# Reinforcement Learning for Taxi Driver Re-positioning Problem in NYC

Tian Wang†, Yingyu Cao†, Bo Jumrustanasan†, Tianyi Wang†, Xue Xia†, Vineet Goyal†,
Zhiwei Qin‡, Shuaiji Li‡, and Qun Li‡
†Columbia University, ‡Didi Chuxing
{tian.w, yc3713, pj2356, tw2731, xx2338, vg2277}@columbia.edu
{qinzhiwei, shuaijili, liquntracy}@didiglobal.com

## I. Introduction

With the rise of for-hire vehicle (FHV) companies such as Uber, Lyft, Juno, and Via, the transportation system and taxi section is heavily impacted in New York City (NYC). One of the changes is the ever-worsening traffic congestion. From 2010 to 2018, the average weekday travel speed in Midtown Manhattan has dropped from 6.1 mph to 4.3 mph, which is close to walking pace. Another change is the excess supply. As a result, drivers need to spend $41\%$ of their time on cruising, which means driving around the city without a passenger [1]. The fierce competition, under-utilization and the declined traffic speed hinder the region's economy, worsen air quality, and enlarge the gap of drivers' income. The challenges for improving traffic efficiency, supply-demand balance, and drivers' income require the platform and taxi drivers to allocate their time wisely.

Taxi drivers' income varies depending on their working behaviors and preferences. Two drivers do not receive the same daily earning, even though they work on the same day for the same number of hours, due to the dynamic demand and supply of yellow taxis in the areas they have traveled. In this project, we want to disclose key working behaviors and preferences that account for the significant income difference among NYC yellow taxi drivers. The behaviors and preferences include when and where to start the shift and where to cruise to in order to maximize a driver's income (and this is hereby defined as re-positioning problem). The finding will play an important role in guiding drivers' working decisions on a ride-hailing platform, as well as improving the efficiency of the platform.

Our approach to the problem is guided by existing thriving literature applying reinforcement learning. We aim to find the best spatially and temporally varied policies that maximize drivers' daily earning using SARSA.

The paper is organized as follows: In Section II, we discuss how previous studies tackle this problem. Then in Section III, we describe the NYC dataset we use and the data cleaning process. Next, we provide exploratory data analysis in Section IV and our estimation using the given dataset in Section V. We set up our reinforcement learning model in Section VI, and present our results of estimation in Section VII. After that, we conclude the report and discuss some future research directions in Section VIII. Lastly, we include the acknowledgement, individual contribution, ethical concerns and GitHub link for this project in Section IX.

## II. Related Work

As taxi GPS trajectories data become widely available and applying reinforcement learning models to optimizing drivers' decision-making process becomes popular, in recent years, many studies have employed Markov Decision Process (MDP) to optimize efficiency in driver's searching strategy. The following serves as a summary of previous research and gives a guideline on potential directions of our project.

The first major dimension of difference in research is the model mechanism. With the rise of E-hailing platforms (such as Lyft and Didi), studies on drivers' searching strategy in online platforms have been conducted [2] [3], in contrast to previous work focusing on traditional taxi drivers (especially yellow cab drivers in NYC) [4]. The decision process for drivers in these two types of business models differs. Traditionally, when a cruising driver sees a passenger, the driver will start a ride and the searching process ends. In E-hailing platforms, there's a centralized platform deciding the customer-passenger match. Even though a driver sees a passenger, as long as the platform doesn't match them, the driver cannot stop searching for the next ride. However, drivers can decide to re-position themselves to other areas to maximize their probability of getting the next valuable ride.

This gives rise to the second dimension of major difference. As the platform is capable of managing the entire fleet, studies are conducted from a platform-centric perspective, optimizing the overall efficiency of the platform [5] [6], whereas other studies take a driver-centric perspective as each driver aims at maximizing their own reward function [7].

Finally, the third major difference is reinforcement learning algorithms. Some studies choose model-based algorithms where the complete environment dynamics have to be known before the learning process. In this setting, Dynamic Programming is widely used [8]. The major drawbacks are that: 1. as all parameters regarding the environment have to be known, they need to be estimated beforehand and therefore these models don't scale well; 2. it's hardly applicable to real-world networks. Other studies choose model-free algorithms. In this setting, Monte Carlo [9], Temporal-Difference [10], Deep Q Learning [6] are widely used.

## III. Data

### A. Data Schema

The data that we used is publicly available: NYC Taxi Trip Record data [11]. The dataset contains Yellow Taxi trip records since January 2009, Green Taxi trip records since August 2013 and FHV trip records since January 2015. It's noteworthy that more recent data is also available, but they lack a key feature: trip fare. Therefore, we decide to use data of previous years. Table I summarizes the features of the dataset.

| Features | Description |
|---|---|
| medallion | Taxi medallion, also known as a CPNC (Certificate of Public Necessity and Convenience). This can be treated as the unique ID for the vehicle. |
| hack_license | New York hack license to drive a yellow taxi. This can be treated as the unique ID for the driver. |
| vendor_id | A code indicating the TPEP provider that provided the record.<br>1= Creative Mobile Technologies, LLC<br>2= VeriFone Inc. |
| rate_code | The final rate code in effect at the end of the trip.<br>1= Standard rate, 2=JFK, 3=Newark,<br>4=Nassau or Westchester, 5=Negotiated fare, 6=Group ride |
| store_and_fwd_flag | This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka "store and forward," because the vehicle did not have a connection to the server.<br>Y= store and forward trip<br>N= not a store and forward trip |
| pickup_datetime | The date and time when the meter was engaged. |
| dropoff_datetime | The date and time when the meter was disengaged. |
| passenger_count | The number of passengers in the vehicle. This is a driver-entered value. |
| trip_time_in_secs | The length of trip time in seconds. |
| trip_distance | The elapsed trip distance in miles. |
| pickup_longitude | Longitude where the meter was engaged. |
| pickup_latitude | Latitude where the meter was engaged. |
| dropoff_longitude | Longitude where the meter was disengaged. |
| dropoff_latitude | Latitude where the meter was disengaged. |

TABLE I: Features of dataset

### B. Data Preprocessing

At first sight, the data seems easy to work with. However, after careful exploration, there are three preprocessing steps that we must conduct before we can smoothly use the data. Sampling and removing outliers are trivial, and transforming coordinates into taxi zones requires more effort. Besides that, because the data only contains order information, to build a more rigorous environment, we have made estimations based on this cleaned dataset. This will be introduced in detail in Section V.

*1) Sampling:* Due to computational limitation, we choose Yellow Taxi trip records in June 2013 as our training data. Data of one month contain sufficient information we need to proceed.

*2) Removal of abnormal data:* We remove abnormal data based on an exploratory data analysis of trip time, trip distance, trip fare, and pick-up and drop-off locations. The thresholds we choose to drop abnormal data are as follows. About $3\%$ of data is dropped in this process.

- Trip time: the serving time of a single trip. We drop trips with traveling time that is less than 1 minute or longer than 3 hours as most of these records have extremely short or long travelling distance.
- Trip distance: the travelling distance of a single trip. We calculate the average speed of each trip based on trip distance and trip time, and drop records that have speed over 50 mph, which is the highest speed limit in New York[1]. Trips that are longer than 30 miles are also removed because they are about twice as long as the length of Manhattan[2].
- Trip fare: the total money a driver earns from a single trip. We drop records with trip fare that is higher than $\$150$, which is the trip fare of travelling 60 miles.
- Pick-up and drop-off locations. We drop the trips whose pick-up or drop-off location is not in NYC.

[1]https://www1.nyc.gov/html/dot/downloads/pdf/current-pre-vision-zero-speed-limit-maps.pdf

[2]https://www.nycgo.com/plan-your-trip/basic-information/

*3) Transformation from coordinates to taxi zones:* The early version of this dataset does not contain zone IDs of pick-up and drop-off locations, but only pick-up and drop-off latitudes and longitudes. Continuous locations would be extremely hard to model and therefore we transformed the coordinates to taxi zones defined by NYC Taxi & Limousine Commission. The trivial way of tackling this problem is a point in polygon problem: given coordinates of a point and boundary of a polygon, decide whether this point is inside the polygon or not. Shapely library [12] provides a way to form polygons using vertices and determines if a point is inside a specific polygon or not. However, this method is time consuming, because the algorithm needs to loop through all polygons for each pick-up or drop-off location to check for a taxi zone ID. To speed up the transformation process, we introduce the bounding boxes of taxi zones. For each point, we first find the bounding boxes that contain the point and solve the point in polygon problem only for those zones. The bounding boxes can be calculated efficiently by R-tree and the time spent on this transformation is reduced significantly in this manner [4].

