A Simple but Quantifiable Approach to Dynamic Price Prediction in Ride-on-demand Services Leveraging Multi-source Urban Data

SUIMING GUO, Jinan University, China
CHAO CHEN*, Chongqing University, China
JINGYUAN WANG, Beihang University, China
YAXIAO LIU, Tsinghua University, China
KE XU, Tsinghua University, China
DAQING ZHANG, Peking University, China
DAH MING CHIU, The Chinese University of Hong Kong, China

Ride-on-demand (RoD) services such as Uber and Didi are becoming increasingly popular, and in these services dynamic prices play an important role in balancing the supply and demand to benefit both drivers and passengers. However, dynamic prices also create concerns. For passengers, the "unpredictable" prices sometimes prevent them from making quick decisions: one may wonder if it is possible to get a lower price if s/he chooses to wait a while. It is necessary to provide more information to them, and predicting the dynamic prices is a possible solution. For the transportation industry and policy makers, there are also concerns about the relationship between RoD services and their more traditional counterparts such as metro, bus, and taxi: whether they affect each other and how.

In this paper we tackle these two concerns by predicting the dynamic prices using multi-source urban data. Price prediction could help passengers understand whether they could get a lower price in neighboring locations or within a short time, thus alleviating their concerns. The prediction is based on urban data from multiple sources, including the RoD service itself, taxi service, public transportation, weather, the map of a city, etc. We train a simple linear regression model with high-dimensional composite features to perform the prediction. By combining simple basic features into composite features, we compensate for the loss of expressiveness in a linear model due to the lack of non-linearity. Additionally, the use of multi-source data and a linear model enables us to quantify and explain the relationship between multiple means of transportation by examining the weights of different features in the model. Our hope is that the study not only serves as an accurate prediction to make passengers more satisfied, but also sheds light on the concern about the relationship between different means of transportation for either the industry or policy makers.

CCS Concepts: • **Information systems** → **Location based services**; **Data mining**; • **Human-centered computing** → *Empirical studies in ubiquitous and mobile computing*;

Additional Key Words and Phrases: Prediction, Ride-on-demand service, Dynamic pricing, Urban Transportation

*This is the corresponding author.

Authors' addresses: Suiming Guo, College of Information Science and Technology, Jinan University, Guangzhou, China; Chao Chen, cschaochen@cqu.edu.cn, Chongqing University, Chongqing, China; Jingyuan Wang, Beihang University, Beijing, China; Yaxiao Liu, Tsinghua University, Beijing, China; Ke Xu, Tsinghua University, Beijing, China; Daqing Zhang, Peking University, Beijing, China; Dah Ming Chiu, The Chinese University of Hong Kong, Hong Kong, China.

2474-9567/2018/9-ART112 $15.00
https://doi.org/10.1145/3264922

ACM Reference Format:
Suiming Guo, Chao Chen, Jingyuan Wang, Yaxiao Liu, Ke Xu, Daqing Zhang, and Dah Ming Chiu. 2018. A Simple but Quantifiable Approach to Dynamic Price Prediction in Ride-on-demand Services Leveraging Multi-source Urban Data. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 112 (September 2018), 24 pages. https://doi.org/10.1145/3264922

1 INTRODUCTION

Emerging Ride-on-demand (RoD) services such as Uber and Didi are becoming increasingly popular in recent years. They attract passengers by their convenience, as well as flexible and affordable prices; and attract drivers who want to drive more flexibly with their own cars.

Dynamic pricing is the core and distinctive feature in RoD service, and it reflects the effort in balancing the supply (the number of cars on the road) and demand (the number of passengers' requests): a higher price reduces demand and increases supply in a busy area, and a lower price does the opposite in a non-busy area. This makes the service more responsive for both drivers and passengers. To be more specific, dynamic pricing is always represented by a "price multiplier": the price of a trip is the product of the price multiplier (based on the supply & demand condition nearby) and a fixed normal price (based on the estimated distance & time of the trip). The fixed normal price is similar to the price of a taxi trip, so we only focus on the price multiplier in this paper.

Despite the convenience and flexibility, dynamic pricing exerts mental burden on passengers and makes them less satisfied. In traditional taxi service with fixed pricing, passengers can estimate the trip fare based on personal experience. In emerging RoD service, however, they have an extra task before making decisions: guessing the price multipliers based on their estimate of the supply & demand condition nearby. For an individual passenger without relevant information, the estimate is invariably inaccurate and usually prevents them from making decisions at ease. Giving more information to passengers help ease the burden, including, for example, 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 them, the most direct one is price prediction, and passenger could rely on the results to make quick decisions.

Predicting dynamic prices can be helpful for passengers. Any third-party can use the result of price prediction to benefit passengers. For example, a mobile-app can be developed to help passengers make decisions at ease and compare the price multipliers among one or more service providers. Specifically, the app can use the price prediction results to answer the following questions:

- For each service provider, what's the dynamic price for one passenger request (at a particular time and location)?
- For each service provider, what's the dynamic price for this request if the passenger can wait for a while, or can walk away for hundreds of meters to get a car?

With the above information, one can then make decisions at ease by knowing: (a) by choosing which provider s/he can get the lowest price and (b) whether it is a good time and location to request a ride.

