It Can be Cheaper: Using Price Prediction to Obtain Better Prices from Dynamic Pricing in Ride-on-demand Services

Suiming Guo
The Chinese University of Hong Kong
Hong Kong, China

Chao Chen
Chongqing University
Chongqing, China

Yaxiao Liu
Tsinghua University
Beijing, China

Ke Xu
Tsinghua University
Beijing, China

Dah Ming Chiu
The Chinese University of Hong Kong
Hong Kong, China

ABSTRACT

In emerging ride-on-demand (RoD) services such as Uber or Didi (in China), dynamic pricing plays an important role in regulating supply and demand, trying to make such service, to some extent, more convenient for passengers. Despite the convenience, dynamic pricing also exerts mental burden on passengers: they wonder whether the current price is low enough to accept, or if it is not, what they could do to get a lower price. Without extra information, passengers sometimes feel anxious and lose satisfaction. It is thus necessary to provide more information to relieve the anxiety, and price prediction is one of the solutions.

In this paper we predict the dynamic prices to help passengers understand whether they could get a lower price in neighboring locations or within a short time. We first divide a city into rectangular cells, and use entropy and the temporal correlation of prices to characterize the predictability of prices of each cell. Based on the predictability of prices, we claim that different prediction algorithms should be used in different city areas, to balance between efficiency and accuracy. We design and implement two predictors, namely a Markov predictor and a neural network predictor, and evaluate their performance based on the real data we collected from a major RoD service provider in China. The results verify that the Markov predictor works well enough in highly-predictable areas, and the neural network predictor, while requiring more computation time, works better in areas with lower predictabilities. Finally, we also evaluate the effects of our prediction, i.e., the probability that passengers (in different city areas) could benefit from the prediction and get a lower price.

MobiQuitous 2017, November 7–10, 2017, Melbourne, VIC, Australia

ACM ISBN 978-1-4503-5368-7/17/11…$15.00
https://doi.org/10.1145/3144457.3144476

CCS CONCEPTS

• **Information systems** → **Location based services**; • **Human-centered computing** → **Empirical studies in ubiquitous and mobile computing**; • **Computing methodologies** → *Machine learning approaches*;

KEYWORDS

Price prediction, ride-on-demand service, dynamic pricing, limit of predictability

ACM Reference Format:
Suiming Guo, Chao Chen, Yaxiao Liu, Ke Xu, and Dah Ming Chiu. 2017. It Can be Cheaper: Using Price Prediction to Obtain Better Prices from Dynamic Pricing in Ride-on-demand Services. In *Proceedings of MobiQuitous 2017*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3144457.3144476

1 INTRODUCTION

Emerging ride-on-demand (RoD) services such as Uber and Didi have drawn increasing attention recently. As a supplement of the traditional taxi service, RoD service attracts passengers by its cleanness, convenience, as well as flexible and affordable prices, and on the other hand, attracts drivers who want to make use of their idle cars without applying for licenses. Meanwhile, RoD service also creates concerns that sometimes its dynamic prices go so high (as high as 5 or 10 times the normal prices) during big event or in bad weather.

Dynamic pricing is the core and distinctive feature in RoD service, and it reflects the effort in controlling the supply (the number of cars) and demand (the number of requests) in a particular location so that an equilibrium is approached: a higher price reduces demand and increases supply in a busy area, and vice versa in a not-busy area. Specifically, the effects of a higher price on the supply include not only bringing more cars onto the roads, but also motivating surplus supply to flow from low- to high-demand areas.

On the other hand, dynamic pricing exerts mental burden on passengers. In traditional taxi service with fixed pricing, passengers are able to estimate the trip fare based on their personal experience. In emerging RoD services, however, they have an extra task before making decisions – predicting the dynamic prices – based on their estimate of the supply &

demand condition nearby, and this is invariably inaccurate for most individual passengers. Without relevant information, passengers may wonder:

(1) *Could I get a lower price if I choose to wait for a short time (e.g., 10 to 20 minutes)?*
(2) *Could I get a lower price if I choose to walk away for a short distance (e.g., hundreds of meters)?*

These questions create hesitation and exert mental burden.

Giving more information to passengers help ease the anxiety, including, for example, by explaining why the current price is high or low, giving a recent history of prices to passengers, predicting the prices in the next time slot in the neighboring locations, etc. Among these ideas, the most direct one is price prediction, and passengers could answer the above questions and make decision based on the predicted prices, instead of relying on a rough and unreliable estimate of the supply & demand nearby.

Price prediction have not received much attention in RoD services. [3] tried to predict the future prices by guessing the relationship between dynamic prices and a combination of supply and demand. Because most RoD services keep their dynamic pricing algorithms as secrets, guessing the relationship from data is not accurate enough to generate a good prediction.

Instead, we propose to predict dynamic prices based on the historical data, instead of trying to learn some unknown, internal relationship from outside. This is inspired by the demand prediction work that was common in the studies on taxi service. These studies try to predict the demand in taxi service by studying the historical data, using methods such as time-series analysis, SVM or neural networks.

In RoD services, dynamic pricing is represented by a price multiplier. Specifically, the price of a trip is the product of a dynamic price multiplier (dependent on the supply & demand condition) and a fixed normal price (dependent on the estimate distance & time of the trip). The geographical and temporal variation of the price multiplier is the source of anxiety, and is also the target of prediction in our paper.

The regularity of price multipliers varies among different areas in a city. For example, at the airport terminals or train stations, the price multiplier is always stable during the day, not only because the demand is stable throughout the day, but also because drivers know this and are more inclined to cruise in these areas looking for passengers. In business areas (e.g., the Financial Street in Beijing), contrarily, the price multiplier is more volatile, as there are different kinds of passengers leaving a business area, including those in travel, business visit, going/leaving workplaces, etc. and there is not a regularity of passengers' behavior from day to day.

This inspires us to use a metric – *maximum predictability* Π^{max} – to first characterize the patterns of price multipliers in different areas of a city. The calculation of predictability is based on the entropy of price multiplier and captures both the randomness and the temporal correlation behind it. The predictability metric expresses both the regularity and randomness of price multipliers. For example, an area with a Π^{max} of 0.3 means that for 30% of the time the price multiplier is regular and predictable, and for the rest 70% of the time, the price multiplier appears to be random.