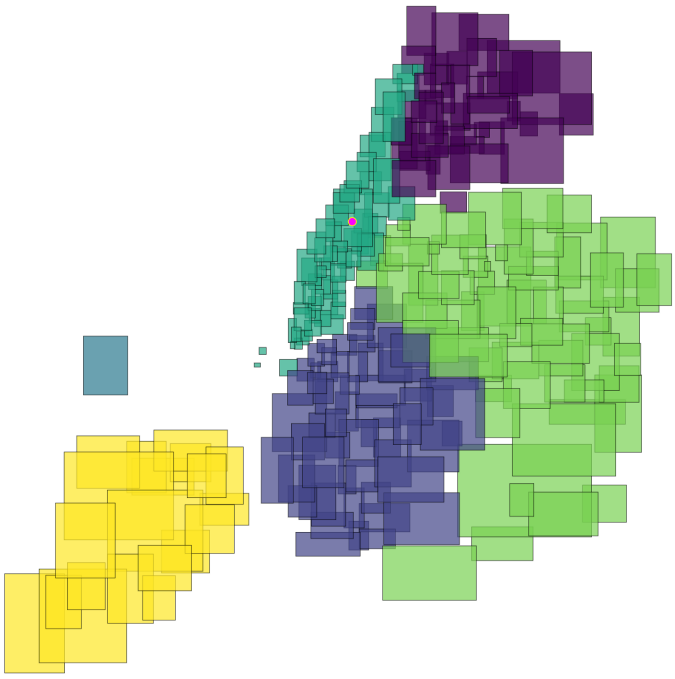


Fig. 1: Bounding Box of Taxi Zones [13]

## IV. Analysis

In this section, we provide an analysis of trip and driver features that are important for the estimation step. We show the empirical driver income in Section IV-A, discuss the discretization and the imputation of cruise time in Section IV-B, and present a potential solution to fuel cost estimation that incurs during a cruise in Section IV-C.

### A. Driver Income

We examine the distribution of drivers income. Figure 2 shows the distribution of driver income for each shift. The distribution spreads wider on weekends (right figures), which might be because drivers' have more flexible working hours on weekends. The bottom right figure shows that drivers tend to earn more from a weekend evening shift (about $400 per shift). Despite the different income spread, most drivers earn about $300 per shift in all shifts.

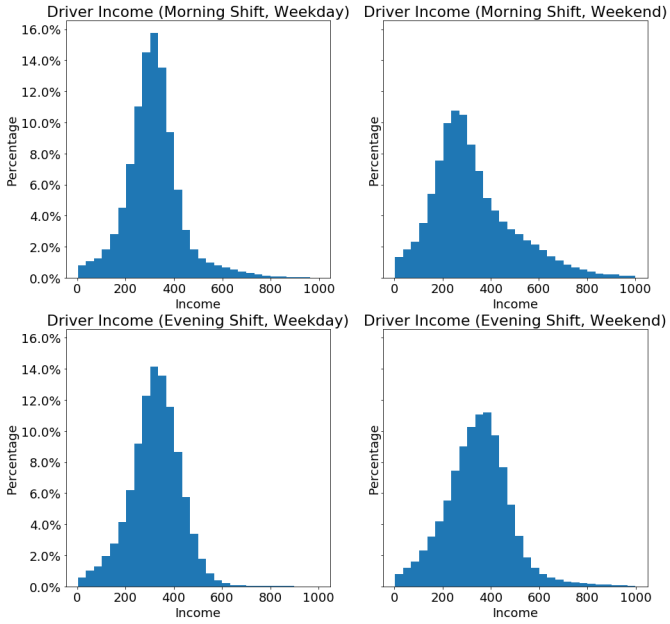



Fig. 2: Distribution of income per shift. Weekday morning shift: top left. Weekend evening shift: top right. Weekday evening shift: bottom left. Weekend evening shift: bottom right

### B. Cruise Time

Cruising is one action that greatly impacts drivers' income at the end of the day. On the one hand, if drivers cruise for new passengers in their drop-off zones, they may encounter higher taxi demand than repositioning to and looking for passengers in other taxi zones. On the other hand, the longer drivers travel to another zones without a passenger, the higher operating cost they have to bear. The historical data shows that most taxi zones have a short cruise time. That is, drivers get new passengers in a short time when they cruise in those taxi zones. The 75 percentiles of the cruise time in almost all taxi zones are below 15 minutes and the 50 percentiles are below 7.5 minutes (figure 3). The zones that do not conform include (i)

JFK airport (zone 132) and LGA airport (zone 138) with cruise time greater than 15 minutes for at least three-fourths of the time, (ii) East Flatbush/Remsen Village, Brooklyn (zone 72) with cruise time over 40 minutes at least one-fourth of the time, and (iii) Far Rockaway, Queens. Nonetheless, the last taxi zone has only 2 historical data points for the analysis so the abnormally high cruise time can be just an outlier.

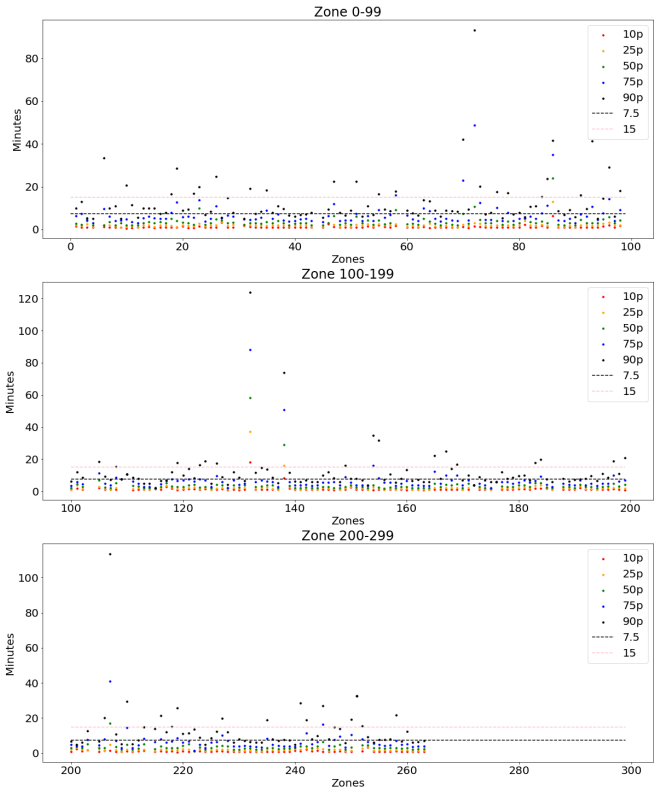



Fig. 3: Percentiles of cruise time in all 263 taxi zones

We go further by considering cruise time in each taxi zone at each time interval. However, about half of the taxi zones do not have the data for some 15-minute time intervals (Figure 4).

One explanation for missing cruise time is that these zones are known to be less in need of taxi during some specific time intervals. Experienced NYC taxi drivers are aware of this fact and prefer to not cruise in these zones. We impute the missing data with a significantly large value to reflect impossible driver-passenger matching. The imputation results in a high frequency of the one large value and, consequently, low cruise time variance.

Rather than associating these taxi zones with absolutely low taxi demand, one can argue that these zones just have relatively low taxi demand compared to nearby zones. As a result, drivers always pick nearby zones that have relatively higher demand for taxi, even just slightly. To create an environment

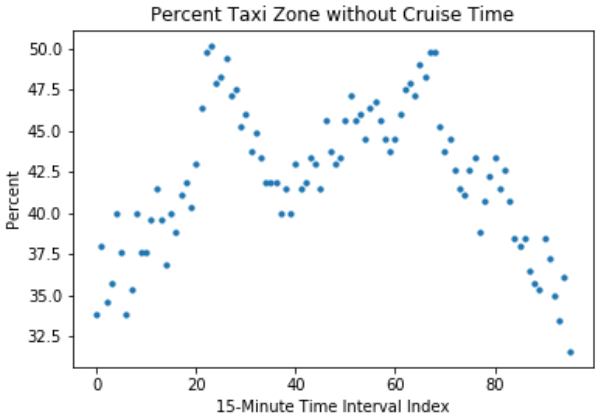



Fig. 4: Percentage of taxi zones that do not have historical cruise time using 15-minute interval.

that is not biased by NYC taxi driver's behavior, we follow two approaches to recover the missing data. First, we can think of the average cruise time in neighboring zones as a cruise time for the zone that does not have the data. Figure 5 compares heat maps of the high-value imputation (left) with the neighbor-average imputation (right). The color represents the value of log median cruise time in minutes for each taxi zone. More than 50% of the location-time buckets without the data are imputed with varying cruise times. Figure 6 shows similar information as figure 5 but the minutes are converted into 15-minute time intervals.

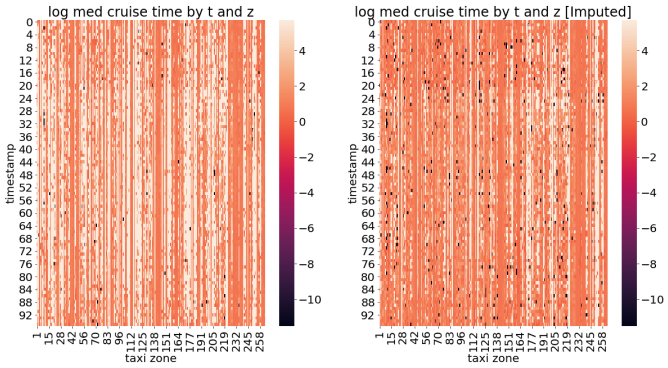



Fig. 5: Heat map comparison the of high-value imputation (left) and the neighbor-average imputation (right) in minutes.

