between the origin (the previous drop-off) and the destination (the next pick-up). Then we map each estimated seeking trip obtained to corresponding roads and count how many times each road is passed by a vacant taxi during each time slot.

Previously, we simply calculated $n_{find}$ and $n_{pass}$ as the historical average of pickups and passing vacant taxis for each time slot. Also we assumed that these two parameters are constant within each hourly time slot. However, this method may result in the following issues: (1) Temporal non-stationarity of urban trips may result in significant over-estimation of $P_{find}$. Consider a road segment where historically many pick-up events occur that take the taxis to other attractive locations. If many taxi drivers adopt our recommendations, these locations will have much lower $P_{find}$ than before. Recommending more taxis to seek the same location for passengers will adversely affect both the earnings of the drivers and waiting time of the passengers. (2) Equation (7) generates a fixed pickup probability within each time slot, which changes abruptly at the end of the slot and results in inconsistent probabilities for the same location during a very short period of time. (3) Taxis in the same states might be recommended to the same routes, leading to unnecessary competitions. We discuss our design decisions to address the above three issues.

**Addressing Issue 1 (Time-Variant $P_{find}$).** We propose that $P_{find}$ consists of two parts, The probability calculated from historical data ($P_{history}$) and the probability calculated from the data of current day ($P_{immediate}$). $P_{immediate}$ reflects the specific circumstances of the current day. This way, we solve the issue by accounting for the temporal non-stationarity of the pick-up probabilities by combining data from historical and recent pick-ups, which reflect the recent behavior of the passengers and the taxis.

**Addressing Issue 2 (Abrupt $P_{find}$ Change).** We propose a sliding window method with empirical distributions. To calculate $P_{history}$ for time $t_{move}$ we only use the number of historical passenger pickups and the number of vacant taxis in the time window $[t_{move}, t_{move} + T_w]$. Suppose $X(l)$ is a random variable representing the number of passenger pickups at the current road $l$ within the above time window in a day, and $Y(l)$ is a random variable representing the number of all passing vacant taxis at the same time of day at location $l$. We calculate the empirical distribution of $X(l)$ and $Y(l)$ among all the days over one year. The pickup probability when $X(l) = X_j^l$ and $Y = Y_j^l$ is $\frac{X_j^l}{Y_j^l}$. With $Y_j^l = 0$ the probability is set to 0. We consider all the possible $X_j^l$ and $Y_j^l$ and their empirical probabilities when calculating the overall pickup probability. Formally it is calculated as shown.
When the pick-up was observed on an example road segment.

Fig. 11 shows $P_{\text{history}}$ in each area for four time slots (best viewed in color).

Fig. 12. Comparison of pickup probability computed by our prior work and SN-MDP.

(a) Empirical Distribution of Successful Pickups
(b) Empirical Distribution of Number of Vacant Taxis

Fig. 10. Distribution of successful pick-ups ($P_X(\cdot)$) and passing taxi ($P_Y(\cdot)$) on road $l = 523$.

$$P_{\text{history}}(l) = \frac{x_1}{y_1} \times P_{X_1}(\cdot) \times P_{Y_1}(\cdot) + \frac{x_1}{y_2} \times P_{X_2}(\cdot) \times P_{Y_2}(\cdot)$$
$$+ \cdots + \frac{x_m}{y_1} \times P_{X_m}(\cdot) \times P_{Y_m}(\cdot) + \frac{x_m}{y_2} \times P_{X_m}(\cdot) \times P_{Y_m}(\cdot)$$
$$+ \cdots + \frac{x_m}{y_2} \times P_{X_m}(\cdot) \times P_{Y_m}(\cdot) + \frac{x_m}{y_2} \times P_{X_m}(\cdot) \times P_{Y_m}(\cdot)$$
$$+ \cdots + \frac{x_m}{y_2} \times P_{X_m}(\cdot) \times P_{Y_m}(\cdot) = \sum_{i=1}^{m} \left( \frac{x_i}{y_1} \times P_{X_i}(\cdot) \times P_{Y_i}(\cdot) \right).$$

Equation (4) solves the overestimation problem of Equation (7). This is illustrated in the following example: Suppose at location $l$ we had 1 pick-up only on one day throughout the entire year while only 1 vacant taxi passed through $l$. This means $X^1 = 1$ and $Y^1 = 1$ on the day of pick-up and 0 on all other days. In this case, Equation (7) calculates the pick-up probability as $\hat{p} = 100\%$, which is a significant over-estimation. However, Equation (8) will result in a much lower value. Because all the terms, except the term corresponding to the day the pick-up was observed on, will be zero and $P_X(X^1 = 1) = \frac{1}{365}$ and $P_Y(Y^1 = 2) = \frac{1}{100}$, thus resulting in a reasonably low value. This way, we address the issue of substantial overestimation by Equation (7). Fig. 10 shows the empirical distributions of $X(l)$ and $Y(l)$ on an example road segment.