We then use different prediction algorithms to predict price multipliers in areas with different Π^{max}. An area with high Π^{max} have higher regularity in price multipliers, and thus the prediction could be done by easier and faster algorithm such as Markov-chain predictor, considering only temporal correlation between price multipliers. Alternatively, the prediction algorithm should consider more factors when used in an area with lower Π^{max} including, for example, temperature, precipitation, wind speed, local events, etc. A prediction algorithm such as neural network predictor could accomplish this.

In this paper, we tackle the price multiplier prediction problem based on the data collected in Beijing from a major RoD service provider in China. We first introduce the concept of maximum predictability Π^{max}. We then divide the map of Beijing into small and rectangular cells, and calculate the Π^{max} of each cell, in an effort to present an overall picture of Π^{max}. We also show that different functional areas (e.g., business, residential and transportation area), as a larger and more representative unit than cells, have different Π^{max}s. This shows that Π^{max} is a metric that could characterize both city cells and functional areas. We then implement two predictors, namely a Markov-chain predictor and a neural network (NN) predictor, and evaluate their performance in different city cells and functional areas. Our evaluation results show that while in areas with high predictability the Markov-chain predictor is both faster and more accurate, the NN predictor significantly outperforms the Markov-chain predictor in areas with low predictability. This justifies our claim that different prediction algorithms should be used in different areas. Finally, we also show the effect of our prediction: if passengers in different functional areas follow our prediction and choose the lowest prices nearby, how many of them could get a lower price.

The remainder of this paper is organized as follows. §2 presents some background information and our dataset. In §3 we discuss the maximum predictability of price multipliers and uses it to characterize city cells and areas. Two predictors are implemented in §4 and we also evaluate the performance and effect of our prediction. §5 discusses related work and §6 concludes the paper.

2 BACKGROUND AND DATASET

2.1 Ride-on-demand Services

Taxi service is probably the oldest RoD service: people could request a ride when they wish, by hailing on the road, calling a taxi dispatch center, or waiting at a taxi stand. The pricing scheme in taxi is mostly fixed, and is dependent on the time and distance between two locations. In recent years some new RoD service providers come into the market, and the major difference between them and taxi service is the use of dynamic pricing scheme. The price of a trip now is the product of a dynamic price multiplier and the normal price (comparative to the price in taxi). Another difference is that new services

all rely on GPS-assisted mobile apps to accurately locate both drivers and passengers.

In this subsection we explain the user interface of a mobile app of a typical RoD service, shown in Fig. 1, to give a generic explanation of how such a service works. Usually a passenger first opens the app on his/her mobile phone when s/he wants to travel from a boarding location A to an arriving location B, and types the address of both locations in the app. The passenger could also choose "when to ride (now or several minutes later)" and "using coupon".

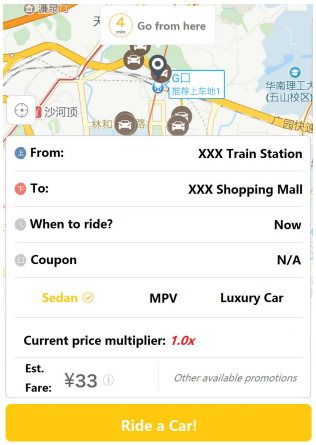

Figure 1: The user interface of a typical RoD service.

After the passenger has specified the locations and chosen all available options, the mobile app sends all the information to the service provider, and obtains in return (a) the estimated trip fare and (b) the current dynamic price multiplier. The price multiplier reflects the current supply and demand condition around the boarding location A. The service provider sets a lower and upper bound on the price multiplier. The passenger then chooses either to accept the current price (by pressing "Ride a Car!" button) or give up the current fare estimation if s/he considers the price multiplier too high.

2.2 Anxiety from Dynamic Pricing

As mentioned in the introduction, passengers' anxiety comes from their uncertainty about the dynamic prices: they don't know whether the current dynamic price multiplier is low enough, or rather, whether they could get a lower one if they choose to wait for a couple of minutes or to walk away for hundreds of meters.

The possibility of getting a lower price has been validated by [7] and [3]. The authors in [7] concludes that during rush hours, if a passenger is getting a too high price multiplier, s/he could usually walk away for 1 to 2km to get a lower multiplier. [3] also proposes a way to avoid getting a high price multiplier – request a ride starting from hundreds of meters away, where the price multiplier is lower, and then walk there before the car arrives.

Passengers' reaction to dynamic prices also indicates their anxiety. [6] measures passengers' reaction, i.e., how many times of fare estimations they perform before finally giving up or getting on a car. As mentioned in §2.1, a passenger may choose to give up or accept the fare estimation. If s/he considers the price multiplier is too high, he may choose to wait for a while or walk away and estimate the fare again. [6] shows that only in 39.77% cases one accepts the price multiplier after only one fare estimation. This justifies that most passengers feel uncertain about the dynamic price multiplier.

2.3 Our Dataset

Our data of the RoD service is collected from Shenzhou UCar[1], one of the major RoD service providers in China. By the end of 2015, Shenzhou UCar's service covers more than 50 cities in China, with a fleet of more than 30,000 cars, offering more than 300,000 trips per day [12].

We collect the event-log dataset from Shenzhou UCar's service in Beijing, China. The event-log dataset contains two major events: ***EstimateFee*** and ***CreateOrder*** event. The ***EstimateFee*** event is triggered when the mobile app sends all the information of a passenger's request (including the two locations, the requested group of cars, the time of the request, etc.) to the service provider, and returns the current price multiplier and the estimated trip fare. When one performs multiple fare estimations, the same number of ***EstimateFee*** events are generated. The ***CreateOrder*** event is triggered only when the passenger is satisfied with the current price multiplier and presses the "Ride a Car!" button.

The event-log dataset contains the complete record of events in the complete 4 weeks in Oct, 2016 (from Oct. 3 to Oct. 30) in Beijing. Each entry includes the time it happens, the event code (i.e., *EstimateFee* or *CreateOrder*), the IMEI of the passenger's device, location information, the estimated trip fare and the price multiplier for *EstimateFee*, etc. The volume of the dataset is about 5.3 million, and all entries are properly anonymized.

We also find out from the event-log dataset that the service provider sets a lower and upper bound for the price multiplier. The lower bound is $m = 1.0$ and the upper bound is $U = 1.6$.

We also collect climate data for building NN predictor. Specifically, we collect the daily precipitation and hourly temperature data reported at the Beijing Capital International Airport from TuTiempo[2]. The granularity of the precipitation data may not be small enough: we could not find the hourly precipitation data, and the precipitation data at the airport doesn't represent the precipitation in other locations of Beijing. The temperature data doesn't have this problem.