Dynamic price prediction has not received much attention in RoD services. In general, there are two ways of predicting prices. One way is to predict the supply & demand and then guess the relationship between dynamic prices and the supply & demand, as seen in [10]. Because most RoD services keep their dynamic pricing algorithms as secrets, guessing this relationship from data is not accurate enough to generate a good prediction. Furthermore, the prediction of supply & demand itself also brings some inaccuracies. Another way omits the details in between, and predicts the price multiplier directly based on historical data, including the price multipliers and features relevant to supply & demand. This way does not try to unveil the "secret algorithms", but is easier to generalize: the prediction is achievable as long as one can collect historical data, regardless of the service provider-specific algorithms. Our past work [18] uses methods such as time-series analysis and neural networks to predict dynamic prices, but only at the granularity of a city cell or functional area. We also do the prediction in this manner here.

The adoption of dynamic pricing distinguishes RoD services with traditional means of transportation such as bus, metro, and especially, taxi, but meanwhile it also raises concerns about the relationship between them. For example, there are always worries among taxi practitioners that new RoD services would undercut their business, and this is part of the reasons why Uber, the most influential RoD service provider in the world, is involved in some disputes, protests and lawsuits in different cities. As another example, there are also concerns if RoD services are killing other means of public transportation such as bus or metro. These concerns necessitate a study on the relationship between them: whether different means of transportation affect each other and how.

In this paper, we address the above concerns by predicting dynamic price multipliers based on multi-source urban data. Features are extracted from urban data, and a linear regression model is trained to predict price multipliers. The rationale behind using multi-source urban data and linear regression model is explained as follows:

Multi-source Urban Data. We collect urban data from multiple sources, including the RoD service itself, taxi service, public transportation, weather, the map of a city, etc. This helps us to:

- extract more features from data, and to make our prediction more accurate;
- understand the relationship between different means of transportation by combining features from different data sources together.

Linear Regression Model with High-dimensional Features. There are generally two paradigms of predictive models: (a) complicated non-linear models with a small number of features [15, 25] and (b) simple linear models with a large number of features [24, 35, 41]. While the inherent non-linearity in the former paradigm helps to describe a model accurate enough with a few features, it is hard to determine how different features contribute to the prediction result. By comparison, in a linear regression model one can clearly determine the contribution of different features based on the corresponding weights. To make the prediction result more accurate with a linear model, we make use of our multi-source urban data and integrate features from different data sources to form new, high-dimensional composite features. This feature-integration procedure generates new sets of features with significantly higher dimension, and boosts the descriptive power of a linear model by characterizing the relationship between different features through composite features. Furthermore, while the use of multi-source urban data helps us to understand the relationship between means of transportation, the use of a linear regression model helps us to step further and to quantify the relationship.

Contributions. Our contributions are three-fold:

- This paper is in the series of our study on RoD services, and trains a prediction model for dynamic prices at a much finer granularity, following [18]. Different from [18] that only predicts the hourly average price multiplier at the granularity of city cells and functional areas, now we predict the dynamic prices in RoD services on the basis of individual user request in any location. We train a linear regression model with features of thousands of dimensions, and discuss the contribution of various features.
- We introduce multi-source urban data in price multiplier prediction, as price multipliers are influenced by many different factors in addition to time and location. Specifically, we consider the influence of other transportation services on the dynamic prices in RoD service, including bus, metro and taxi. This not only makes our prediction more accurate, but enables us to quantify the relationship between different means of transportation as well. We also use POI information

from the map of city to characterize the locations passengers requesting for rides. The description and features from POI information make our characterization more accurate and having a finer granularity.
- To the best of our knowledge, this is the first effort that tries to involve multi-source urban data to quantify the relationship between different means of transportation. Besides presenting an accurate prediction result and providing insights about different features, we also discuss quantitatively how the availability of taxi and the presence of bus and metro affect the dynamic prices of RoD services. Besides technical results, we hope our effort could help the industry or policy makers regarding the concern about the relationship between different means of transportation.

The remainder of the paper is organized as follows. We review related work in §2. In §3 we describe the multi-source urban datasets used in the study, and §4 introduces our feature engineering. §5 presents the linear regression model. We evaluate our model and discuss the relationship between different means of transportation in §6. Finally, §7 concludes the paper.

2 RELATED WORK

Ride-on-demand Service. Most studies on RoD services are centered on dynamic pricing. [10] tries to evaluate Uber's surge pricing mechanism based on the measurement treating Uber as a black-box, and predicts future prices based on a guessed relationship between price multiplier and supply & demand. The prediction is not accurate enough, due to the lack of real service data and the inaccuracies in guessing the relationship. The authors in [19–21] study and analyze the demand, the effect of dynamic pricing and passengers' reaction to prices in RoD services. In [18] the authors present a preliminary study on predicting price multipliers, but at a coarser granularity spatio-temporally. Specifically, [18] only uses the RoD service data and weather data to predict the *hourly average price multiplier* in city cells or specific city functional areas. The authors define a metric to characterize the variation pattern and the predictability of price multipliers in different regions in the city, and use different predictors such as Markov-chain predictor or neural network predictor in different regions based on the defined metric. Their work is a reflection of the varying price multipliers in different time and locations, and can tell passengers "when and where you may get a lower hourly average price multiplier". Other works focus on economic analysis of the effects of dynamic pricing [22], the supply elasticity [11], consumer surplus [12], etc.