Fig. 11 shows $P_{\text{history}}$ heat map with $t_{\text{move}} = 6$ AM, 9 AM, 12 PM, and 3 PM, respectively, and $T_0 = 60$. As can be observed, downtown areas with high population density and pickup counts generally have higher pickup probability. This is consistent with common sense and real situations.

Fig. 12 compares the pickup probability calculated by Equation (7) in our prior work (blue) and Equation (8) (red) in SN-MDP with $t_{\text{move}} = 9$ AM and 12 PM, respectively. Both probabilities show positive correlations with pickup counts. However, Equation (8) shows less noise and stronger correlation. The number of roads with low pickup count but high pickup probability is significantly reduced. Roads with very high pickup counts may have lower pickup probability than roads with moderately high pickup counts due to taxi competitions. These results show that our newly proposed method (Equation (8)) generates more realistic estimations of the parameter $P_{\text{history}}$.

To calculate $P_{\text{history}}$ at location $l$ and time $t$, we use total number of successful pickups at road $l$ before the time $t$ on that day divided by the total number of all the vacant passing taxis at time $t$ on that day. The $P_{\text{immediate}}$ is thus calculated as follows:

$$P_{\text{immediate}}(l, t) = \frac{n_{\text{find}}(l, t)}{n_{\text{find}}(l, t) + n_{\text{pass}}(l, t)}. \quad (9)$$

Where $n_{\text{find}}(l, t)$ and $n_{\text{pass}}(l, t)$ denote the total number of successful pickups before the time $t$ at location $l$ that day and the total number of all the vacant taxis passing before the time $t$ at location $l$ on that day, respectively.

**Addressing Issue 3 (Competing Taxis).** We also need to consider the competition between taxis. If there are multiple taxis at the same location that need recommendations during a short time interval, then we need to attenuate the probability of finding passengers on each adjacent road based on the competition between taxis. Here we employ an online adjustment strategy, where the probability of finding a passenger on a specific road is a linear combination of the historical probability and the recent probability. The recent probability ($P_{\text{immediate}}$) is discounted when more taxis are cruising on this road. For example, the probability of finding a passenger on a road is $P$. If $n$ taxis have already been recommended for the road, the probability of the road is updated to $P/n$ when the $n + 1$th taxi is recommended. In the end, the new $P_{\text{find}}$ is calculated as follows:

$$P_{\text{find}}(l, t) = \frac{1}{n} \cdot (f(t) P_{\text{history}}(l) + (1 - f(t)) P_{\text{immediate}}(l, t)). \quad (10)$$

Where $n$ indicates the number of competing taxis on road $l$ at time $t$, $f(t) = e^{\omega(1-t)}$ is a function that represents weight of each of the historical and/or the immediate probabilities,
and ranges between 0 and 1 while decreasing with time. $u$ is a coefficient that controls the convergence rate. As time goes on, the weight of $P_{\text{history}}$ gradually decreases and the weight of $P_{\text{immediate}}$ increases, because the trend of the current day becomes more obvious and therefore given more weights. The above updates occur in every single time unit ($t$). In our problem settings, the frequency is every minute.

It should be noted that $P_{\text{find}}$ should not be attenuated indefinitely. For every RT minutes, it should be restored to the initial value (that is, the value of $\frac{1}{u}$ is removed from the Equation (10)). We evaluate the impact of different RT choices in the Evaluation section.