3 PREDICTABILITY OF PRICE MULTIPLIERS

In this section, we first define city cells and areas of Beijing, and present the variation of price multipliers in different city areas. We then define the *maximum predictability* Π^{max}, and show the distribution of Π^{max}. The difference of Π^{max} in

[1]Shenzhou UCar: http://www.10101111.com/
[2]TuTiempo: https://en.tutiempo.net/records/zbaa

various city cells and areas makes it necessary to use different prediction algorithms.

3.1 City Cells and Areas

To show the predictability of price multipliers in different locations of a city, we first divide the city map into $N * N$ rectangular cells of equal sizes, and cell (i, j) denotes the cell on row i, column j $(i, j = 1, 2, ..., N)$. In this paper, we set $N = 100$, so that each cell is small enough to enhance finer granularity of data analysis, and is also large enough to have enough events happening inside each cell. When $N = 100$, each cell is about $420 * 300$ square meters.

In addition to cells, we also consider a larger and also more representative unit – the functional areas of a city. A large city always has a clear partitioning of functional areas including, for example:

- *business area*: the place for working. Different industries (e.g., financial or IT) may have different areas.
- *residential area*: the place for living. In China, some large residential areas accommodate ≥10,000 residents.
- *transportation area*: typical transportation areas include airport terminals, railway stations for inter-city trains, etc.

The partitioning of functional areas could be acquired either from the city plan of Beijing[3], or by analyzing real data. The task of identifying functional areas of a large city such as Beijing is a sophisticated task, and has been studied using various techniques based on different sources of data. For example, in our previous work [7], we cluster the boarding and arriving locations of trips from frequent passengers (i.e., those passengers having more trips monthly) in RoD service using the k-means algorithm and verify that the clustering result matches the city plan as well as our knowledge of the distribution of functional areas in Beijing.

Fig. 2 shows some typical functional areas found in [7] and validated by the city plan: area 1 and 2 are typical business areas; area 3 and 4 are typical residential areas; and area 5 to 7 are the airport terminals and a major train station – typical transportation areas. We use these typical functional areas in the remainder of the paper.

3.2 Variation of Price Multipliers

The target of study is the price multiplier a passenger is faced with when performing a fare estimation. We divide the whole time range of the data (i.e., 4 weeks) into K time slots with equal time length Δt. We then calculate the average price multiplier in all *EstimateFee* events happening in a particular city cell/area during each time slot. So for each city cell/area, we obtain a sequence of length K, and the k-th element R_k of it is the average price multiplier in the k-th time slot.

In our data, we choose $\Delta t = 1$ hr to include enough number of *EstimateFee* events in each time slot. In other words, we focus on the hourly average price multiplier in each city cell/area. In this case, $K = 24 * 7 * 4 = 672$.

[3]Beijing's city plan (in Chinese): http://bit.ly/2sDE3e2

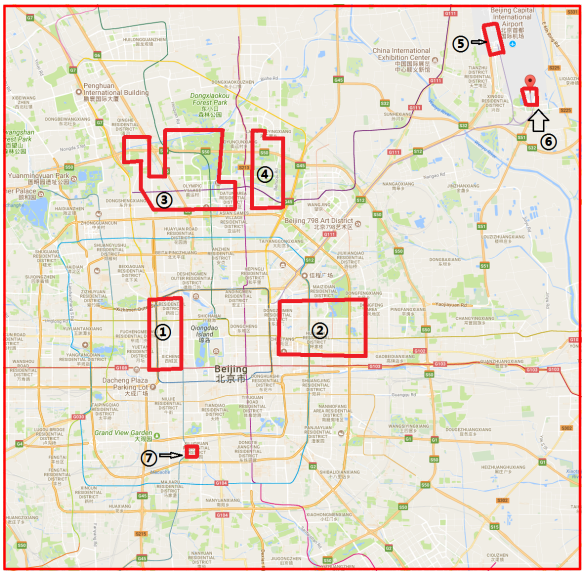

Figure 2: The map of Beijing and typical functional areas.

The variation of price multipliers may have different regularities in different locations. In Fig. 3 we give the variation of the hourly average price multiplier in typical business, residential and transportation area during the 4-week time range of our data (from 0:00 on Day 0 to 23:59 on Day 27).

The variation in transportation area (Fig. 3(c)) exhibits a strong temporal pattern. The price multiplier is higher than 1.2 from 11pm to 5am, due to the higher demand and lower supply at midnight. During the evening (from about 6pm to 11pm), the price multiplier is always 1.1. In other hours of a weekday, the price multiplier is always as low as 1.0, as transportation area such as airport terminal is a place where most drivers are willing to go because of stable and predictable demand. The highest price multiplier appears at the end of Sundays (in most cases it is 1.4): most passengers are eager to go back home to prepare for Monday morning.

The variation in business area (Fig. 3(a)), on the other hand, seems to be more random. There is no clear temporal pattern in price multiplier, except that during weekends the multiplier is lower (smaller than 1.1x in most cases). The reason for the lack of regularity is that there are different intentions for passengers leaving the business area including, for example, business visit, travel, shopping, going to/leaving from workplaces, etc. Passengers of different intentions have varying demand patterns, and hence the price multiplier does not show a high regularity.

For residential area (Fig. 3(b)), the regularity stands in between the other two areas. In order words, it shows a combination of randomness and regularity.

The differences of regularity in price multipliers inspires us that we should define a metric to characterize the regularity, and that we should consider using different algorithms to predict future price multipliers.

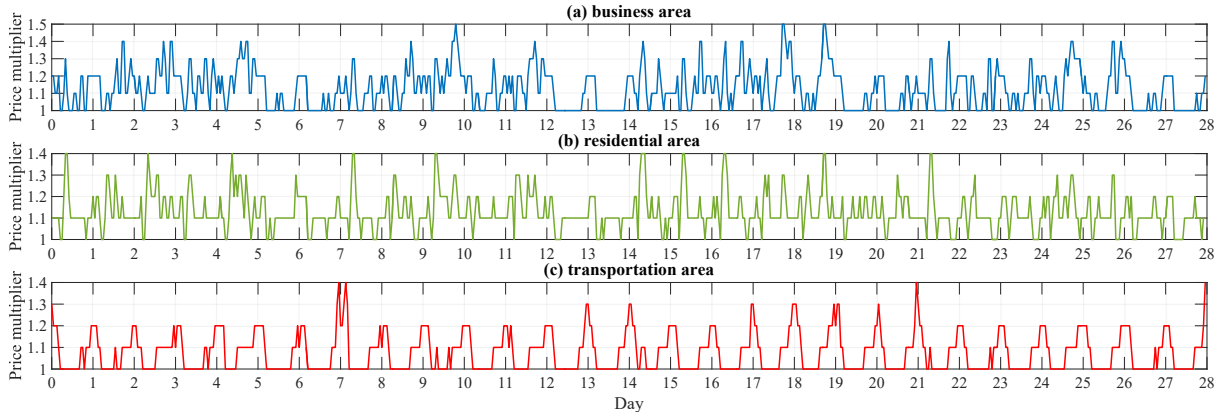

Figure 3: The variation of price multipliers in different functional areas.