Taxi and Other Transportation Services. Our work on price multiplier prediction is inspired by previous work on taxi demand prediction. [31] uses neural network to forecast the taxi demand from historical data; [29] uses SVM to determine the most related feature of taxi demand; [7] uses taxi GPS trajectories to detect anomalous trips; etc. The availability of public taxi dataset leads to a number of related studies. For example, [52] uses taxi trajectory data to detect flawed urban planning, and [48] recommends driving directions based on patterns mined from historical taxi trajectory data. Besides, data from taxi and other transportation services have also been used in traffic speed prediction [43], event detection [3], city pattern prediction [16], city structure discovery [42], human mobility [6, 37, 38], safe driving [47], crowd management [14, 49] and taxi ride-sharing [23, 33, 36]. Crowdsensing is a heated research topic in studying transportation services in recent years, and can help to collect multi-source urban data, especially the service data of transportation services. [17] reviews the features, frameworks and applications of mobile crowd sending. [44] studies how to leverage the spatial and temporal correlation of the data to reach an equilibrium between sensing cost and data quality. [8] leverages the crowd power of taxi drivers to achieve city-wide package deliveries. [9, 46] discusses two application scenarios of crowd sensing.

Dynamic Pricing and Concerns. Dynamic pricing is not an invention in RoD service, and it has been used in lots of services and scenarios to either improve service efficiency or manipulate supply and demand in different forms. For examples, it has been used in Internet retail [5], inventory management [4], hotel booking [28] and airline pricing [34]. For the RoD service, the mental burden created by dynamic prices have been discussed previously. [21] shows that during morning rush hours, the probability of finding a lower price multiplier within 1km is about 75.99%. The probability is 76.10% and 34.21% for evening rush hours and non-rush hours. [20] shows that only in 39.77% cases a passenger accepts the price multiplier after only one fare estimation. Concerns about the relationship between RoD service and taxi service or public transportation could also be found in news reports such as [1, 27, 32].

3 MULTI-SOURCE URBAN DATASETS

In this section we present the multi-source urban datasets used in predicting price multiplier, including the event-log data from a RoD service, the GPS trajectory data from taxi service, the bus & metro distribution data, the POI data and the weather data. Tab. 1 summarizes our datasets and their fields.

3.1 RoD Service Event-log Data

Our data of the RoD service is collected from Shenzhou UCar (http://www.10101111.com/), 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 [39].

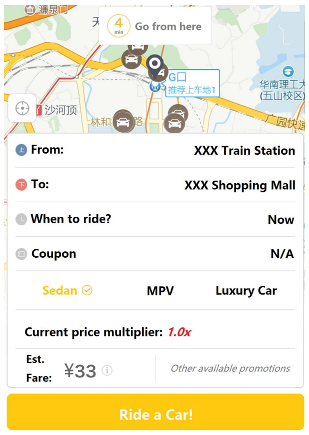
We first explain the user interface of the mobile app, as shown in Fig. 1, to illustrate the work-flow of a typical RoD service. A user usually opens the app and types the boarding location A and arriving location B. S/he could also choose "when to ride (now or several minutes later)" and "using coupon". After the user has specified the locations and chosen all available options, the app sends the relevant information back to the service provider and obtains (a) the estimated trip fare and (b) the current dynamic price multiplier, which are displayed to the user. Note that the service provider often sets a lower and upper bound on the price multiplier in the service policy. The user 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.

Each time when the mobile app sends all the information to the service provider and returns the current price multiplier and the estimated trip fare, an ***EstimateFee*** event is generated, and this is the source of our event-log dataset. Our dataset contains the complete record of ***EstimateFee*** events in the complete 4 months from Aug to Nov, 2016 in Beijing. Each entry corresponds to a single event, and includes fields such as *event_time*, *event_location* (longitude and latitude), *estimated_fare*, *price_multiplier*, *passenger_device_IMEI*, etc. The dataset contains 14,587,353 entries, and all are properly anonymized.

In the dataset, we find out 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 $U = 1.6$. So all possible multipliers are 1.0, 1.1, 1.2, 1.3, 1.4, 1.5 and 1.6.

In this subsection, we present some results that show how different price multipliers could be in different locations or during different time periods. We first show the variation of hourly average price multiplier at the level of city functional areas. We select some typical business (i.e., the place for working), residential (i.e., the place for living) and transportation (e.g., airport terminals and railway stations) areas in Beijing, and show the variation of hourly average price multipliers in Fig. 2. The criteria of selecting these typical functional areas could be found in [19] and is not discussed here.

We then divide the map of Beijing into 2500 ($= 50 * 50$) rectangular cells of the same size, and investigate the average price multiplier of all events taking place in each cell during some particular time



From: XXX Train Station
To: XXX Shopping Mall
When to ride? Now
Coupon N/A
Sedan MPV Luxury Car
Current price multiplier: *1.0x*
Est. Fare: ¥33 *Other available promotions*
Ride a Car!

Fig. 1. The user interface of a typical RoD service.

Table 1. A summary of datasets and fields.