The second attempt is to treat pickups in adjacent zones as if there are in the current taxi zone. Figure 7 and 8 compare the heat map of the high-value imputation (left) with of the neighbor-sampling imputation (right) in both minutes and 15-minute interval. However, this approach does not recover as much data as the first one does.

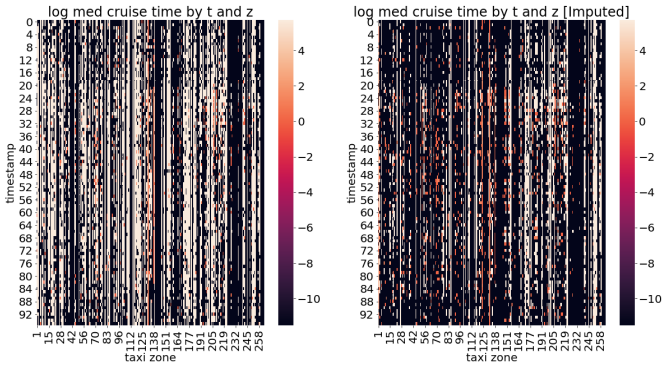



Fig. 6: Heat map comparison of the high-value imputation (left) and the neighbor-average imputation (right) in 15-minute time interval.

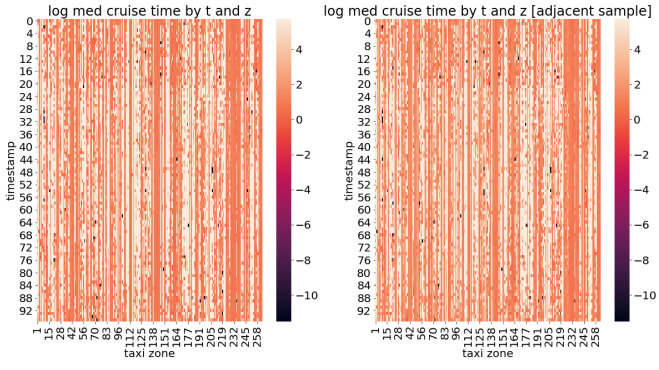



Fig. 7: Heat map comparison of the high-value imputation (left) and the neighbor-sampling imputation (right) in minutes.

### C. Trip Time and Trip Distance

We can estimate the operating cost using the estimated fuel cost and the estimated distance travelled. However, one major issue is that the distance travelled during cruising is not available. Therefore, we cannot reasonably approximate total fuel cost especially during a cruise. One solution is to proxy trip distance using trip time. Trip time allows us to infer some information that trip distance cannot during a cruise.

We explore how historical trip time and trip distance are related. Figure 9 shows a positive relationship between trip time and trip distance. For the same distance, trip time on weekdays tends to be more fluctuate than on weekends, which indicates that there is more traffic congestion on weekdays. Figure 10 shows the distance-to-time ratio at different hours in a day. High ratio indicates less congestion. The ratio at night is higher than that during the daytime, suggesting more congestion during the daytime. Moreover, we observe rather constant ratio at around 0.2 through out a day for both weekdays and weekends. These results support that historical trip time and trip distance are linearly and positively related

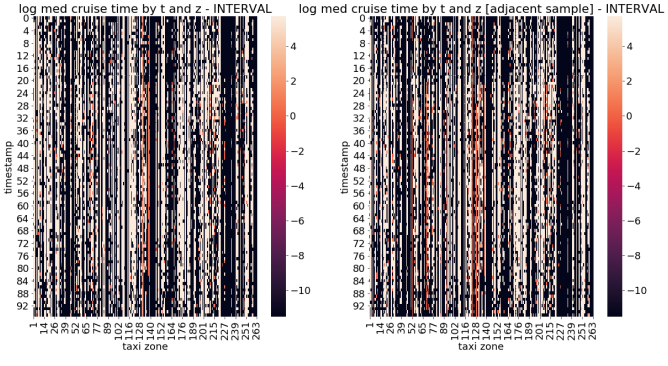



Fig. 8: Heat map comparison of the high-value imputation (left) and the neighbor-sampling imputation (right) in 15-minute time interval.

and validate the use of trip time as a proxy for trip distance.

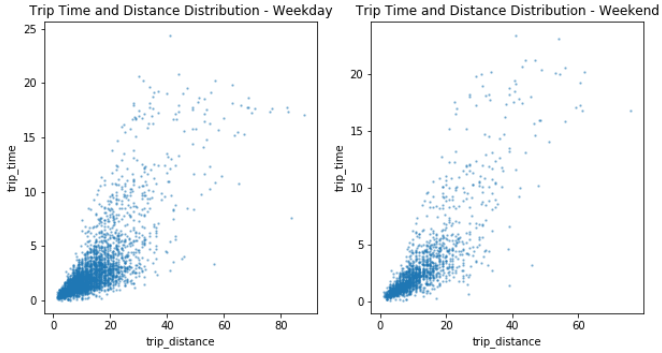



Fig. 9: Trip time and distance distribution on weekdays (left) and weekends (right)

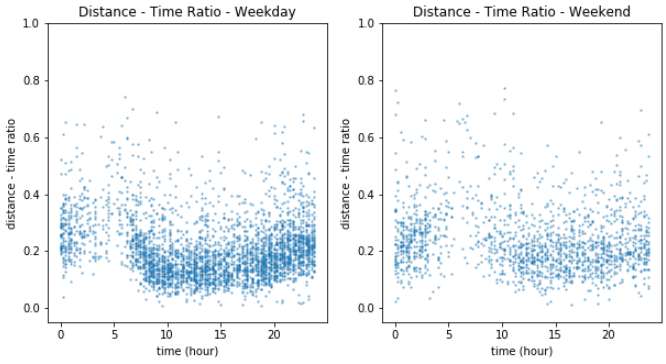



Fig. 10: Trip distance-time ratio on weekdays (left) and weekends (right)

## V. Estimation

Before setting up the reinforcement learning environment, we need to estimate the states and reward system. In this section, we estimate the online time of day and night shifts in section V-A to get the start and end condition of the training.

Then we estimate the shortest path between taxi zones to recover driver trajectory in section V-B, and fuel price in section V-C to estimate the cost. After that, we estimate the trip time and trip fare for different state in section V-D. The trip fare and cost constitute the return of each action. At last, we estimate the matching probability in section V-E.

### A. Online Time and Working Shifts

To create a training set, we need to identify working shifts from the historical dataset, as well as whether a shift is a morning or a night shift.

In order to get online time, we need to know how long a driver's shift is. The dataset does not provide us with any information about when a shift starts and ends. Therefore, we make an assumption that a driver is offline if there is at least 5 hours in between his or her two consecutive trips. The last trip before the five-hour interval is considered the last trip a driver's previous shift and the first trip before the time interval is considered the first trip of the next shift. Once the first and the last trip of a shift is identified, we can compute how long that shift takes. Algorithm 1 summarizes that steps we take:

**Algorithm 1** Calculate Online Time

```
k = 5
cruising_time_i = t_{(i+1)p} − t_{id}
for each trip i do
    if cruising_time_i > k then
        first_trip_{i+1} = True
        last_trip_i = True
        online_tme = first_trip_{i+1} - last_trip_i
    else
        first_trip_{i+1} = False
        last_trip_i = False
    end if
end for
```

where $t_{(i+1)p}$ and $t_{id}$ denote the pick-up time of the $(i+1)$-th trip and the drop-off time of the $i$-th trip. We assume if the interval between two trips is longer than 5 hours ($k = 5$), then they are the last and first trip of two consecutive shifts.

The distributions of first pick-ups and last drop-offs of a day in figure 11 reveal two shifts on both weekdays and weekends. The peaks of pick-up distributions closely follow the peaks of drop-off distributions, which reflect the fact that two shifts are back to back. Therefore, it is reasonable to assume that yellow taxi drivers who constitute a peak of pick-up distributions are the same people who constitute a peak of drop-off distributions that are further from the pick-up peak. As a result, we estimate online time of a driver according to a shift a driver is taking. The two shifts, referred to as shift A (morning shift) and shift B (night shift), are manually broken at the hours that are roughly the midpoints between their peaks. Tables II summarize time ranges of each shift on both weekdays and weekends. With a day broken into shifts, we estimate the proportion of drivers working in each shift by dividing the number of first pick-ups in a shift over the number of first pickups in both shifts. The proportion of shift A and shift B drivers in a day are similar (shown in table III) and there is no difference across weekdays and weekends within a shift.

To get some statistics for both shifts, we sample 1,000 shifts and assign labels—either A or B—to them according to their start time. We only sample shifts on weekends and repeat the process for weekdays. Table VI shows the means and the standard deviations of the sampled start and end times of shift A and B on weekdays and weekends. Then, we compute online time, the difference between a pair of start and end time. Online time from the sampling is plotted in figure 22. Most drivers have around 8 hours of active time on weekdays and 10 hours of active time on weekends.