3.3 Maximum Predictability Π^{max}

To characterize the regularity of price multiplier, we try to answer the following problem, and define the maximum predictability Π^{max}:

Problem 1. *Given a sequence of the price multipliers* $R^{(i,j)} = \{R_k^{(i,j)}, k = 1, 2, ..., L, L \leq K\}$ *of cell* (i, j)*, from time slot* $1, 2, ...L$*, considering both the randomness and the temporal correlation of the price multipliers, we want to find out the maximum predictability* $0 \leq \Pi^{max} \leq 1$ *(the highest potential accuracy of prediction) that any prediction can reach.*

Note that in the problem definition, we define the predictability on a sub-sequence of price multipliers: the total length of our data is K, and the length of the sub-sequence is $L \leq K$. This is because in prediction algorithm that will be discussed later, we need to train the algorithm based on a training set, which is usually a sub-sequence of the original data. Tab. 1 summarizes the main symbols used in this paper.

3.3.1 Entropy. The problem of maximum predictability has been discussed in a number of previous work about the prediction of, for example, taxi demand, human mobility, etc. The calculation of Π^{max} is based on the entropy of the price multiplier sequence, and using different entropies have different meaning on the predictability. Here we discuss the Shannon entropy and the real entropy.

Shannon Entropy. The Shannon entropy measures only the uncertainty of price multipliers, and does not take into account the temporal correlation:

$$S_{Shannon}^{(i,j)} = -\sum_{m=1}^{N^{(i,j)}} p(r_m^{(i,j)}) \log_2 p(r_m^{(i,j)}), \quad (1)$$

with $p(r_m^{(i,j)})$ the probability of multiplier $r_m^{(i,j)}$ in cell (i, j).

Real Entropy. The concept of real entropy has been discussed in [11, 13], and here it tries to consider both the uncertainty of price multipliers and the temporal correlation:

$$S_{real}^{(i,j)} = -\sum_{s^{(i,j)} \in S^{(i,j)}} p(s^{(i,j)}) \log_2 p(s^{(i,j)}) \quad (2)$$

Table 1: Main symbols used in this paper.

Symbol	Meaning
Δt	the length of each time slot
K	the total length of price multiplier sequence
$R^{(i,j)}$	the (sub)-sequence of price multipliers at cell (i, j): $R^{(i,j)} = \{R_k^{(i,j)}, k = 1, 2, ..., L, L \leq K\}$
L	the length of the (sub)-sequence $R^{(i,j)}$
$N^{(i,j)}$	the number of distinct price multipliers at cell (i, j)
$r_m^{(i,j)}$	the distinct price multipliers in $R^{(i,j)}$, $1 \leq m \leq N^{(i,j)}$
$S_{Shannon}^{(i,j)}$	Shannon entropy of $R^{(i,j)}$
$S_{real}^{(i,j)}$	real entropy of $R^{(i,j)}$
$S^{(i,j)}$	the set of time-ordered sub-sequences of $R^{(i,j)}$, $S^{(i,j)} = \{s^{(i,j)} \mid s^{(i,j)} \subseteq R^{(i,j)}\}$
$s_k^{'(i,j)}$	the length of the shortest unseen sub-sequence of $R^{(i,j)}$ starting at time k
Π^{max}	the maximum predictability of any prediction algorithm

Here $p(s^{(i,j)})$ is the probability of finding a particular time-ordered sub-sequence $s^{(i,j)}$ in the price multiplier sequence $R^{(i,j)}$. We could see that unlike the Shannon entropy, the real entropy not only considers the frequency of different price multipliers in $R^{(i,j)}$, but also the order of the temporal patterns of the price multipliers. The calculation of (2) has an exponential complexity, and [13] has proposed an approximation based on Lempel-Ziv estimator:

$$S_{real}^{(i,j)} \approx (\frac{1}{L}\sum_{k=1}^{L} s_k^{'(i,j)})^{-1} \ln n \quad (3)$$

Here $s_k^{'(i,j)}$ refers to the length of the shortest unseen sub-sequence of $R^{(i,j)}$ starting at time k.

3.3.2 Maximum Predictability. We have already obtained two different entropy measures for each cell, and could then

calculate the corresponding predictability measure: the success rate that the most accurate algorithm could correctly predict the future price multiplier at a particular cell. The predictability measure Π is subject to the Fano's inequality, and could be calculated by [13]:

$$S = -\Pi \log_2(\Pi) - (1-\Pi)\log_2(1-\Pi) + (1-\Pi)\log_2(N^{(i,j)}-1) \quad (4)$$

In (4), we obtain $\Pi^{Shannon}$ and Π^{real} when we let $S = S^{(i,j)}_{Shannon}$ and $S^{(i,j)}_{real}$, respectively. It has been proven in [13] that $\Pi^{Shannon} \leq \Pi^{real}$, so that the maximum predictability is $\Pi^{max} = \Pi^{real}$.

The meaning of these two predictability measures, is the highest prediction accuracy with and without considering the temporal correlation of price multipliers (for Π^{real} and $\Pi^{Shannon}$, respectively). The fact that $\Pi^{max} = \Pi^{real}$ indicates that if we want to have more accurate prediction, we need to consider the temporal patterns of price multipliers.

3.3.3 Predictability in Different City Cells/Areas. For the $N * N$ ($N = 100$) cells, we count the number of fare estimations in each cell during the 4-week period, and only consider those cells with the number of fare estimations higher than the median value. In other words, we exclude the cells without enough passenger activity – these cells maybe unreachable areas such as mountains, lakes, parks, etc. After this exclusion, the total number of cells to study is 3760.

In Fig. 4 we show the probability distribution function of the predictability measures in all these cells: the $\Pi^{max}(=\Pi^{real})$ and $\Pi^{Shannon}$ when we consider the whole dataset, and when considering only the weekdays in the 4-week period.