Dataset	Fields
RoD	event_time, event_location, estimated_fare, price_multiplier, passenger_device_IMEI.
Taxi	upload_time, latitude, longitude, heading, speed, full_flag, car_plate.
Bus & metro	# of bus stations, # of bus lines, # of metro stations, # of metro lines.
POI	# of POIs of 14 categories (*car service*, *restaurant*, *shopping*, *sports & entertainment*, *hospital*, *hotel*, *scenic spot*, *business & residential building*, *government*, *education & culture*, *transportation facility*, *finance & insurance*, *business* and *lifestyle*).
Weather	temperature, wind speed, humidity, pressure, visibility, weather condition.

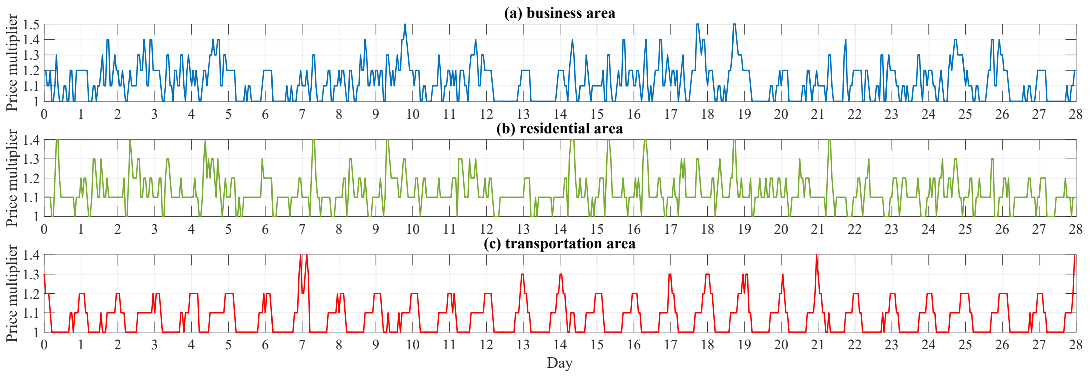


Fig. 2. The variation of hourly average price multipliers in different functional areas.

periods. Fig. 3 to 5 show the average price multiplier during morning rush hour (i.e., 8am to 9am), a non-busy hour (i.e., at noon) and evening rush hour (i.e., 6pm to 7pm) on weekdays. By comparison, the average price multiplier during [8am, 9am] on weekends is shown in Fig. 6. We also show the number of ***EstimateFee*** events taking place in each cell during morning (Fig. 7) and evening rush hour (Fig. 8) on weekdays. Note that the number of events is an approximate of the total demand, i.e., the sum of met and unmet demand from passengers. In Fig. 3 to 8, the deeper the color of a cell, the larger the corresponding metric (i.e., the average price multiplier or the number of events).

There are some basic observations we obtain from these figures:

- The regularity of the variation of price multipliers is closely related to the locations of passengers. In some location (e.g., transportation area) the variation is more regular, whereas in some locations (e.g., business area) it is more random.
- The average price multiplier of a location is related to hour-of-day, day-of-week, and the location.

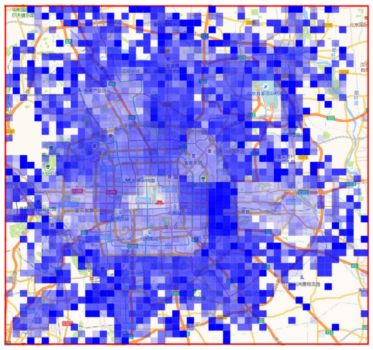


Fig. 3. The average price multiplier during [8am, 9am] on weekdays.

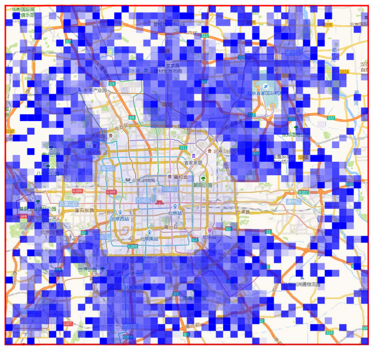


Fig. 4. The average price multiplier during [12pm, 1pm] on weekdays.

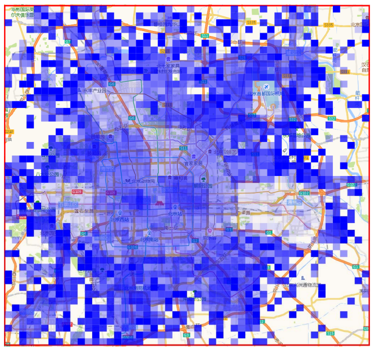


Fig. 5. The average price multiplier during [6pm,7pm] on weekdays.

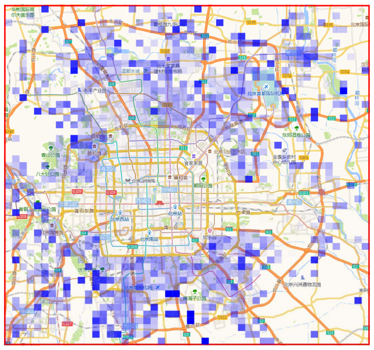


Fig. 6. The average price multiplier during [8am, 9am] on weekends.