Once we are able to identify shifts and their types, we create two training dataset, one for a day shift and one for a night shift.

### B. Recovery of Driver Trajectory

Our data only contains order information of the starting and ending position of each trip. We cannot tract the detailed trajectory of taxi's cruising directly from the data. Therefore, the trajectory need to be recovered. The trajectory is recovered by the shortest path between the starting and ending zone weighted on the average travel distance between each pair of adjacent zones. If there is no history order information between two adjacent zones, the weight is the average physical distance between these two zones.

### C. Fuel Unit Price

It's intuitive that cruising and servicing both incur fuel cost as part of the reward. Fuel cost is calculated as fuel cost $=$ distance $\times$ fuel unit price, where the fuel unit price is a hyper-parameter that needs to be estimated. The average gas price in 2013 is $\$3.602$ per gallon and the taxi fleet average fuel economy is 29 MPG (miles per gallon) [14]. Thus, the estimated fuel unit price in 2013 is $\$3.602/29MPG = \$0.124/mile$.

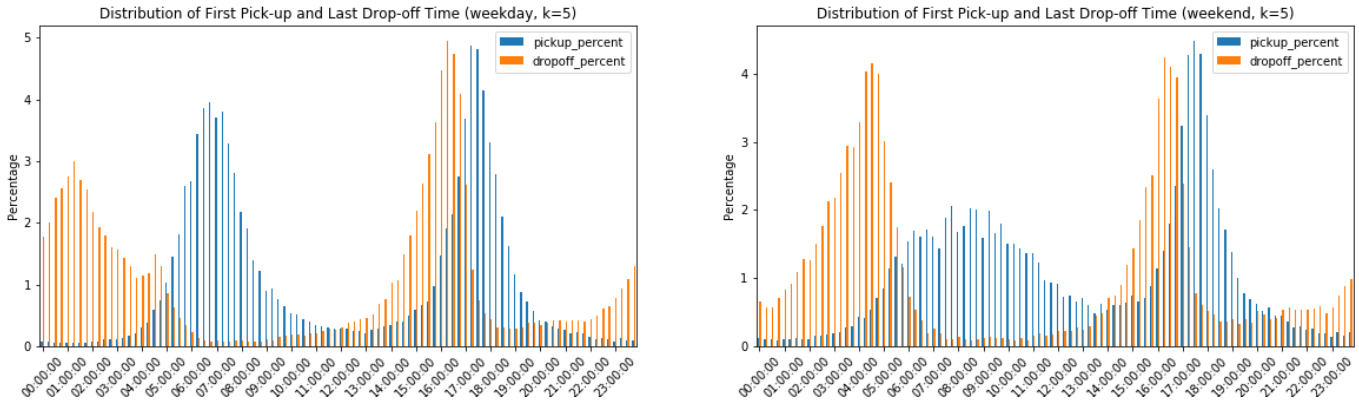



Fig. 11: Distribution of First Pick-off and Last Drop-off Time.

### *D. Travel Time and Trip Fare between Taxi Zones*

To estimate the travel time between each pair of pick-up and drop-off taxi zones, we obtain the travel time for all trips from the pick-up zone to the drop-off zone and attempt to fit the travel time to a distribution that best describes the data. Then we get a mean and standard deviation of the distribution. We considered all distributions in the SciPy library initially and narrowed down the range to speed up the procedure.

**Algorithm 2** Find the Best Fit Distribution of Travel Time

$distributions = [normal, gamma, chi2, etc.]$
$best\_distribution = normal$
$best\_params = (0.0, 1.0)$
$best\_sse = infinity$
**for** each distribution $d$ **do**
  $params = d.fit(data)$
  $pdf = d.pdf(x, mean, sdv)$
  $sse = sum((y - pdf)^2)$
  **if** $sse < best\_sse$ **then**
    $best\_distribution = d$
    $best\_params = params$
    $best\_sse = sse$
  **end if**
**end for**

Take the travel time from zone 237 to zone 236, which is the busiest pair of zones that contains the most trips (n = 57,948), as an example, we fit the time for all possible distributions as in figure 12.

Among all distributions, Johnson's SU-distribution with $\alpha = -3.22$ and $\beta = 2.06$ has the lowest error sum of squares as shown in figure 13.

The travel time from zone 262 to zone 107, which is a less busy pair of zones(n = 2580) that are farther apart, followed a similar pattern with a larger mean and variance as in figure 14.

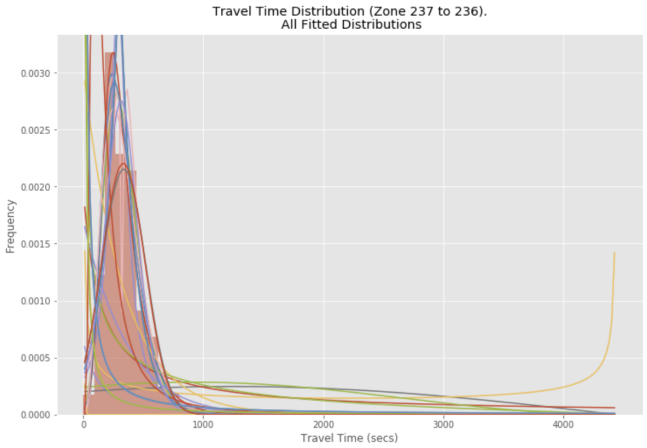



Fig. 12: All Fitted Distributions for Zone 237 to Zone 236

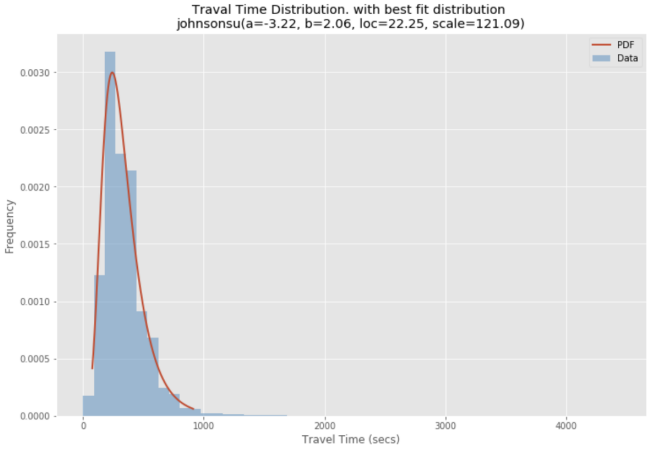



Fig. 13: Johnson's SU-distribution best fitted the data for Zone 237 to 236

The distance and fare between each pair of pick-up and drop-off taxi zones is estimated using similar procedures.

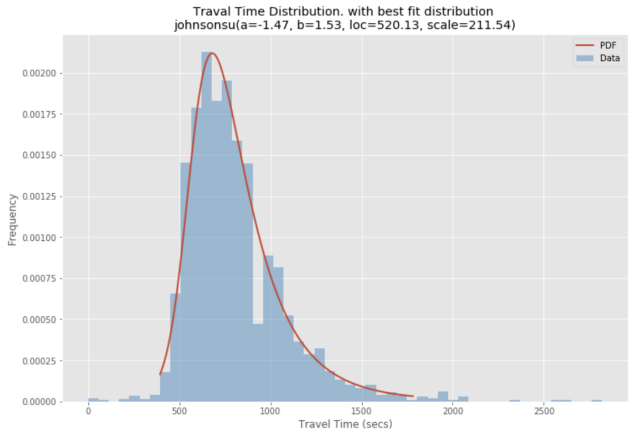



Fig. 14: Johnson's SU-distribution best fitted the data for Zone 262 to 107

### *E. Matching Probability*

The matching probability at a given time in a zone is calculated by dividing the number of pick-ups by the number of drop-offs at the corresponding time and zone. If there is no drop-off, the probability is set to 1.

$$P(matching) = \begin{cases} min(\frac{\text{\# pick-up}}{\text{\# drop-off}}, 1), & \text{if \# drop-off} \neq 0 \\ 1, & \text{if \# drop-off = 0} \end{cases}$$

Where the number of pick-ups in a zone represents the demand and the number of drop-offs represents the number of idle drivers.

Figure 15 shows the matching probability for each zone at time = 44. The darker the blue, the closer the probability is to 0. The darker the red, the greater the probability. As shown, the matching probability in Manhattan is generally high.

Evidently, this method is crude. using number of drop-offs at a specific time $t$ neglects many factors: drivers at previous time $t-1$ may still cruise at $t$, or drivers may decide to cruise in other taxi zones. However, as these behaviors are unobservable, this metric is the best proxy we can have.