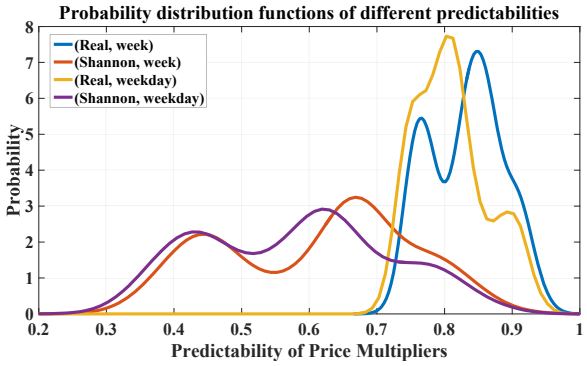

Figure 4: The PDFs of predictabilities in city cells.

We have the following observations from data:

(1) It is necessary to consider temporal correlation in price multipliers. The mean Π^{real} is 0.8315 and 0.8089 for all the days and for weekdays only; while the counterpart of $\Pi^{Shannon}$ is only 0.6097 and 0.5789.

(2) It is necessary to consider all the days instead of only weekdays. This is a trade-off between regularity and data size. Price multipliers of only weekdays may have more regularity, but meanwhile the data size is larger when considering all the days. Fig. 4 shows that considering the whole week provides higher predictability.

Table 2: Predictability in different functional areas.

	Business	Residential	Transportation
$\Pi^{Shannon}$	0.4739	0.6677	0.6107
Π^{real}	0.8322	0.8622	0.9554

In addition to city cells, we also calculate the $\Pi^{Shannon}$ and Π^{real} for the business, residential and transportation areas and show the results in Tab. 2. We have the following observations on the predictability in different functional areas:

(1) Predictability measures could also used to characterize city areas, and this agrees with our observation in §3.2 and Fig. 3. Business area has the lowest Π^{real} at 0.8322 and transportation area has the highest Π^{real} at 0.9554, and there is significant difference in Π^{real} between these functional areas.

(2) Again, considering temporal correlation in price multipliers is necessary. This brings a 75.61%, 29.13% and 56.44% increase in predictability for business, residential and transportation area, respectively.

4 PREDICTING PRICE MULTIPLIERS

In this section, we implement two predictors, namely a Markov-chain predictor and a neural network (NN) predictor, to predict future price multipliers based on historical data. We evaluate the performance and effect of these two predictors in different city cells and functional areas.

4.1 Markov-chain Predictor

We first introduce the Markov-chain predictor. In a Markov-chain predictor, the price multiplier in the next time slot is determined only by the price multiplier in the current time slot (first-order Markov-chain) or in the current and a few past time slots (high-order Markov chain).

We use the high-order Markov chain to predict price multipliers. The order is denoted by q. Assume that we already have the historical data $R^{(i,j)} = \{R^{(i,j)}_k, 1 \leq k \leq L\}$, an order-$q$ Markov-chain predictor predicts the next price multiplier $R^{(i,j)}_{L+1}$ based on the current and past price multipliers $R^{(i,j)}_{L-q+1}, ..., R^{(i,j)}_{L-1}, R^{(i,j)}_L$. The Markov property assumes that the transition probability $P(R^{(i,j)}_{L+1}|R^{(i,j)}_{L-q+1}, ..., R^{(i,j)}_{L-1}, R^{(i,j)}_L)$ is independent of the current time slot L:

$$P(R^{(i,j)}_{L+1}|R^{(i,j)}_{L-q+1}, ..., R^{(i,j)}_{L-1}, R^{(i,j)}_L) = P(R^{(i,j)}_t|R^{(i,j)}_{t-q}, ..., R^{(i,j)}_{t-2}, R^{(i,j)}_{t-1}), \forall q < t \leq L. \quad (5)$$

The transition probabilities form the transition matrix T, and each element of T, denoted by $T_{c,c'}$, is:

$$T_{c,c'} = P(R^{(i,j)}_t = c'|R^{(i,j)}_{t-q}, ..., R^{(i,j)}_{t-2}, R^{(i,j)}_{t-1} = c). \quad (6)$$

Note that in (6), c is a sequence of price multipliers of length q, and c' is a single price multiplier.

Algorithm 1 The Markov-chain Predictor

Input: the Markov-chain order q, historical data of price multipliers $R^{(i,j)} = \{R_k^{(i,j)}, 1 \le k \le L\}$, transition matrix T.

1: Extract the most recent q price multipliers c from $R^{(i,j)}$, i.e., $c = (R_{L-q+1}^{(i,j)}, ..., R_{L-1}^{(i,j)}, R_L^{(i,j)})$.
2: **if** c exists in T's rows **then**
3: Find in this row the largest value $T_{c,c'}$.
4: return c'.
5: **else**
6: Find the most frequent price multiplier appearing in $R^{(i,j)}$, denoted by c'.
7: return c'.
8: **end if**
9: Append c' to $R^{(i,j)}$ to form the new historical data $R^{(i,j)} = \{R_k^{(i,j)}, 1 \le k \le L+1\}$.
10: Update the transition matrix T based on $R^{(i,j)}$.

The probability transition matrix T is learned from historical data. We first find out every tuple of length q (i.e., a series of price multipliers in q consecutive time slots) from the historical data of length L (i.e., $R^{(i,j)} = \{R_k^{(i,j)}, 1 \le k \le L\}$). The set of distinct tuples are the rows of T. For each distinct tuple c, we find out all the possible next price multiplier in every occurrence of tuple c. The set of all distinct next price multipliers are the columns of T. For each distinct next price multiplier c', we obtain the corresponding element $T_{c,c'}$ of T – the probability of the next price multiplier c' occurring immediately after the presence of a tuple c.

Predicting the price multiplier in the next time slot is straightforward based on the learned probability transition matrix T and historical data. Assume that we already have the historical data $R^{(i,j)} = \{R_k^{(i,j)}, 1 \le k \le L\}$ of length L, and we first extract the most recent q price multipliers to form a tuple $c = (R_{L-q+1}^{(i,j)}, ..., R_{L-1}^{(i,j)}, R_L^{(i,j)})$. The next step is to search the rows of T for tuple c:

- If tuple c exists in T's rows, then we inspect this row and obtain the largest value $T_{c,c'}$. Then c' is the predicted next price multiplier.
- Otherwise, then tuple c has not appeared before, hence the matrix T could not decide what should be the next price multiplier. In this case, we choose the price multiplier that are the most frequently seen in the historical data to be the prediction result.