Learning Passenger Destination Probability $P_{\text{dest}}$. To estimate the destination probability for each time window \(t_{\text{move}} \times T_w + T_w\), we calculate a matrix $W$, where each element $W_{ij}$ is the number of trips between each pair of source $i$ and destination $j$ in that time window. We normalize $W$ in each row (each value divided by the sum of the entire row, except for zeros). In the resulting matrix $P_{\text{dest}}$, each row $i$ has the empirical probability distribution of a passenger choosing destination $j$ when picked-up on road $i$. Fig. 13 shows the heat maps of destination probability distribution of four day-time slots. As can be observed, high probabilities clustered along diagonal of the matrix. Since we number nearby roads sequentially, the results suggest that most trips are short.

Estimating the Reward Function $r(i,j)$. In our model, the reward $(r)$ is the fare income for a trip. We simply calculate the average fare of trips between each pair of source and destinations at the same time slot as the expected fare

$$r(i,j) = \frac{\text{Total trip fare between (i,j)}}{\text{Total number of trips between (i,j)}}.$$ \hspace{1cm} (11)

Estimating the Driving Time $t_{\text{drive}}$. We use average driving time. $t_{\text{drive}}(i,j)$ means average driving time between a pick up on road $i$ and a drop off on road $j$, as shown in Equation (12). We calculate this parameter for every time step $t_{\text{move}}$.

$$t_{\text{drive}}(i,j) = \frac{\text{Total driving time between (i,j)}}{\text{Total number of trips between (i,j)}}.$$ \hspace{1cm} (12)

4.5 Solving the SN-MDP Model

For each time step $t_{\text{move}}$, we build an MDP with parameters learned from our data. Then the MDP is solved using a dynamic programming approach.

For each state in a specific MDP, the result is the best action to take for that state to maximize total expected rewards in the remaining time of the current time slot $[t_{\text{move}}, t_{\text{move}} + T_w]$. Once the system reaches a state with $t = t_{\text{move}} + T_w$ it terminates. The pseudo code of the algorithm is presented in Algorithm 1. The algorithm iterates for every $t_{\text{move}}$ of day and every $t_{\text{move}}$ will only output a unique recommendation. However, when calculating the maximum expected reward at $t_{\text{move}}$, we need to consider the reward at each time after $t_{\text{move}}$. Therefore, for each $t_{\text{move}}$ algorithm iterates from the state $t = t_{\text{move}} + T_w$ back to the state $t_{\text{move}}$ and finds the maximum expected reward. Thus, the optimal policy at $t_{\text{move}}$ is calculated. For each state $s$, the algorithm examines all the possible actions and calculates the maximum expected reward (Lines 4-6). The total expected reward of state $s$ after taking action $a$, $V^*(s, a)$, is calculated using Equation (4) (Line 7). Here $V(l_s, t + t_{\text{seek}}(l_s, a')$ and $V(k, t + t_{\text{seek}}(j) + t_{\text{drive}}(j, k), 0)$ must have been calculated already since they have larger $t$ value than $s$. Then the action with the maximum $V^*$ for $s$ (denoted as $a_{\text{max}}$) is selected (Line 8). The maximum expected reward of $s$ is thus set to $V^*(s, a_{\text{max}})$ (Line 9). The output of this algorithm is one action for each state.

\textbf{Algorithm 1. Solving MDP using Dynamic Programming}

\textbf{Input}: $L, A, D, T_w, T_{\text{day}}$
\textbf{Output}: The set of best policy $\Pi$

1: \textbf{for} $t_{\text{move}} = 1$ to $T_{\text{day}}$ \textbf{do}

2: \quad Get newest parameter $(P_{\text{find}}, P_{\text{dest}}, R, t_{\text{seek}})$

3: \quad \text{according to } $t_{\text{move}}$

4: \quad $V$ is a $[L] \times [T]$ matrix; $V \leftarrow 0$

5: \quad \text{for } $t = t_{\text{move}} + T_w$ \text{to } $t_{\text{move}}$ \text{do}