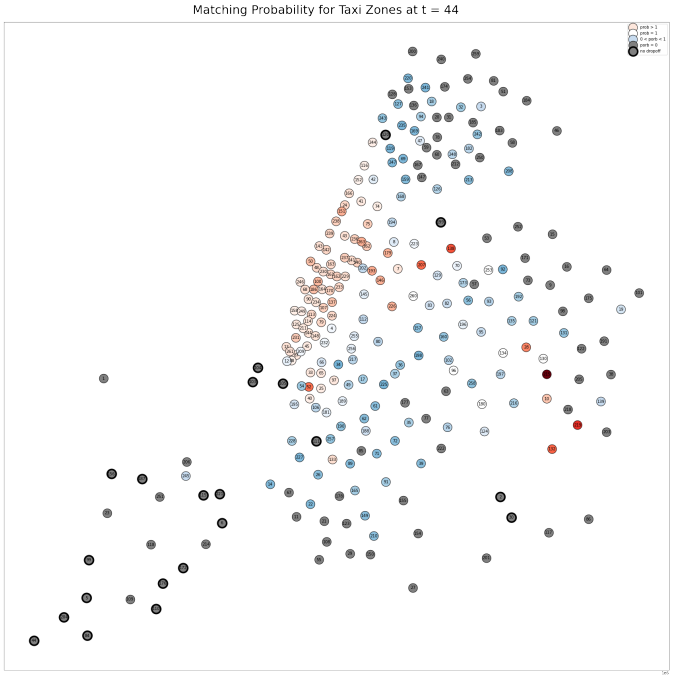



Fig. 15: The matching probability at time 44

## VI. Methodology

With all cleaned data in Section III and estimations in Section V, we can build a reinforcement learning environment. In this section, we give an overview of our approach to the re-positioning problem. First, we give technical definitions to key terms and metrics crucial to this problem in Section VI-A. Then, we introduce key assumptions we have made to create a simplified simulated environment for our reinforcement learning model in Section VI-B. Finally, we set up the environment in Section VI-C by defining five key components in reinforcement learning: agent, environment, state, action and reward. Finally, we introduce the algorithm SARSA we adopted to train optimal policy.

### *A. Term Definition and Key Metrics*

We take a driver-centric perspective on this problem so we first define drivers' possible status in the system:

**Definition VI.1** (Online/Offline)**.** When a driver starts a shift, he/she is online. Otherwise, he/she is offline.

This is a trivial definition and especially intuitive if we consider drivers within E-hailing platforms. Here, we want to be consistent with their terminology and therefore our model can be easily transferred to a wider application.

**Definition VI.2** (Cruising)**.** When a driver is online and not carrying passengers, he/she is cruising.

**Definition VI.3** (Servicing/In Service)**.** When a driver is online and not cruising, he/she is servicing/in service.

Drivers cruise because they are waiting to receive their next trips. In E-hailing platforms, they may also cruise if they are driving to pick up their next passengers. Cruising time is when they are working but not earning, and thus we want to minimize the cruising time.

Specifying drivers' status naturally leads to key metrics on which we focus to measure the efficiency and effectiveness of our policy:

**Definition VI.4** (Hourly driver earnings)**.** The hourly pay of drivers after expenses.

**Definition VI.5** (Utilization rate)**.** The share of time within a shift during which drivers are carrying passengers.

Given a fixed charge rate, it's trivial that these two metrics are highly correlated. However, to compute these two metrics, we need to know when a driver starts and ends his/her shift. As we have estimated these values, we have sufficient data to move forward.

### *B. Key Assumptions*

As the dynamic of real street network is continuous, complex and changeable, we have made the following assumptions to make our model feasible:

1) Coordinates are discretized into taxi zones. The probability of receiving an order, the value distribution of that order, distance towards another taxi zones (and related fuel cost), and time spent to travel to another taxi zones are invariant within a taxi zone. In other word, taxi zone is the smallest unit recording the driver's location.
2) Time is discretized into time intervals $\Delta t$. We denote the time space as $\mathcal{T} = [1, 2, \ldots, T]$. A driver can only pick up or drop off a passenger and make decisions on next travel at the end of each timestamp. $T$ is a finite value given $\Delta t$ as there's a mandatory time limit for each shift in NYC.
3) There is only one driver following the optimized policy the model derives, i.e. one agent. Otherwise, a taxi zone will manifest excess supply and localized competition among taxis. It will change the distribution of waiting time for each driver.
4) Drivers can cruise to other taxi zones without taking any orders in current taxi zone (drivers can refuse rides).

### *C. Environment Setup*

The environment setup is highly motivated by [15], where they used E-hailing platform dataset in Beijing from Didi Chuxing. We adjust and fit the setting into yellow cab operational model in NYC.

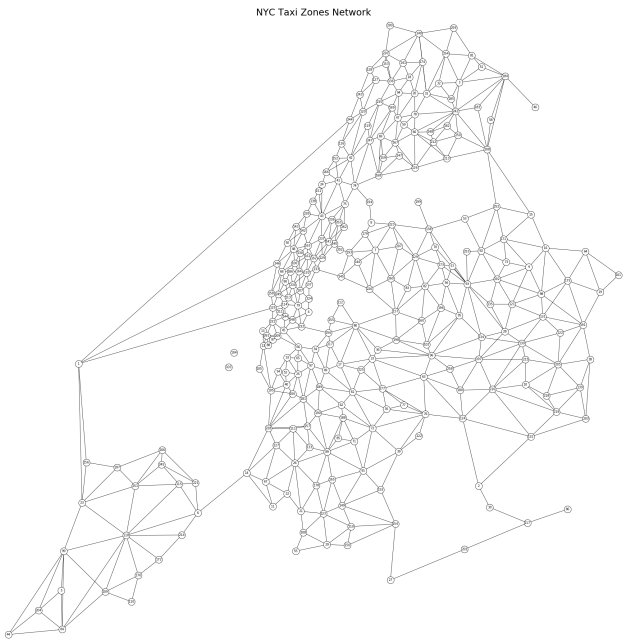



Fig. 16: NYC street network graph

*1) Agent:* One single driver will independently take actions to start its shift, take orders and make decisions on next move.

*2) Environment:* After discretization of Manhattan, street network can be represented by a weighted graph, where each taxi zone is a node and the distance between two taxi zones is the weight. After each action an agent takes, the environment will return a reward and a new state to the agent.

*3) State space:* We treat state $s = (l, t)$ where taxi zone index $l \in \mathcal{L}$ and timestamp $t \in \mathcal{T}$.

*4) Action space:* We treat action $a \in \{0, 1, \ldots, n_{\text{taxi\_zones}}\}$, where $a = x, x \neq 0$ indicates going to taxi zone $x$, and $a = 0$ indicates waiting in the current zone. It's noteworthy that:

1) Agent can cruise within the current taxi zone as well as wait within current taxi zone. We differentiate these two behaviors as 1. waiting is a common behavior for drivers especially when they are in hot spots in that area, such as downtown or transportation terminals; 2. drivers can relax and refresh during waiting, such as lunch break. Selecting between these two actions would change Agent's probability of receiving an order and reward (fuel costs).
2) Even though Agent can cruise to any taxi zones in NYC, the environment will give a infinite penalty if Agent tries to cruise to taxi zones not adjacent to current taxi zone. This makes sure that the driver is cruising realistically.

*5) Reward function:* Reward function is composed of two parts: a positive reward, trip fare $f(l_p, l_d)$, and a negative reward, fuel cost, proportional to distances traveled $-\alpha * d$. As mentioned in Section VI-C4, cruising will incur fuel cost, while waiting won't. Besides that, servicing will incur fuel cost. However, it will gather a positive reward, trip fare, as well.

### D. State Transition

When Agent starts its shift or completes a ride, the current state is denoted as $s_0 = (l_0, t_0)$. In this section, we discuss how Environment returns the reward and state for each type of action the agent takes.

If Agent chooses to wait ($a = 0$) in the current taxi zone, Environment returns $R = 0, l = l_0, t = t_0 + \Delta t$.

If Agent chooses to cruise neither to taxi zone adjacent to current taxi zone nor in current taxi zone ($a = x, x \notin$ adj$(l_0) \cup l_0$), Environment returns $R = -\infty, l = l_0, t = t_0 + \infty$.

If Agent chooses to cruise to adjacent taxi zone ($a = x, x \in$ adj$(l_0)$), Environment returns $R = -\alpha \times d_{drive}(l_0, x), l = x, t = t_0 + t_{\text{drive}}(l_0, x)$, where $\alpha$ is fuel consumption and other operating cost per unit distance during driving, $d_{\text{drive}}(l_0, x)$ is the distance it takes to drive from $l_0$ to $x$, and $t_{\text{drive}}(l_0, x)$ is the time it takes to drive from $l_0$ to $x$.

Otherwise, Agent chooses to cruise in current taxi zone ($a = l_0$), Environment returns with probability $p(s_0)$ that it will receive an order:

1) Reward $R = f(l_0, x) - \alpha \times d_{\text{drive}}(l_0, x)$, where $x$ is the destination for a generated trip, and $f(l_0, x)$ is the trip fare earned in this trip.
2) Taxi zone $l = x$.
3) Time $t = t_0 + t_{\text{cruise}}(l_0) + t_{\text{drive}}(l_0, x)$, where $t_{\text{cruise}}(l_0)$ is the time it takes to find the next order in current taxi zone.