After having predicted the next price multiplier, this multiplier is appended to the historical data, which now becomes $R^{(i,j)} = \{R_k^{(i,j)}, 1 \le k \le L+1\}$. The transition matrix T is then updated based on the new historical data. Algorithm 1 summarizes the Markov-chain predictor.

4.2 Neural Network Predictor

In this subsection we introduce the neural network (NN) predictor. Unlike the Markov-chain predictor that predicts the next price multiplier solely based on the temporal correlation, in NN predictor we are able to include more factors that decides the prediction result.

In our implementation of the NN predictor, we consider four features. Note that we have stated in Tab. 1 that the length of each time slot is 1 hour, so that we actually collect hourly average price multiplier in each city cell/functional area. The features are as below:

(1) *hour-of-a-day*: the value ranges from 0 to 23, and it represents the hour of a particular day.
(2) *day-of-a-week*: the value ranges from 0 to 6, and it represents the day of a week (from Monday to Sunday).
(3) *daily precipitation*: we obtain the daily precipitation (in millimeters) in Beijing International Airport from a public data source. Note that this data may not be accurate enough, as the precipitation around the airport may not be the precipitation around a particular city cell or functional area. Also, we only the daily precipitation instead of hourly data, this adds up to the inaccuracy. But we consider this enough to catch the essence, and in the future we will try to get more accurate data to refine our NN predictor.
(4) *hourly temperature*: similarly, we also obtain the hourly temperature (in centigrades) in the airport from a public data source. This data is better than the precipitation data because (a) this is an hourly data and (b) the temperature around the airport is more representative of the temperature in other city cells/areas.

Our NN predictor uses a two-layer structure: a ReLU activation layer follows the input layer and then a Softmax output layer. The data fed to the input layer is a four-element tuple (i.e., values of the four features above), and the output from the Softmax layer is a categorical value (each possible value represents a possible price multiplier, so in our data we have 7 categories as the service provider sets the lower and upper bound of price multiplier to be 1.0 and 1.6).

4.3 Evaluation Setup

In this subsection we discuss how to evaluate the performance of our price multiplier predictors from three perspectives: preparing the training and test set, choosing evaluation metrics, and sampling city cells.

4.3.1 Training and Test Set. Both the Markov-chain predictor and NN predictor need training and test set. In our dataset, we split the 4-week data into 4 parts – one-week data for each part. We then use any one part as the test set, and the remaining three parts as the training set. As a result, the length of test set, L_{test}, is 168 ($= 24 * 7$) and the length of the training set, L_{train}, is 504 ($= 3 * 24 * 7$).

For the Markov-chain predictor, we choose the first three weeks of data as the training set, and the last week of data as the test set. We first generate the transition matrix T based on training set, and then successively predict the next price multiplier, until we reach the end of the 4^{th} week, while at the same time updating the matrix T in each prediction.

For the NN predictor, we run it for 500 times for each city cell or functional area, and in each run, we randomly choose

the test set (and the corresponding training set) and obtain the evaluation result as an average among multiple runs. In each run, all the parameters (i.e., weights and biases) of the neural network are trained based on the training set. The features corresponding to each hour in the week of the test set are then fed into the network, with the output as the test results to be compared by our ground-truth data.

4.3.2 Evaluation Metrics. The usual way to evaluate the performance of a prediction algorithm is based on the accuracy measure, i.e., how many of the predicted items are equal to the corresponding ground-truth items. In other words, if we use $R_{test}^{(i,j)} = \{R_{test,k}^{(i,j)}, k = 1, 2, ..., L_{test}\}$ to denote the price multiplier sequence in cell (i, j) of the test set and use $R_{pred}^{(i,j)} = \{R_{pred,k}^{(i,j)}, k = 1, 2, ..., L_{test}\}$ to denote the predicted price multiplier sequence, then the metric is:

$$Count_{equal}^{(i,j)} = \sum_{k=1}^{L_{test}} \delta_k^{(i,j)}, \tag{7}$$

where $\delta_k^{(i,j)} = 1$ if $R_{pred,k}^{(i,j)} = R_{test,k}^{(i,j)}$, and is 0 otherwise.

In predicting price multipliers, on the other hand, we don't care that much about the accuracy measure. In some cases, even though there is a slight difference between the predicted price multiplier and the ground truth, it is not a problem for passengers. For example, a user faced with a price multiplier 1.3 in a particular location may only want to know if there is any possibility to get a multiplier lower than 1.3 in neighboring locations or in a short time, but does not care about whether it is 1.1 or 1.2.

In our evaluation of the predictors, we use the symmetric mean absolute percentage error (sMAPE) [14], an accuracy measure based on the relative difference (error):

$$sMAPE^{(i,j)} = \frac{1}{L_{test}} \sum_{k=1}^{L_{test}} \frac{|R_{pred,k}^{(i,j)} - R_{test,k}^{(i,j)}|}{R_{pred,k}^{(i,j)} + R_{test,k}^{(i,j)}} \tag{8}$$

To sum up, a higher sMAPE means lower prediction accuracy, and a lower $Count_{equal}$ means the same.

4.3.3 Sampling of City Cells. We evaluate the two predictors in both functional areas and city cells. The limited number of functional areas in our study makes it easy to evaluate the predictors, but as we mentioned earlier in §3.3.3, we still have 3760 city cells to study after excluding those cells with few fare estimation events.

We sample some representative city cells for the evaluation. We first sort all the city cells according to the ascending order of their maximum predictability Π^{max}, and then split them into ten groups of equal sizes. In our case, each group contains 376 city cells. For examle, the first group (denoted by "0%-10%") means the bottom 10% cells according to Π^{max}. For each group, we take out the median 10 cells as samples, and use the average sMAPE of these 10 cells as the sMAPE of this group of cells. In all we have 100 sample cells. Tab. 3 summarizes the median Π^{max} of sample cells in each group.

Table 3: Π^{max} of sample cells in each group.

group	median Π^{max}	group	median Π^{max}
0%-10%	0.7460	50%-60%	0.8436
10%-20%	0.7632	60%-70%	0.8564
20%-30%	0.7799	70%-80%	0.8705
30%-40%	0.8107	80%-90%	0.8932
40%-50%	0.8306	90%-100%	0.9196

4.4 Performance Evaluation of Predictors

In this subsection, we evaluate the performance of the two predictors, on both city cells and functional areas.