6: \quad \text{for } $l = 1$ to $|L|$ \text{do}

7: \quad \quad $a_{\text{max}} \leftarrow a$ that maximizes $V^*(s, a)$

8: \quad \quad $s = (l, t, d)$

9: \quad \quad $\pi(s) \leftarrow a_{\text{max}}$

10: \quad \quad $V(s) \leftarrow V^*(s, a_{\text{max}})$

11: \quad \text{end}

12: \text{end}

13: \text{add } $\pi$ to $\Pi$

14: \text{end}

15: \text{Return } $\Pi$

The algorithm has time complexity of $O(|T_w| \times |D| \times |L| \times |A| \times |T_{\text{day}}|)$. $|T_{\text{day}}|$ is the minutes of daytime we used, and we set it as 540. Since $|D|$ and $|A|$ are small constant numbers, the complexity can be simplified to $O(|T_w|)$.
The space complexity is $O(|L| \cdot |T_{day}|)$. Note in this algorithm, the total time cost increases with the number of states (i.e., number of roads, time lengths). However, the time cost does not increase when there are more taxis since in the same minute, recommendations for all the taxis in the same state are the same. If there are multiple taxis needing recommendations in a short time period, we control the competition by dynamically adjusting the $P_{find}$ parameter of each road, as detailed in Section 4.4.

5 EVALUATION

In this section, we perform a number of experiments to evaluate the quality of our results. Specifically, we hope to answer the following questions: (1) How much more profit could be generated if a driver used our recommended driving directions? (2) Is our proposed method better than methods in related work and the Grid-Based Model in our prior work? (3) How will the profit efficiency and computational cost change with different parameter settings?

In order to calculate the profit efficiency, we deduct the fuel cost from the earned revenue. According to the official records of the manufacturer of the taxi vehicles in our data, the fuel consumption is 6.1 liters per 100 km. Because the taxis operate in urban areas, we add 20 percent, in the end, fuel consumption is 7.3 liters per 100 km. And based on historical data, we estimated that the taxi runs about 2.7 minutes per kilometer. At that time the price of fuel was 7.5 yuan per liter. Therefore, we can calculate the fuel cost to be about 0.2 yuan per minute. Therefore, the profit efficiency can be calculated. Parameter $u$ of $f(t)$ of Equation (10) is set to 0.015. This value gave the best results after performing some tuning.

5.1 Simulation Setup

We use the data from March 2012 to February 2013 as a training data set to learn the necessary parameters of our SN-MDP model. To better address the variations of parameters over time, we separate weekends and weekdays. We use another three months’ data from March to May in 2013 as a validation data set.

For the validation dataset, we simulate the behaviors of passengers based on the real data. Specifically, if there is a pick up event in the real data, then in the simulation we assume that the corresponding passenger shows up before the real pick up time at the same location. The waiting time for each passenger before pickup is a random variable between 0 and $t_{max}$ minutes, with a uniform distribution. We will discuss how $t_{max}$ affect the performance of our algorithm later.

The initial location of a simulated taxi is the same as its first drop off location in the real data during the simulation time period. Then based on the MDP results we obtained from the training data, we recommend the next move to the taxi. After the taxi arrived at a new road segment, it will pick up one of the waiting passenger in the same location. The probability of a pick-up for a seeking taxi in a road segment is the number of waiting passengers divided by the number of seeking taxis. If the number of passengers is higher, the probability is set to 1. If the passenger is not found, the taxi receives a recommendation for the next location. If a passenger is found, the taxi travels to the destination in the dataset, which is specified by the passenger. For every trip that the taxi makes, the fare and the cost are added to the earned profit. Also, the travel time to the destination will be added to the simulated clock time. This process is simulated until the time reaches the end of taxi’s shift. This constitutes one simulation trial. We repeat the simulation for every taxi in the same time period and evaluate the average performance of our proposed model.

Next, we discuss how to set the recovery interval length $RT$ for the probability of finding passengers ($P_{find}$). We test different values of $RT$, and then for each of them we run our simulation and measure the final average profit efficiency of all the taxi drivers. The results are shown in Fig. 14. As can be observed, when $RT = 8$, the profit efficiency is optimal. Therefore we set $RT$ to 8 minutes in all the following experiments.

5.2 Profit Improvement over Real Data

In order to verify the effectiveness of our recommendation, we carry out simulations by providing recommendation to the top 10 percent and bottom 10 percent taxis. These taxis are chosen based on their driving efficiency, defined in Equation (3). Since this is based on historical data to simulate, so we only use $P_{history}$ to calculate the pickup probability. We compare the performance of these taxis when receiving recommendations with their actual performance. We simulate the driving of taxis from 7 AM to 3 PM for every day in the validation dataset. Then we calculate the average profit efficiency of drivers.