With probability $1 - p(s_0)$, it won't receive an order and therefore:

1) Reward $R = 0$, where $x$ is the destination for a generated trip, and $f(l_0, x)$ is the trip fare earned in this trip.
2) Taxi zone $l = l_0$.
3) Time $t = t_0 + t_{\text{cruise}}(l_0)$

### E. Learning Off-Policy Control with SARSA

SARSA is an off-policy algorithm to learn Q-value. With estimated $Q$, we can decide on which action to take in each state by following a greedy policy, and therefore formulate an optimal re-positioning policy. The update rule for SARSA is:

$$Q(s_t, a_t) += \alpha (r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$$

where $\alpha$ is learning rate and $\gamma$ is discount factor.

Here, we train our SARSA model with historical data. Pseudocode for our algorithm is as in Algorithm 3.

**Algorithm 3** SARSA

```
Initialize Q(s, a) arbitrarily.
for Each observed episode do
    for Each step in the episode do
        Observe S, A, R, S', A'
        δ = R + γ * Q(S', A') − Q(S, A)
        Q(S, A) = Q(S, A) + α * δ
    end for
end for
```

## VII. Experiments

As mentioned in section V-A, we train two models using two different datasets: a day-shift dataset and a night shift dataset. Both datasets contain the same five main features: current states, current actions, next states, next actions, and rewards. The only difference is that the data that is used to create the datasets are from different shifts.

Each row of the datasets represents one step that a driver can make a decision about where to reposition to. Some actions may lead to another reposition decision point without receiving any passengers while some lead to picking up, delivering, and making another reposition decision in just one step.

### A. Evaluation

The nature of reinforcement learning machine learning is different from that of traditional supervised machine learning in a sense that we do not have one true target output as the ground truth during the training but still try to estimate possibly stochastic output. As a result, we cannot compare what we estimate in reinforcement machine learning problem directly as we traditionally do in supervised machine learning.

We evaluate the derived policy in two aspects. Firstly, we observe the convergence of mean optimal Q values (mean$_s \max_a Q(s, a)$) and TD errors ($\delta$) of the model over training iterations. The convergence of optimal Q values indicates that the derived policy is stable. TD errors encode how much a Q value of a training input from iteration i deviates from an estimated Q value for that input at iteration i. We say a model converges when all optimal Q values converge to some value and TD errors converge to 0.

Secondly, we observe how much on average a taxi driver earns if he or she follows the derived policy. This is done by creating a simulated environment that mimics the reality

using the estimations we described in section V. Then we run simulations by following the derived policy when an agent has to make any reposition decision. We are interested in how much money an agent gets at the end of each simulation.

Instead of comparing the number with the empirical earning per shift we did in Section IV-A, we transform the empirical trip data into the same format as the dataset that we use for the training. The transformation aims to mitigate impact of the time unit difference between the empirical (continuous time) and the training data (discrete time index). Then, we replicate the simulations on the transformed empirical data to inspect earning per shift. We call this empirical per shift earning and make a comparison with the one from our derived policy.

We simulated both of the policies for 100 steps, that is, drivers can make up to 100 decisions in each simulation, and stopped whenever the shift time is up or the environment returns a negative infinite reward, which means the driver is trying to reposition to a non-adjacent zone. We implemented the simulation 500 times and averaged the rewards to mitigate the influence of random initial state.

### B. Result

*1) Model Convergence:* To measure the convergence of SARSA algorithm, we track the mean $\max_a Q(s, a)$. As the algorithm converges, this value should also converge. Figure 17 shows mean $\max_a Q(s, a)$ for every 100 iterations and figure 18 TD errors for every iteration of two models that are trained on the day-shift and the night-shift datasets over 10 million iterations. The values of mean $\max_a Q(s, a)$ of both shifts converge at about the same rate and to about the same value. The TD errors oscillate around 0. However, this tells us nothing other than that we have achieved stable models for a job. The next section discusses further their performance in the context of per-shift earning.

*2) Strategy Performance:* To measure how much drivers can actually benefit from our policy, we calculate the average of simulated income of 500 epochs. In the simulated environment, drivers earn $215 in one shift by following the empirical policy, and earn $236 by following the derived policy, which means that drivers can earn 9% more per shift if following the new policy. Note that income of both policy is lower than the observed average driver income. It's partially because we include fuel cost and majorly due to the simulated environment is not accurate enough.

*3) Largest Q value for Taxi Zones:* Given a time, for each starting taxi zone with Q, there is a maximum value, which means the smallest penalty. Figure 19 shows this maximum value for taxi zones. Most zones have very large maximum Q value. We may infer that the preference to the optimal policy is very obvious and significant in all zones.

*4) Optimal Action:* Based on these maximum Q values, there is an optimal action for each taxi zone, that is, either waiting in the same zone or travelling to some zone. Figure 21 shows this optimal action from 10:00 to 10:15 in the map as an example. We can find that there are some "hot spots" which are the ultimate destination of many other nodes. Quantitative results of this popularity can be calculated using PageRank [16]. For example, Zone 94 is the most popular destination during 10:00 to 10:15, following by zone18, 221 and 6. Their page rank value is significantly higher than that of other zones, which can be observed from 20. Taking a close look at special zones like zone 1 for Newark, zone 132 for JFK and zone 138 for LaGuardia, we can find that they are very often popular destinations in day time. Looking through the optimized policy at different time intervals, we may find that the percentage of zones that travel within itself is much higher during working hours, especially rush hours in Manhattan and Brooklyn areas. For other hours will less zones traveling within itself, most of these zones are in Manhattan area. On the contrary, in Queens, upper Bronx and Staten Island, optimal policies are mostly cross zones. We may thus infer that shorter trips are preferred when and where traffic is heavy.

The flow of optimal action and how it changes within a day can be observed more clearly by using the dynamic graph tool we implement by choosing different time. We can also zoom in to Manhattan area as in Figure 21 to see the flow more clearly when the nodes are too close to each other in the whole map.

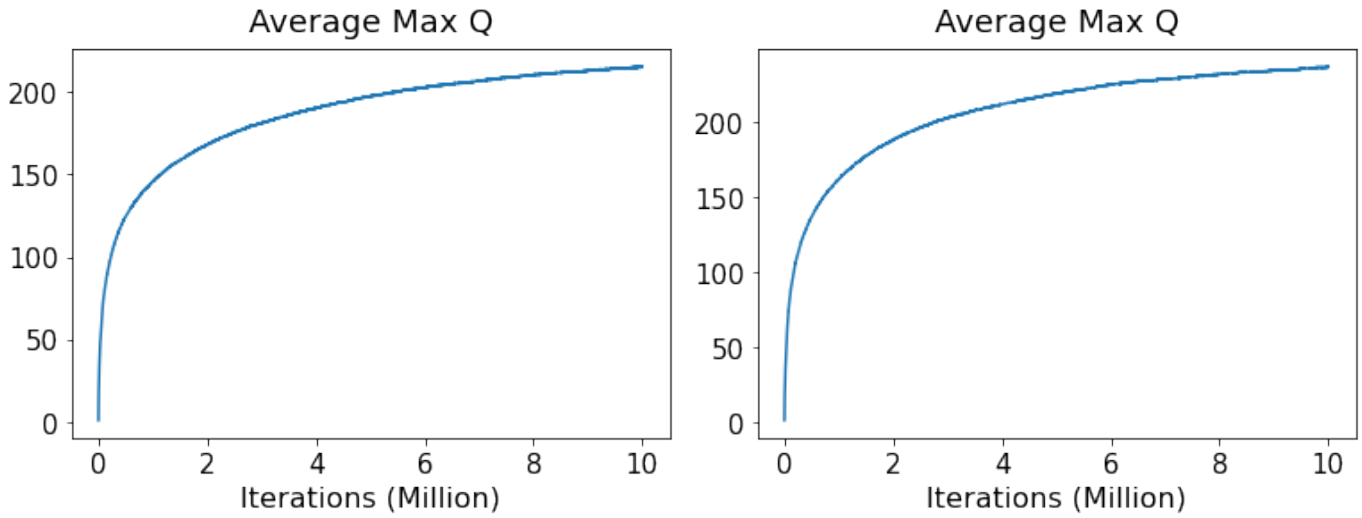
## VIII. Conclusion

### A. Summary

In this project, we have explored NYC Taxi Trip Record dataset. With it, we have conducted numerous estimations and built a reinforcement learning environment to simulate demand for NYC taxi. Then, we have applied SARSA, a widely-used algorithm to historical data and trained best policy. The loss converges and we have shown a 9% increase of income per shift if the driver follows our policy in comparison with the empirical per shift earning.