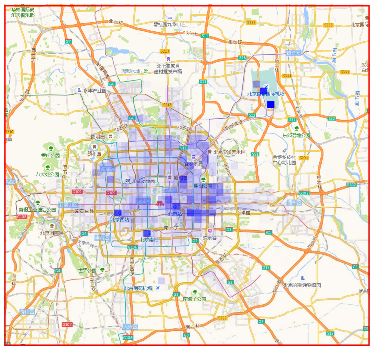


Fig. 7. The number of *EstimateFee* events during [8am, 9am] on weekdays.

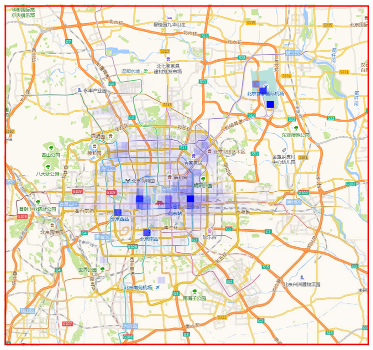


Fig. 8. The number of *EstimateFee* events during [6pm,7pm] on weekdays.

- Passengers' potential demand (i.e., the number of ***EstimateFee*** events) also varies significantly in different locations, hour-of-day, and day-of-week.

3.2 Taxi Service GPS Trajectory Data

In order to figure out the relationship between RoD service and the traditional taxi service, we also collect GPS trajectory data from the taxi service in Beijing. The taxi data helps us to (a) capture the operating status of taxi service in the city and (b) characterize the general traffic condition of different locations. For examples, "*whether a particular region is busy during a particular time period*", "*the number of available taxis around a particular region*", etc.

Our dataset covers the GPS trajectory data of about 30,000 taxis in Beijing in November, 2016. Each taxi uploads one GPS data entry about every 30 seconds during operation. For each day, the volume of dataset ranges from 45 to 50 million entries. Each entry contains the following fields:

- *upload_time*: the timestamp of this entry;
- *latitude* & *longitude*: the location of the taxi;
- *heading* & *speed*: the heading and driving speed of the taxi;
- *full_flag*: whether the taxi is full or available;

- *car_plate*: the MD5-encrypted string of the taxi's plate number.

With the GPS trajectory and especially the *full_flag* of a taxi, we can easily determine all the trips a particular taxi serves each day. Specifically, the *full_flag* changing from "available" to "full" indicates that a passenger is getting on a taxi; and the reverse direction indicates that a trip is finished.

With the GPS trajectory and trip information of taxis, we could then extract relevant features from the taxi service's dataset. This is explained in §4.1.

3.3 Bus and Metro Distribution Data

The distribution of bus & metro helps to characterize the availability of public transportation around different locations, and enables our study on the relationship between them and RoD service.

The most accurate description of the bus and metro distribution should be like "*the number of buses around a particular location during a particular time period*", and could be obtained by, for example, examining the smart-card usage data (i.e., "*how many people wipe their smart-card on a bus*") or collecting the GPS data of bus & metro. However, bus & metro have relatively fixed time tables, and most people decide whether to take public transportation based on the the availability of bus & metro lines/stations nearby, instead of the availability of bus & metro nearby. So we turn to an easier method to acquire our datasets by simply counting the number of bus & metro lines and stations nearby.

We crawl the above data from AMap service (one of the most popular digital map service providers in China) using its JavaScript API [2]. Specifically, for a location (i.e., a pair of longitude and latitude) given in an entry of the RoD service dataset, we count the number of bus & metro lines and stations within a 500-meter radius of the location. As a result, the volume of this dataset is the same as that of the RoD service dataset. For the whole city, there are more than 7,700 bus stations and about 380 metro stations, and we plot the distribution of bus & metro stations in Beijing in Fig. 9 and 10. In these figures, each point represents a bus or metro station.

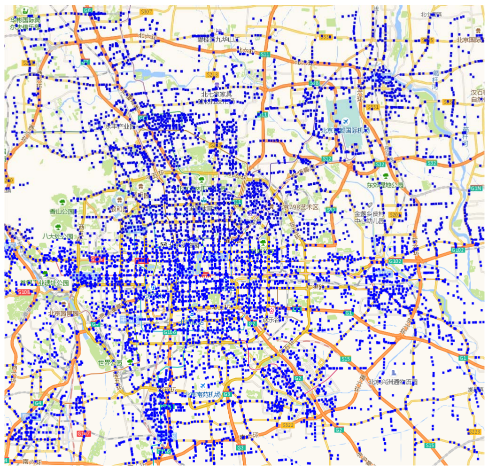


Fig. 9. The distribution of bus stations in Beijing.

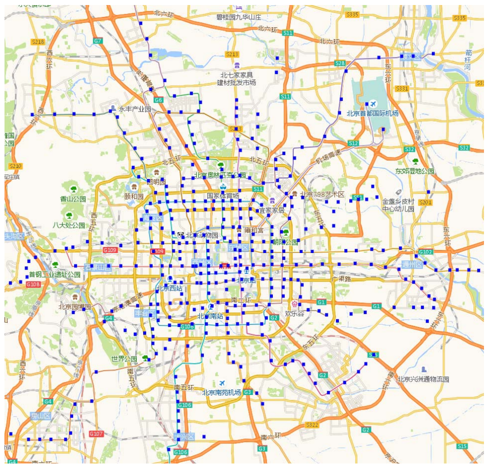


Fig. 10. The distribution of metro stations in Beijing.