Fig. 15 shows the profit efficiency for the top 10 percent with and without recommendation in two time slots: 12-13 and 13-14. Figs. 15a and 15b show the profit efficiency distribution of the top 10 percent drivers (left, green) and our simulated results using the top 10 percent drivers’ driving efficiency (right, red) for time slot 12-13. The average profit efficiency for 12-13 is improved from 0.722 yuan/min to 0.766 yuan/min, yielding a 6.1 percent improvement. Figs. 15c and 15d show that the profit efficiency for 13-14 is improved from 0.745 to 0.776, yielding a 4.1 percent improvement.

Figs. 16a and 16b show the profit efficiency distribution of the bottom 10 percent drivers (left, yellow) and our simulated results using the bottom 10 percent drivers’ driving efficiency (right, red) for the same time slot. The average profit efficiency is improved from 0.658 to 0.715 yuan/min, yielding a 8.7 percent improvement. We also repeat the same evaluation over time slot 13-14 and achieved an improvement of 7.2 percent, as shown in Figs. 16c and 16d.

Finally, we perform another simulation recommending pick-up strategies only to the bottom 10 percent of the taxis.
drivers to compare them with the top performing drivers. The average simulated profit efficiency for each time slot are compared with the average profit efficiency of the top 10 percent drivers in real data. Fig. 17c shows the results. The simulated profit efficiency (red line) are close to the top 10 percent drivers’ profit efficiency for almost all the time slots. This result validates the effectiveness of our proposed method—the bottom drivers with limited driving efficiency can still make the same amount of money as the top drivers if they follow our driving strategy.

5.3 Profit Efficiency Improvement over Baselines

We compare the profit efficiency achieved by our method against the profit efficiency achieved by three baseline methods: (1) A greedy heuristic algorithm (Greedy), which recommends the adjacent road with the most pick-ups in the historic data in each recommendation. (2) The Grid-Based MDP model proposed in our prior paper [1] (Grid-MDP). (3) the MNP algorithm (MNP) in related work [4], which recommends the next five road segments to seek passengers in each recommendation. The objective of the MNP algorithm is to maximize the total expected net profit along the seeking trip. If no passenger is found, the algorithm will recommend another five road segments based on the new location of the taxi.

We perform simulations on our testing dataset and evaluate the cumulative profit efficiency over time. We use Equation (10) to calculate the pick-up probabilities. The cumulative profit efficiency is calculated as the total profit divided by the total business time of a taxi, starting from 7 AM. To account for the differences in the travel patterns between weekdays and weekends, we calculate parameters for SN-MDP based on weekdays and weekends separately.

Fig. 18a shows the cumulative profit efficiency comparison between Grid-MDP and SN-MDP, where a 9.4 percent increase is achieved. Fig. 18b shows the comparison between SN-MDP, MNP and greedy heuristic algorithm. The profit efficiency of SN-MDP is 9.22 and 9.31 percent over that of the MNP and the greedy heuristic of all the days. Figs. 18c and 18d show a more detailed breakdown of cumulative profit efficiency on workdays and during weekends. Compared with MNP and the greedy heuristic, SN-MDP achieves about 7.68 and 13.89 percent improvement in weekdays and 12.68 percent improvement in weekends, respectively.

In addition, we compare the recommended routes by our method and by the baseline solution. Examples in Figs. 19a and 19b show that our recommended routes (red arrows) go to areas with high pickup probability, while the MNP method (yellow arrows) sometimes recommend the taxi to follow a path that will end up in areas with low pick-up probability (residential or suburbs). As a result longer seeking trips might occur in the future.
5.4 Simulation for Multi-Taxi Recommendation

Next, we verify the effectiveness of our method in the case of recommending routes for multiple taxis in a short period of time. We randomly selected two scenes from our simulation, as shown in Fig. 20.

In the above two cases, three taxis appeared within a short time on the same road, and our recommended method will recommend different routes for them. It should be noted that the routes we recommend for them are not necessarily inconsistent. In fact, if the $P_{\text{find}}$ of a road is large, its ability to withstand taxi competition is strong. Even if $P_{\text{find}}$ decreases with the number of taxis, it is still larger than the ones of other roads. Therefore, such a road can be recommended repeatedly. This is also in line with the actual.

The above results show that our method can be applied to scenarios where multiple taxis are simultaneously recommended.