### B. Future Work

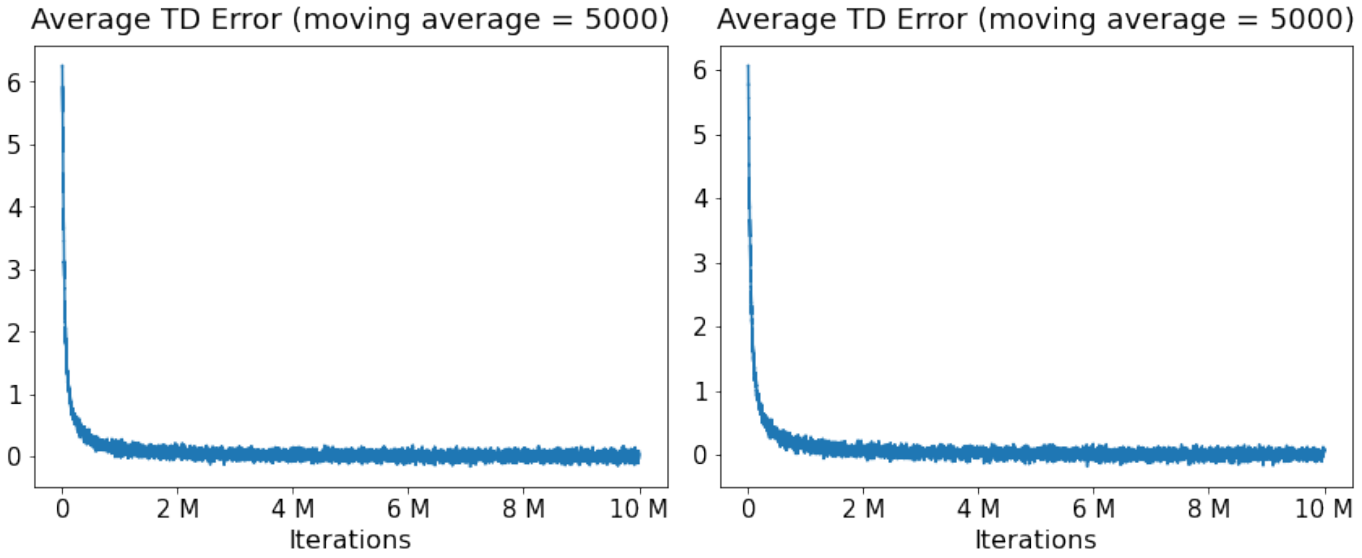
Due to limited computational resources and time, we have made several simplifications in our projects, and we believe lifting these restrictions will be a good start to improve our work.




(a) Day Shift (A)

(b) Night Shift (B)

Fig. 17: Average optimal Q over training iterations




(a) Day Shift (A)

(b) Night Shift (B)

Fig. 18: TD errors over training iterations

Data-wise, we have sampled only one month to explore and train on. Seasonality and short-term fluctuation may cause our result to deviate from the true distribution. With more computational resources, a model that generalizes better can be trained.

Environment-wise, we have estimated many key values in the environment in a deterministic fashion and therefore lost variance that real road network possesses. With a more careful design of estimations of these values, the environment will be more alike to the real world. Eventually, we can use this environment to train on-policy algorithms and more interesting results may emerge.

Training-wise, because of the reasonable size of discretized states, we have used a tabular approach to approximate value function. More value function approximation methods can be experimented.

## IX. Miscellany

### A. Acknowledgements

We would like to express our gratitude to Vineet Goyal, Zhiwei Qin, Shuaiji Li and Qun Li for mentoring us throughout the term. They have guided us in each phase of the project and provided invaluable advice when we encountered technical

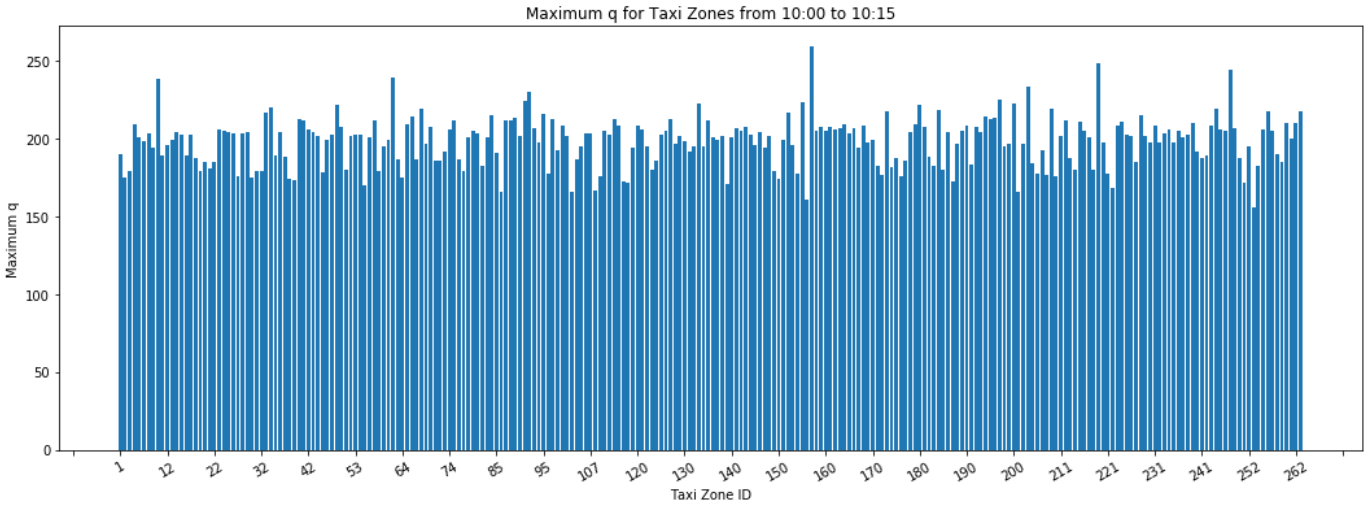



Fig. 19: Maximum q value for each zone.

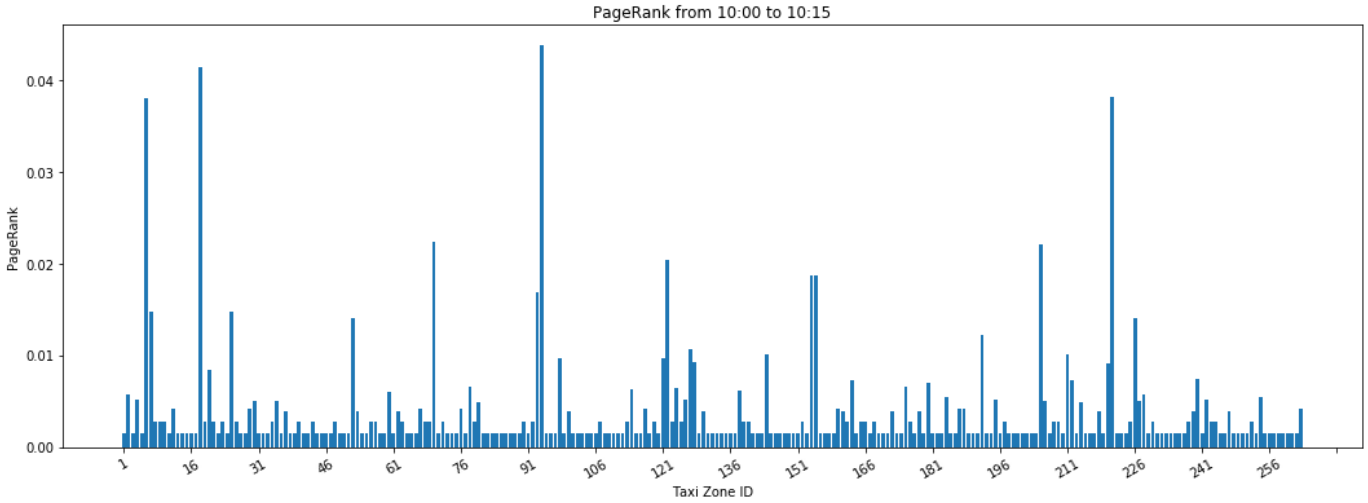



Fig. 20: PageRank value for each zone.

obstacles. We would also love to thank Sining Chen, Adam Kelleher and Eleni Drinea for organizing this capstone project and providing us with this unique opportunity to apply theory to practice.

### B. Contribution

*1) Tian Wang:* Organized weekly meetings, managed project progress, conducted literature review, designed reinforcement learning environment and implemented SARSA algorithm.

*2) Bo Jumrustanasan:* Conducted data cleaning, analyzed cruise time, estimated shift, and trained optimal policy with empirical data.

*3) Yingyu Cao:* Conducted data cleaning, estimated fuel unit price, and evaluated model performance.

*4) Xue Xia:* Estimated distribution of trip distance and trip time, produced a solution to point-in-polygon problem and shortest path problem, and provided static plots and dynamic for optimal actions.

*5) Tianyi Wang:* Estimated distribution of trip distance and trip time, generated random requests, and estimated matching probability.

### C. Ethical Consideration

This data is publicly accessible and therefore there won't be a leakage of proprietary data. Passenger information is omitted in the dataset and thus passengers' privacy is protected.

### D. Github

We have hosted our code on Github.