3.4 POI Data

This dataset mainly contains the POI (point of interest) information. The goal of using POI information in our study, is that we hope some properly selected POI features could represent the location information given a pair of longitude and latitude values. As we point out in Fig. 3 to 8, either the price multiplier or the number of events is closely related to the location in which an ***EstimateFee*** event takes place. For example, the number of events is significantly higher around airport terminals or railway stations; the price multiplier is, on average, much higher in some business areas during evening rush hour than in other locations. We want to find our some features to accurately describe this sort of location information.

Similar to the bus and metro distribution data, we also crawl POI data from AMap service. This map service provider categorizes each POI on the map into 14 coarse categories: *car service*, *restaurant*, *shopping*, *sports & entertainment*, *hospital*, *hotel*, *scenic spot*, *business & residential building*, *government*, *education & culture*, *transportation facility*, *finance & insurance*, *business* and *lifestyle*. For a location (i.e., a pair of longitude and latitude) given in an entry of the RoD service dataset, we count the number of POIs of each of these 14 categories within a 500-meter radius of the location, and use the resulting vector as our POI data. The volume of the POI dataset is the same as that of the RoD service dataset.

Some previous work may associate a location with its nearest POI and use its category to describe the location. We consider this way to be not accurate enough. For example, a passenger is standing out of a big shopping mall and there are also some restaurants or lifestyle services around him. It is possible that a particular restaurant is the nearest POI, but the big shopping mall turns out to be the reason why the passenger is standing here requesting for the RoD service.

3.5 Weather Data

Weather should also be a factor that influences either the dynamic prices or the number of ***EstimateFee*** events. Intuitively, a higher demand is triggered when there is a bad weather, such as rain, fog, low visibility, extreme temperature, etc.

We turn to *timeanddate.com* for the weather data. We crawl the weather data in every 3 hours in the complete 4 months from August to November, 2016 in Beijing, corresponding to the time range of the RoD service data. The weather data includes the following fields: *temperature*, *wind speed*, *humidity*, *pressure*, *visibility* and *weather condition*. The first five fields have self-explanatory names, and *weather condition* categorizes the weather into the the following 17 types: *ice fog*, *partly sunny*, *sprinkles*, *scattered clouds*, *heavy rain*, *dense fog*, *sunny*, *clear*, *overcast*, *light rain*, *low clouds*, *haze*, *fog*, *rain*, *passing clouds*, *light fog* and *light snow*.

It is true that the weather dataset has a coarser granularity compared to the other four datasets. Firstly, we only have the weather of the whole city of Beijing, instead of having the weather information associated to each smaller region. Secondly, the weather information is updated every 3 hours. This is due to the availability of the weather history data, but we consider our crawled data is enough to make some sense: compared to other factors, the weather condition is a factor that affects a far larger area and its effect usually lasts much longer.

3.6 Data Collection

We provide some discussions on data collection here. In this paper we mainly discuss our methodology based on the collected multi-source urban data, and as long as the required datasets can be obtained, researchers can adopt our methodology to perform dynamic price prediction. Except the RoD service data, other datasets either can be crawled from public services or have many public datasets online. The RoD service data may not be readily available as it is still an emerging service, but we suggest that there are still some possible ways to collect RoD service data, and in the following we give some examples:

- The data can be collected from a service provider through, for example, open API or research collaboration.

- The data can be collected in a crowd-sourcing fashion. For example, researchers or developers can develop a mobile app that asks users to report their experience of using RoD service, and those who report regularly can enjoy the free price prediction service or other forms of incentives.
- Government agencies can build a platform of monitoring purpose, and require different service providers to disclose certain level of information with proper anonymization.

As a result, researchers, personal developers, government agencies or any other third-parties have different methods of collecting the required data and perform price prediction for passengers. The flexibility of data collection also makes our study on dynamic price prediction practical and meaningful.

4 FEATURE ENGINEERING

As we point out in §1, we use a linear regression model in order to quantify the contribution of each feature to the dynamic price multiplier. A linear regression model requires us to generate high-dimensional features to compensate the loss of non-linearity in the model itself. We generate high-dimensional features by integrating features from different data sources, called *basic features*, to form new, high-dimensional *composite features*. In this section, we describe how to extract and integrate such features. Tab. 2 summarizes all basic features and shows some illustrative examples of composite features.

4.1 Basic Features

Basic features are extracted from each of our datasets. Features of the RoD service and taxi dataset are processed from their corresponding data fields, whereas features of other datasets are simply the corresponding data fields.

4.1.1 RoD Service Features. We extract the following features from RoD service dataset: *month*, *hour of day*, *day of week*, *day of month*, *estimated fare*, *isHoliday*, *isWeekend*, and *historical price multipliers*. The *historical price multipliers* include the average price multiplier in the last 1, 2 and 3 hours within a 500-meter radius of the location of the corresponding RoD data entry. Because the price multiplier is regular, to different extents, in different locations as shown in Fig. 2, we expect the historical price multipliers will influence the current price multiplier. The *estimated fare* is an indication of the travelling distance, and when averaged, the average estimated fare in a small region describes the travelling habit of people in this region from a particular perspective. Other features are temporal features, and as we can see from Fig. 2 to 8, these features play an important role in predicting the price multiplier. The location of the ***EstimateFee*** event (i.e., a pair of longitude and latitude values) is not used as a RoD service feature; alternatively, we use the location of the event to extract features from the other 4 datasets. In the following 4 datasets, "around the location" means "within a 500-meter radius of the location".