5.5 Maximum Waiting Time

One assumption we make in SN-MDP is the maximum waiting time of a passenger. In order to find the effect of different waiting times on profit efficiency, we set maximum waiting time from 5 minutes to 25 minutes, and get the relationship as shown in Fig. 21. Based on the result of this experiment, we observe that the longer the passenger is willing to wait, the higher the profit efficiency can be achieved. But when the maximum waiting time is over 15 minutes, the improvement of profit efficiency becomes marginal.

5.6 Computational Efficiency

Finally we evaluate the computational efficiency of the proposed method and the impact of input parameter values. As illustrated in Section 4.5, the time cost to solve the MDP and make recommendation depends on the number of unique states rather than the number taxis, because for all the taxis that are in the same states at the same time, the recommendations are the same. Even for competing taxis, we update all the parameters every minute instead of for every taxi. Therefore, the number of taxis will have very slight impact on the running time.

In this experiment, we first change the number of road segments in the input with $|T_w| = 60$. A sub-region with $k$ roads in the center of the current study area is selected as input, where $k$ increases from 270 to 810. Fig. 22a shows the trend, where the computation time is nearly linear to the number of roads on each side of the study area. Second, we change the length of the time slots $T_w$ from 60 minutes to 180 minutes with $L = 821$. Fig. 22b shows the trend, where the computation time is linear to the length of time window. This shows that the proposed method is scalable to larger datasets. The run-time for 3 hours is less than one second. Since the recommendations are made every minute, this time cost is totally acceptable for a real-time system.
6 RELATED WORK

Prior literature related to our work can be classified into three categories.

The first category of papers are the most relevant to this paper. They learn seeking strategies from historical data and recommend seeking locations or routes for drivers to improve energy efficiency and profit or reducing travel distance and seeking time before the next trip. The work by Qu et al. [4] recommends the most profitable seeking route for a driver looking for the next passenger. Given the current location of a taxi driver, the proposed MNP algorithm will recommend a sequence of five consecutive road segments for the taxi. This method is used in our Evaluation part as a baseline method. However, they do not consider long-term profit. Rather, they maximize the profit of the immediate next trip by suggesting a path with five roads. Yuan et al. [7] proposed an approach to detect parking places based on a large number of GPS trajectories generated by taxis and devised a probabilistic model to formulate the time-dependent taxi behaviors (picking-up/dropping-off/cruising/parking) and enable a city-wide recommendation system for both taxi drivers and passengers. Yuan et al. built a model to recommend routes to parking place for taxi drivers, which could maximize the expected income of the next trip [3]. They used detailed GPS points to detect parking places and focused on the routes to these places. In our work we do not consider parking and recommend connected seeking paths with no stops. The two problems have different settings. Ge et al. [2] proposed a computational approach to identify routes that minimize the potential travel distance before picking up the next passenger. The ideas are similar to our baseline method in [4]. Zhang et al. [5] predicted the revenue of taxi drivers based on their strategies and achieve a prediction residual as less as 2.35 RMB/h. This work focused on analyzing the successfulness of given driving strategies rather than making new recommendations. Huang et al. [8] presented a dynamic programming method to solve the high computational complexity of mobile sequential recommendation, which greatly improves the pruning effect of the algorithms in [2]. Gan et al. [11] propose two novel algorithms—FLORA and FLORA-A—to address the inadequacies of related work. Using convex polytope representation techniques, FLORA provides a fully compact representation for taxi drivers’ strategy space and scales up more efficiently than existing algorithms. Chen et al. [12] proposes an app running on drivers’ smartphones that recommends detour routes in order to optimize the taxiing performance. They formulate the problem to maximize the profit of a driver with constraints on charging the passenger according to the normal route, and saving the taxiing time.

As illustrated in the Introduction, these work focused on optimizing measure of the immediate next trip but ignored the future revenue since they didn’t consider the impact of passenger destination choice in their decision process. This is the most important difference between our work and these related work.