In the following evaluation, the order used in the Markov-chain predictor (§4.1) is set to $q = 3$, as we find from running the predictor that increasing q over this value does not significantly increase the accuracy of the prediction. For the NN predictor (§4.2), the dimension of the input layer is 4, as we have four features; the number of neurons in the ReLU activation layer is 5, and this is the result from our parameter tuning; the number of neurons in the Softmax output layer is 7, as the range of price multiplier is from 1.0 to 1.6.

We first run both predictors on each of the sampled cells for 500 times, and calculate the average sMAPE values. In Fig. 5 we show the sMAPE values for both predictors in the ten sampled groups: group 1 means the "0%-10%" cells; group 2 means the "10%-20%" cells, and so on.

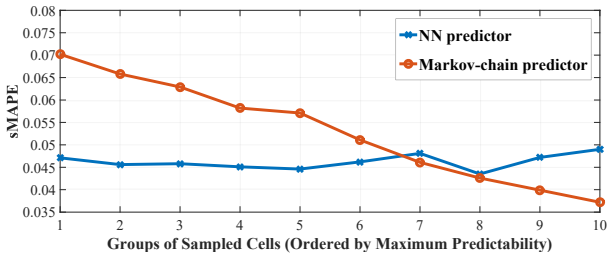

Figure 5: The sMAPE of two predictors in city cells.

Functional areas are much larger than city cells, and we also run both predictors on the three functional areas: business, residential and transportation area. In Tab. 4 we show the result, including the sMAPE and $Count_{equal}$ (only for reference), of running both predictors.

The Markov-chain predictor has a prediction accuracy changing obviously with the corresponding maximum predictability. For city cells with highest median of $\Pi^{max} = 0.9196$, the average sMAPE of the prediction is as low as 0.0372, and for those with lowest median of $\Pi^{max} = 0.7460$, the average sMAPE climbs to 0.0702 – 88.7% higher.

The NN predictor's behavior is just the opposite: the prediction accuracy is much more stable, and the sMAPE fluctuates only slightly. In Fig. 5, the highest and the lowest sMAPE are 0.0490 and 0.0435, respectively. We could safely claim that the maximum predictability does not have a big impact on the prediction accuracy in the NN predictor.

Table 4: The sMAPE of two predictors in functional areas.

Markov-chain predictor			
	Business	**Residential**	**Transportation**
$Count_{equal}$	47.3	54.5	71.6
$sMAPE$	0.0548	0.0468	0.0366
Neural network predictor			
	Business	**Residential**	**Transportation**
$Count_{equal}$	90.2	86.3	82.4
$sMAPE$	0.0448	0.0457	0.0513

Comparing the sMAPE of the two predictors in Fig. 5, if a city cell's $\Pi^{max} <= 0.8564$, the NN predictor has smaller sMAPE, hence higher prediction accuracy; otherwise, the Markov-chain predictor has higher prediction accuracy.

Regarding the prediction accuracy in functional areas, we compare the evaluation results in business and transportation area. Residential area stands in between, so comparing the two extremes is enough. We have the following discussion:

(1) For a functional area with high predictability (e.g., the transportation area), while the NN predictor brings a small increase of 15.1% in $Count_{equal}$, it results in a sMAPE 40.2% higher, compared to the Markov-chain predictor.
(2) For a functional area with low predictability (e.g., the business area), the NN predictor not only increases the $Count_{equal}$ by 90.3%, but also results in a 18.2% lower sMAPE.

Considering the fact that the NN predictor needs more computation time (in our case about 20 times the Markov-chain predictor's computation time) and more feature data (i.e., the temperature and precipitation data in our case), we conclude that only in functional area with low predictability the NN predictor is necessary, and the Markov-chain predictor provides enough prediction accuracy in other functional areas. This justifies our earlier claim that we should use different prediction algorithms in different city cells/functional areas.

4.5 Effects of Price Multiplier Prediction

In this subsection, we put aside the price multiplier prediction problem, and return to our initial goal of predicting price multipliers – relieving the anxiety from dynamic pricing.

The anxiety comes from passengers' uncertainty about the price multipliers nearby, or within a short time. We envision that our work on price multiplier prediction could reduce this uncertainty and make passengers informed about "*how to get a lower price multiplier?*".

We try to answer the following two questions:

- **Q1**: *If there are indeed chances to get a lower price multiplier in neighboring cells at the same time, by what probability could our price multiplier prediction find out such a chance?*
- **Q2**: *With our price multiplier prediction, by what probability a passenger in a particular location could really get a lower price multiplier in neighboring cells at the same time?*

Before answering these questions, we first define "neighboring cells": for cell (i, j), its neighboring cells are $Neighbor^{(i,j)} = \{(i', j')|i - 1 \leq i' \leq i + 1, j - 1 \leq j' \leq j + 1\}$.

We only consider those passengers in business and transportation areas, as these are two representative functional areas with low and high price multiplier predictability, respectively. We use the Markov-chain predictor in transportation area, and the NN predictor in business area.

For **Q1**, if we indeed find from our data that a passenger standing in a cell in a particular functional area is able to get a lower price multiplier in 8 neighboring cells, our evaluation shows that on average in the transportation area the probability that our predictor could find out such a chance is 89.4%, and in the business area the probability is 82.3%.

For **Q2**, the difference with **Q1** is that we have to consider the fact that in some cells there is not any lower price multiplier in neighboring cells. Similarly, on average in transportation area the probability a passenger could use the price multiplier prediction to get a lower price in neighboring cells is 18.7% – because in transportation area the price multiplier is relatively stable and low (see §3.2). In business area, the probability is much higher and reaches 35.4%.

Furthermore, for **Q2** we also try to focus only on those cells on the edge of a functional area. In practice, passengers in these cells are more easier to get lower multipliers, as walking away to a neighboring cell may enter a different functional area. Our evaluation shows that in the edge cells of transportation area, the corresponding probability rises to 31.2%. The probability in business area rises to 67.3%.

These results verify that (a) price multiplier prediction is effective in obtaining lower prices and thus relieving passengers' anxiety and (b) passengers standing on the edge of functional areas are more easier to find lower prices.

5 RELATED WORK

We discuss related work from three perspectives.

Ride-on-demand Service. Most studies on emerging RoD services are centered on dynamic pricing. [3] tries to evaluate Uber's surge pricing mechanism based on the measurement treating Uber as a black-box, and to predict the prices in the coming minutes or nearby locations based on the learned relationship between price multipliers and supply & demand. In our previous work [6, 7] the demand pattern, the effect of dynamic pricing, and passengers' reaction to dynamic pricing in RoD service have been carefully studied and analyzed. Other related studies focus on economic analysis of the effect and impact of dynamic pricing [8], the supply elasticity [4] and consumer surplus [5] in Uber, etc.