4.1.2 Taxi Service Features. In §3.2, we mention that we have the taxi GPS trajectory data and also extract the trip information of taxis. From the trip information, we extract 2 features: *up count* and *down count* – the number of passengers getting on (off) taxis around the location (i.e., the pair of longitude and latitude values in a RoD data entry). From the taxi GPS trajectory, we extract 5 features:

- *aveage speed*: the average speed of full taxis (i.e., with passengers on-board) around the location.
- *speed variance*: the variance of speed among full taxis around the location.
- *taxi count*: the number of taxis appearing around the location.
- *full taxi count*: the number of full taxis appearing around the location.
- *full taxi ratio*: the ratio of full taxis to all taxis around the location.

Table 2. Feature engineering: basic features and some selected composite features.

Basic features		
Dataset	**Feature**	**Description**
RoD	*month*	the month the RoD event takes place
	hour of day	the hour of day the event takes place
	day of week	the day of week the event takes place
	day of month	the day of month the event takes place
	estimated fare	the estimated trip fare for the event
	isHoliday	whether the event takes place in a holiday
	isWeekend	whether the event takes place in weekends
	historical price multipliers	the average price multiplier in the last 1, 2, 3 hours
Taxi	*up count*	# of passengers getting on taxis around the location
	down count	# of passengers getting on taxis around the location
	average speed	average speed of full taxis around the location
	speed variance	variance of speed among full taxis around the location
	taxi count	# of taxis appearing around the location
	full taxi count	# of full taxis appearing around the location
	full taxi ratio	the ratio of full taxis to all taxis around the location
	variance of taxi count	variance of taxi count daily
	variance of full taxi count	variance of full taxi count daily
Bus & metro	*bus station count*	# of bus stations around the location
	bus line count	# of bus lines around the location
	metro station count	# of metro stations around the location
	metro line count	# of metro lines around the location
POI	*POI counts*	# of POIs of 14 categories around the location
Weather	*temperature*	the temperature of the city at the time of the event
	wind speed	the wind speed at the time of the event
	humidity	the humidity at the time of the event
	pressure	the atmosphere pressure at the time of the event
	visibility	the visibility at the time of the event
	weather condition	the type of weather at the time of the event
Composite features		
Type	**Datasets**	**Examples of combinations**
Same dataset	RoD+RoD	(*hour of day*, *day of week*), (*hour of day*, *isWeekend*)...
	Taxi+Taxi	(*full taxi count*, *full taxi ratio*), (*average speed*, *up count*)...
	Bus&metro+Bus&metro	(*bus station count*, *metro station count*)...
	...	...
Different dataset	RoD+Taxi	(*day of week*, *average speed*), (*hour of day*, *up count*)...
	RoD+Bus&metro	(*hour of day*, *bus station count*), (*isHoliday*, *bus line count*)...
	RoD+POI	(*day of week*, *POI counts*), (*isWeekend*, *POI counts*)...
	ROD+Weather	(*day of week*, *weather condition*), (*historical price multipliers* ,*temperature*), (*day of week*, *visibility*)...
	Taxi+POI	(*full taxi count*, *POI counts*), (*average speed*, *POI counts*)...
	Taxi+Bus&metro	(*taxi count*, *bus station count*), (*up count*, *bus line count*)...

The *up/down count* features are an indication of passengers' demand for taxis and the popularity of the location. The *average speed* and *speed variance* reflect the traffic condition around the location. The other 3 features describe the availability of taxis as well as the popularity of the location.

For the above 7 features, we calculate each of them based on the taxi GPS entries that fall in the same hour-of-day (called "*hourly taxi features*") and in the same hour-of-day and day-of-week (called "*daily taxi features*"). Additionally, we extract 2 other daily taxi features: *variance of taxi count* and *variance of full taxi count* to characterize the variance of the availability of taxis and of the location's popularity.

The above taxi service features can not only reveal information about the taxi service, but also provide clues to a number of useful facts about the location.

4.1.3 Features of Other Datasets. Features of other 3 datasets are simply the data fields in each dataset, and we don't elaborate on them here. These features are listed in Tab. 2.

4.1.4 Feature Normalization. There are two different kinds of features: numerical (e.g., *average speed* or *bus station count*) and categorical feature (e.g., *day of week* or *hour of day*). A categorical feature may take a value that represents one of the several categories, but the value itself does not matter. For example, the *day of week* feature have 7 categories (corresponding to Monday, Tuesday, ..., Sunday), and we pay attention to which category it falls in, instead of the value it takes. For a categorical value, we apply one-hot extension to transform it into a vector, and the dimension of the vector is the number of possible categories. In this vector, only one dimension takes the value "1" and other dimensions take the value "0". We still take the *day of week* feature as an example: if we use the 7 dimensions to represent Monday to Sunday, a *day of week* variable representing Wednesday can be transformed into a vector $[0, 0, 1, 0, 0, 0, 0]$.