The second category of work learns knowledge from taxi data for other types of recommendation scenarios or general knowledge discover, such as mapping, fast routing, ride-sharing, or fair-recommendation. Powell et al. [13] mined historical taxi GPS trajectory data to generate Spatio-Temporal Profitability (STP) maps according to the potential profit calculated by the historical data. However they only show the profitability maps instead of giving specific driving directions. Wang et al. [14] propose TaxiRec, a framework for discovering the passenger-finding potentials of road clusters. They build a ranking-based extreme learning machine (ELM) model to evaluate the passenger-finding potential of each road cluster. Yuan et al. [15] propose “T-Drive”, a data-driven recommendation system to suggest fast routes based on taxi drivers’ historical driving records. Castro et al. [16] propose a traffic density model by using a large database of taxi GPS trajectory data to analyze each traffic capacity of road in the city, and combined with the future traffic conditions to give accurate forecast. Chawla et al. [17] propose a two-step method to deduce the anomalies of traffic. Miao et al. [10] proposed a receding horizon control approach of taxi dispatch with real-time sensing data in metropolitan areas. Ma et al. [18], [19] propose a large-scale taxi ride sharing service, which handles real-time requests sent by taxi users and generates ride sharing schedules that reduce the total travel distance significantly. Qian et al. [20] design a sharing considered route assignment mechanism for fair taxi route recommendations, which provides recommendation fairness without sacrificing driving efficiency. Chiang et al. [21] propose a Grid-based Gaussian Mixture Model (GGMM) with spatio-temporal dimensions that groups booking data into a number of spatio-temporal clusters. Further, they apply GGMM to detect anomalous bookings. Xu et al. [22] first investigate the correlation between drivers’ skills and their mutual interactions in the latent vehicle-to-vehicle network and develop a two-stage framework for quantitatively revealing the latent driving pattern propagation within taxi drivers. Han et al. [23] demonstrate that a reinforcement learning algorithm of the Q-learning family, based on a customized exploration and exploitation strategy, is able to learn optimal actions for the routing autonomous taxis in a real scenario. Phiboonbanakit et al. [24] calculate taxi cost using a cost-distance algorithm.

None of these work suggest seeking routes for taxis. They provide suggestions or knowledge for other types of services. Therefore they are not directly related to our problem.

The third category of papers learn knowledge from taxi traces for urban planning and social functions such as urban flow, land use, or prediction of urban human mobility. Zheng et al. [6], [25], [26] introduce urban computing approaches based on trajectory data, which aim to detect flawed and less effective urban planning settings in a city. Vevelo [27] gather taxi GPS data in Lisbon, Portugal and render the cab trajectory data in time and space distribution, and discusses the taxi
operation strategy and the corresponding income levels. Zhang et al. [28] designed a system for real-time sensing of refueling behavior and citywide petrol consumption. Castro et al. [29] propose a traffic density model by using a large database of taxi GPS trajectory data to analyze each traffic capacity of road in the city, and make forecasts. Zhang et al. [30] use human flow dynamics to detect social events and measure their impacts. Li Jin et al. [31] model the urban black holes in each region of New York City (NYC) at different time intervals with a 3-dimensional tensor by fusing cross-domain data sources. Seong et al. [32] describe a proof of concept effort to explore the weaknesses and possible improvements in public transportation systems through mining taxi ride dataset. Vahedian et al. [33], [34] use taxi trajectory data to detect urban gathering events.

Although these work all use taxi trajectory data, the topics of these work are not directly related to our work in this paper.

7 CONCLUSION

This paper investigated the recommendation of the best taxi seeking strategy based on historical data to improve taxi drivers’ business efficiency. This problem is of great societal importance since it helps improve drivers’ income and reduce green house gas emission. However, this task is also challenging due to uncertainty from both the taxi side and passenger side and the large number of possible scenarios. Related work on taxi driving strategy recommendation focused on improving the pick-up chance, profit, or energy efficiency of the immediate next trip, ignoring the impact from the passenger’s destination choice on future profit. Our recent paper proposed a Grid-based Markov Decision Process approach, where the study region is partitioned into grids. Given a set of historical taxi records and the current status (e.g., grid and time) of a vacant taxi, our prior work found the best move for this taxi to maximize the revenue in the remaining time of the current hourly slot. This paper extended our prior work by proposing a novel Spatial Network-based Markov Decision Process (SN-MDP) with a rolling horizon configuration to recommend better driving directions. We also proposed a new statistical model to estimate the necessary parameters (e.g., pickup probability) from the data. In addition, we took fuel cost into consideration, to maximize the profit, rather than only the income. A case study and several experimental evaluations on a real taxi dataset from a major city in China showed that our proposed approach improves the profit efficiency by up to 13.7 percent and outperforms the two baseline methods as well as our previous approach in all the time slots.