## REFERENCES


[1] "Improving efficiency and managing growth in new york's for-hire vehicle sector," New York City Taxi and Limousine Commission and Department of Transportation, Tech. Rep., Jun 2019.

[2] F. He and Z.-J. M. Shen, "Modeling taxi services with smartphone-based e-hailing applications," *Transportation Research Part C: Emerging Technologies*, vol. 58, pp. 93 – 106, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X15002387

[3] X. Qian and S. V. Ukkusuri, "Taxi market equilibrium with third-party hailing service," *Transportation Research Part B: Methodological*, vol. 100, pp. 43 – 63, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0191261516301461

[4] C. Mao, Y. Liu, and Z.-J. M. Shen, "Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 115, p. 102626, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X19312227

[5] K. Lin, R. Zhao, Z. Xu, and J. Zhou, "Efficient large-scale fleet management via multi-agent deep reinforcement learning," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1774–1783. [Online]. Available: https://doi.org/10.1145/3219819.3219993

[6] J. Holler, R. Vuorio, Z. Qin, X. Tang, Y. Jiao, T. Jin, S. Singh, C. Wang, and J. Ye, "Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem," 2019.

[7] Z. Wang, Z. Qin, X. Tang, J. Ye, and H. Zhu, "Deep reinforcement learning with knowledge transfer for online rides order dispatching," in *2018 IEEE International Conference on Data Mining (ICDM)*, 2018, pp. 617–626.

[8] 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 *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, ser. CIKM '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 2329–2334. [Online]. Available: https://doi.org/10.1145/2983323.2983689

[9] T. Verma, P. Varakantham, S. Kraus, and H. C. Lau, "Augmenting decisions of taxi drivers through reinforcement learning for improving revenues," in *Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling, ICAPS 2017, Pittsburgh, Pennsylvania, USA, June 18-23, 2017*, L. Barbulescu, J. Frank, Mausam, and S. F. Smith, Eds. AAAI Press, 2017, pp. 409–418. [Online]. Available: https://aaai.org/ocs/index.php/ICAPS/ICAPS17/paper/view/15746

[10] Y. Gao, D. Jiang, and Y. Xu, "Optimize taxi driving strategies based on reinforcement learning," *International Journal of Geographical Information Science*, vol. 32, no. 8, pp. 1677–1696, 2018. [Online]. Available: https://doi.org/10.1080/13658816.2018.1458984

[11] B. Donovan and D. Work, "New york city taxi trip data (2010-2013)," 2016. [Online]. Available: https://doi.org/10.13012/J8PN93H8

[12] S. Gillies *et al.*, "Shapely: manipulation and analysis of geometric objects," toblerity.org, 2007–. [Online]. Available: "https://github.com/Toblerity/Shapely"

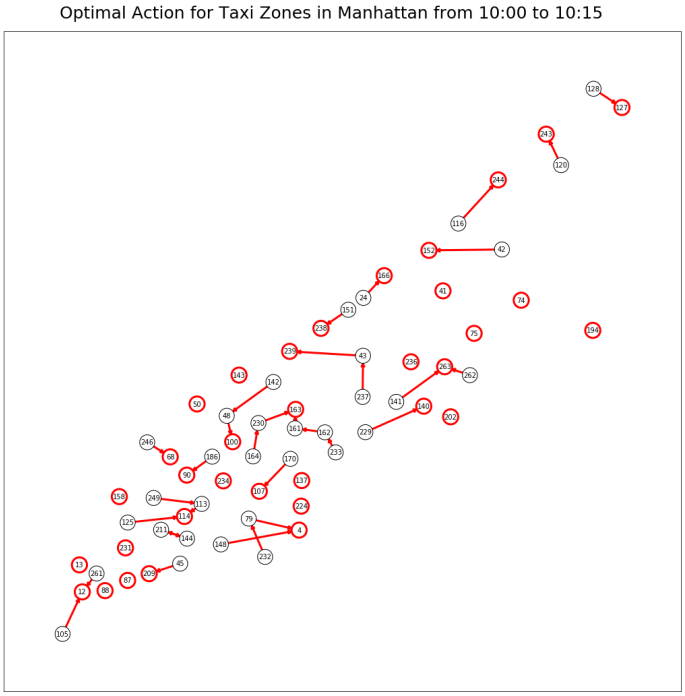
[13] R. Shekhar, "Geospatial operations at scale with dask and geopandas," Nov 2018. [Online]. Available: https://towardsdatascience.com/geospatial-operations-at-scale-with-dask-and-geopandas-4d92d00eb7e8

[14] "2014 taxicab fact book," New York City Taxi and Limousine Commission, Tech. Rep., 2014.

[15] Z. Shou, X. Di, J. Ye, H. Zhu, H. Zhang, and R. Hampshire, "Optimal passenger-seeking policies on e-hailing platforms using markov decision process and imitation learning," *Transportation Research Part C: Emerging Technologies*, vol. 111, p. 91–113, Feb 2020. [Online]. Available: http://dx.doi.org/10.1016/j.trc.2019.12.005

[16] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.


(a) Optimal action for each zone




(b) Optimal action for each zone in Manhattan area

Fig. 21: Blue circle means waiting in the same zone; red means traveling, with circle within the same zone and arrow to the optimal travel destination.

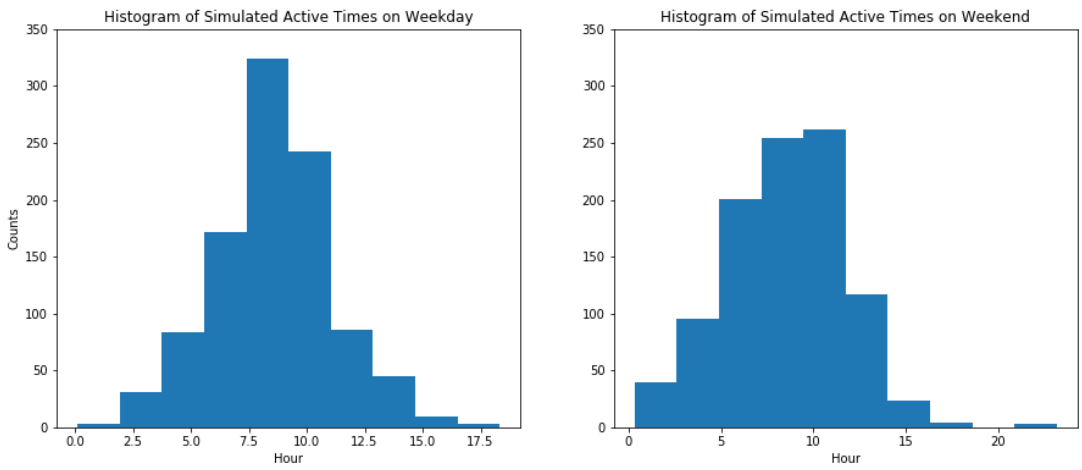



Fig. 22: Histograms of the simulated online times of drivers on weekdays and weekends.

| Weekday | First Pick-up | Last Drop-off |
|---|---|---|
| Shift A | (23:30, 11:30] | (8:30,20:30] |
| Shift B | (11:30, 23:30] | (20:30, 8:30] |

(a) Shifts on weekends

| Weekend | First Pick-up | Last Drop-off |
|---|---|---|
| Shift A | (2:00, 14:00] | (10:00,22:00] |
| Shift B | (14:00, 2:00] | (22:00, 10:00] |

(b) Shifts on weekdays

TABLE II: Hour breakdown for both shifts on weekdays and weekends

| | Weekday | Weekend |
|---|---|---|
| Shift A | 0.51 | 0.51 |
| Shift B | 0.49 | 0.49 |

TABLE III: The proportions of drivers in each shift on weekdays and weekend

| | Shift | Mean Online Time | STD Online Time |
|---|---|---|---|
| Weekday | A | 8.76 | 2.56 |
| | B | 8.36 | 2.77 |
| Weekend | A | 7.77 | 3.32 |
| | B | 9.74 | 2.71 |

TABLE IV: Means and standard deviations of simulated online times

| | Shift | Mean Start Time | Mean End Time | STD Start Time | STD End Time |
|---|---|---|---|---|---|
| Weekday | A | 6:58 | 15:22 | 1:42 | 1:53 |
| | B | 17:17 | 1:43 | 1:48 | 2:55 |
| Weekend | A | 8:32 | 16:05 | 2:35 | 2:08 |
| | B | 17:51 | 3:22 | 1:53 | 2:45 |

TABLE V: Means and standard deviations of observed start and end times.

| | Shift | Mean Start Time | Mean End Time | STD Start Time | STD End Time |
|---|---|---|---|---|---|
| Weekday | A | 6:55 | 15:40 | 1:41 | 1:53 |
| | B | 17:17 | 5:52 | 1:54 | 7:51 |
| Weekend | A | 8:41 | 16:27 | 2:33 | 2:04 |
| | B | 17:27 | 5:19 | 2:31 | 5:24 |

TABLE VI: Means and standard deviations of the simulated start and end times