Taxi demand prediction. Our work on price multiplier prediction is inspired by previous work on taxi demand prediction. The availability of public taxi dataset leads to a number of related studies. Examples include using neural network to forecast the taxi demand from historical data [10], using SVM to select the most related feature that determines

the taxi demand [9], using taxi GPS trajectories to detect anamalous trips [2], etc.

Temporal patterns and predictability. Many studies have found that urban human mobility have strong regularities: people usually go to work, go back home, go shopping/entertaining at specific time and locations. For example, [1] finds this out by studying cell phone user's location records and shows that human mobility exhibits a high degree of spatio-temporal pattern. Furthermore, [13] employs the concept of maximum predictability, and uses it to study the temporal pattern of the individual human mobility. Similarly, [15] also uses this concept to choose different algorithms to predict taxi demand at each building block in New York.

6 CONCLUSION AND FUTURE WORK

In this paper we use price multiplier prediction to relieve the anxiety brought by dynamic pricing in RoD services. The core problem passengers worry about is "*Could I get a lower price multiplier within a short time/distance?*", and price multiplier prediction helps to answer it.

Based on the data collected from a service provider in China, we divide the city of Beijing into cells and functional areas, and use the metric *maximum predictability* Π^{max} to characterize the predictability of price multipliers in each cell/area. Data analysis shows that the mean Π^{max} around the city is 0.8315 (i.e., around 83.15% price multipliers could be predicted), and that Π^{max} varies significantly between cells/areas. It also proves to be necessary to consider temporal correlation in predicting price multipliers, and this brings an increase in Π^{max} up to 75.61% in business area.

With the Π^{max} of city cells/areas, we then implement two predictors, the Markov-chain and neural network (NN) predictor, and evaluate their performance in city cells/areas. The prediction accuracy of the NN predictor remains stable with different Π^{max}, but the accuracy of the Markov-chain predictor decreases significantly with lower Π^{max}. We thus claim that the NN and Markov-chain predictor should be used in areas with low (e.g., business area) and high (e.g., transportation area) Π^{max}. Finally, we evaluate the effect of prediction, and results show that our predictors could find out lower price multipliers nearby with a probability as high as 89.4%. Also, the probability that a passenger in business area could use price prediction to get lower prices nearby is 35.4% and this rises to 67.3% if considering only edge cells.

For the future work, we will collect more data such as wind speed or local events information to refine the NN predictor and improve the accuracy. The effects of price prediction will be evaluated more comprehensively. We will also use deep learning techniques to make our prediction more accurate.

ACKNOWLEDGMENTS

This work is supported by National Key Research and Development Project of China (Grant No. 2017YFB1002000), the National Science Foundation of China (No. 61602067), and the Fundamental Research Funds for the Central Universities (No. 106112017cdjxy180001).

REFERENCES

[1] Marta C. Gonzalez, Cesar A. Hidalgo, and Albert-Laszlo Barabasi. 2008. Understanding individual human mobility patterns. *Nature* 453, 7196 (2008), 779–782.

[2] Chao Chen, Daqing Zhang, P.S. Castro, Nan Li, Lin Sun, and Shijian Li. 2011. Real-Time Detection of Anomalous Taxi Trajectories from GPS Traces. In *8th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous 2011)*. Springer, 63–74.

[3] Le Chen, Alan Mislove, and Christo Wilson. 2015. Peeking Beneath the Hood of Uber. In *Proceedings of the 2015 ACM Conference on Internet Measurement Conference (IMC '15)*. ACM, New York, NY, USA, 495–508.

[4] M. Keith Chen. 2016. Dynamic Pricing in a Labor Market: Surge Pricing and Flexible Work on the Uber Platform. In *Proceedings of the 2016 ACM Conference on Economics and Computation (EC '16)*. ACM, New York, NY, USA, 455–455.

[5] Peter Cohen, Robert Hahn, Jonathan Hall, Steven Levitt, and Robert Metcalfe. 2016. Using Big Data to Estimate Consumer Surplus: The Case of Uber. (2016). Retrieved May 8, 2017 from http://bit.ly/2pqXiWo

[6] Suiming Guo, Yaxiao Liu, Ke Xu, and Dah Ming Chiu. 2017. Understanding Passenger Reaction to Dynamic Prices in Ride-on-demand Service. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2017 IEEE International Conference on*. IEEE, 42–45.

[7] Suiming Guo, Yaxiao Liu, Ke Xu, and Dah Ming Chiu. 2017. Understanding Ride-on-demand Service: Demand and Dynamic Pricing. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2017 IEEE International Conference on*. IEEE, 509–514.

[8] Jonathan Hall, Cory Kendrick, and Chris Nosko. 2015. The effects of Uber's surge pricing: a case study. (Oct. 2015). Retrieved Feb 10, 2017 from http://bit.ly/2kayk9O

[9] Bin Li, Daqing Zhang, Chao Chen, Shijian Li, Guande Qi, and Qiang Yang. 2011. Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2011 IEEE International Conference on*. IEEE, 63–68.

[10] Xiaolong Li, Gang Pan, Zhaohui Wu, Guande Qi, Shijian Li, Daqing Zhang, Wangsheng Zhang, and Zonghui Wang. 2012. Prediction of urban human mobility using large-scale taxi traces and its applications. *Frontiers of Computer Science* 6, 1 (2012), 111–121.

[11] Thomas M. Cover and Joy A. Thomas. 2006. *Elements of Information Theory* (2nd ed.). Wiley-Interscience.

[12] Shenzhou UCar. 2015. Annual results for the year ended 31 Dec 2015. (2015). http://bit.ly/2cFdL6U

[13] Chaoming Song, Zehui Qu, Nicholas Blumm, and Albert-Laszlo Barabasi. 2010. Limits of Predictability in Human Mobility. *Science* 327, 5968 (2010), 1018–1021.

[14] Wikipedia. 2017. Symmetric mean absolute percentage error. (2017). http://bit.ly/2umuNKT

[15] Kai Zhao, Denis Khryashchev, Juliana Freire, Claudio Silva, and Huy Vo. 2016. Predicting Taxi Demand at High Spatial Resolution: Approaching the Limit of Predictability. In *Big Data, 2016 IEEE International Conference on*. IEEE, 833–842.