ACKNOWLEDGMENTS

Xun Zhou and Amin Vahedian K. were supported in part by US National Science Foundation under Grant IIS-1566386. Huigui Rong, Qun Zhang and Chang Yang were supported in part by the National Natural Science Foundation of China under Grant No.61672221, No.61273232 and No.61304184, Hunan Key Research and Development Program under Grant 2017GK2272, and Hunan Province Natural Science Foundation under Grant 2018JJ3259. We also acknowledge the support of the Obermenn Center for Advanced Studies at the University of Iowa. An earlier version of this work [1] appeared in Proc. ACM CIKM 2016.

REFERENCES

Xun Zhou received the BE and MS degrees in computer science and technology from the Harbin Institute of Technology, China, in 2007 and 2009, respectively, and the PhD degree in computer science from the University of Minnesota, Twin Cities, in 2014. He is currently an assistant professor with the Department of Management Sciences, University of Iowa. His research interests include big data management and analytics, spatial and spatio-temporal data mining, and Geographic Information Systems (GIS). He has published more than 40 papers in these areas and has received three best paper awards. He also co-edited Springer’s Encyclopedia of GIS, 2nd Edition.

Huigui Rong received the PhD degree in information management from Wuhan University, China, in 2010. He was a system analyst and senior project manager certificated by the Ministry of Industry and Information Technology (MIIT), and he was a visiting scholar with the Computer Science Department, Michigan State University (MSU) from 2014 to 2015. He is currently a full assistant professor of computer science and technology with Hunan University. His research interests include data mining and machine learning. He has published more than 20 research papers in these areas.

Chang Yang received the master’s degree in software engineering from Hunan University, Changsha, in 2017. He is currently a senior postgraduate student at Shenzhen City. He received three first-class scholarships and a national scholarship in stage of postgraduate. His research interests include data mining, edge computing, and Internet of Things. He has published articles in ACM conference proceedings and IEEE Journals.

Qun Zhang is currently working toward the graduate degree in the Department of Computer Science, Hunan University. He is a CCF member and received first-class scholarships in stage of postgraduate. His main research interests include machine learning, spatiotemporal data mining, and big data analysis.

Amin Vahedian Khezerlou received the bachelor’s and master’s degrees in information technology. He is currently working toward the PhD degree in management sciences in the Tippie College of Business, University of Iowa. His research interests include big data analytics and spatial and spatio-temporal data mining. His thesis focuses on mining big urban mobility data to develop techniques and frameworks that facilitate urban sustainability and growth through advanced data analytics. He has published articles in ACM Transactions and ACM conference proceedings.

Hui Zheng received the master’s degree from Central South University, China, in 2009. She is currently a full lecturer of the Tourism Management Department, Hunan University of Commerce. Her major research areas include tourism data mining and electronic business. She has published more than 10 research papers in these areas.

Zubiar Shafiq received the BE degree in electrical engineering from the National University of Sciences and Technology (NUST) Pakistan, for which he was awarded Dean’s Plaque for Excellence in Undergraduate Research, and the PhD degree in computer science from Michigan State University, for which he was awarded the Fitch Beach Outstanding Graduate Research Award. He is an assistant professor of computer science with the University of Iowa. He is a recipient of Best Paper Awards at the 2017 ACM Internet Measurement Conference (IMC) and the 2012 IEEE International Conference on Network Protocols (ICNP) as well as the Andreas Pfitzmann Best Student Paper Award at the 2018 Privacy Enhancing Technologies Symposium (PETS). He is also a recipient of the NSF CAREER Award. His research focuses on measurement and modeling of networking, security, and privacy issues.

Alex X. Liu received the PhD degree in computer science from the University of Texas at Austin, in 2006, and is a professor with the Department of Computer Science and Engineering, Michigan State University. He received the IEEE & IFIP William C. Carter Award in 2004, a National Science Foundation CAREER award in 2009, and the Michigan State University Withrow Distinguished Scholar Award in 2011. He has served as an editor of the IEEE/ACM Transactions on Networking, and he is currently an associate editor of the IEEE Transactions on Dependable and Secure Computing and the IEEE Transactions on Mobile Computing, and an area editor of the Computer Communications. He has served as the TPC co-chair for ICNP 2014 and IFIP Networking 2019. He received Best Paper Awards from ICNP-2012, SRDS-2012, and LISA-2010. His research interests focus on networking and security.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.