Then we apply normalization on feature variables:

- For categorical feature, there is only one "1" in the vector. So there is nothing to do about normalization.
- For numerical feature, we use N to denote the number of samples in the training set. For one numerical feature x, it has the value $x_1, x_2, ..., x_N$. The normalization transforms each $x_i (1 \leq i \leq N)$ to $x_i^{`}$:
$$x_i^{`} = \frac{x_i - x_{min}}{x_{max} - x_{min}}, \tag{1}$$
where x_{min} and x_{max} are the minimum and maximum value among $x_1, x_2, ..., x_N$.

The goals of normalization are two-fold. Firstly, normalization unifies the meaning and scale of different features, so that we are able to perform further processing on them. Secondly, normalization is necessary for a fast convergence in using gradient descent to train a linear regression model.

4.2 Composite Features

A linear regression model can quantify the contribution of each feature, but it fails to characterize the correlation between different features. As a result, increasing the number of basic features only brings limited improvement to our prediction accuracy. It is thus vital to combine basic features to form composite features that explicitly account for the correlation among basic features. In principle, we can combine as many basic features together, but this may lead to overfitting and prohibitively high dimension for model training. In practice, we only combine two basic features to form composite features.

We can combine any two basic features, and then inspect the corresponding weight of the composite feature in the trained model to see if the combination is necessary. Because of the limited space, we only explain some illustrative examples of composite features in the following, and Tab. 2 gives more examples.

4.2.1 Combining Features from the Same Dataset.

We can combine features from the same dataset to express the correlation between different features. For example, in §3.1 we observe that both *hour of day* and *day of week* influence the price multiplier, so we can combine them as a composite feature. Similar combinations include (*hour of day*, *isWeekend*), (*day of week*, *day of month*), (*full taxi count*, *full taxi ratio*), etc.

4.2.2 Combining Features from Different Datasets.

We can combine features from different datasets, and this is the key to our study on the relationship between different means of transportation. Combining features from different datasets also enables us to study the roles of weather and POI data. For example, we can combine a RoD service feature "*day of week*" with a taxi service feature "*average speed*". Another example involves a RoD service feature and a weather feature: (*historic price multiplier in the last 1 hour*, *temperature*).

4.2.3 Discussions on Composite Features.

We first discuss how to combine the normalized basic features to form composite features. In the following, we discuss the combination of two basic feature variables x_a and x_b to form a composite feature variable x_c:

- Both variables are numerical features: in this case we have $x_c = x_a x_b$.
- x_a is a numerical feature and $\vec{x_b}$ is a categorical feature: in this case, the composite feature is $\vec{x_c} = x_a \vec{x_b}$. Note that both $\vec{x_b}$ and $\vec{x_c}$ are vectors.
- Both variables are categorical features: we use d_a and d_b to represent the dimensions of $\vec{x_a}$ and $\vec{x_b}$, then the resulting variable $\vec{x_c}$ has a dimension of $d_c = d_a d_b$. In particular, if $\vec{x_a} = (x_{a1}, x_{a2}, ..., x_{ad_a})$ and $\vec{x_b} = (x_{b1}, x_{b2}, ..., x_{bd_b})$, then

$$\vec{x_c} = (x_{a1}x_{b1}, x_{a1}x_{b2}, ..., x_{a1}x_{bd_b}, x_{a2}x_{b1}, x_{a2}x_{b2}, ..., x_{a2}x_{bd_b}, ..., x_{ad_a}x_{b1}, x_{ad_a}x_{b2}, ..., x_{ad_a}x_{bd_b}) \quad (2)$$

Using composite features compensates for the loss of non-linearity in a linear regression model by adding product-form terms. We use a very simple example of combining two numerical features to illustrate this. Assuming that we have two basic features x_a and x_b, a simplest linear regression model can be expressed as $y = \omega_a x_a + \omega_b x_b + b$, where y is the predicted value, and ω_a, ω_b, b are the model parameters to be learned. By adding a composite feature the model then becomes $y = \omega_a x_a + \omega_b x_b + \omega_c x_a x_b + b$. This shows how using composite features adds non-linear terms into the linear regression model. Though this example is not rigorous enough, it gives a sense on how composite features work.

5 THE LINEAR REGRESSION MODEL

5.1 The Model

We train a linear regression model on the basic and composite features extracted above to predict dynamic price multipliers. It is true that there are some other regression models in machine learning that are more sophisticated, but a linear model is the best fit in our study because of its interpretability. It is thus possible to determine feature contribution by investigating the weights of corresponding features, so that we can study what features influence the dynamic prices in RoD service the most. The interpretability also makes it possible to quantify the relationship between different means of transportation, by investigating the contribution of those composite features combined from corresponding datasets.

On the contrary, some more sophisticated regression models are not suitable enough to our study due to the lack of interpretability, although some of them can lead to better prediction accuracy. For example, non-linear complex models such as neural network or other deep learning models may be able to achieve a good prediction accuracy with only a small dimension of features, but they are generally hard to interpret. As another example, decision tree is a non-parametric interpretable model, but in practice a decision tree model requires a large number of layers or multiple trees to perform well with high-dimensional features, which diminish its interpretability.

In this section, we present our linear regression model with high-dimensional features. Before discussing the technical details, we first clarify how the prediction model works. Given a passenger request (i.e., an *EstimateFee* event at a particular time and location), we extract the basic and composite features from our multi-source urban data, and build a feature vector containing these features. The feature vector is the input to the linear regression model, and the output of the model is a predicted price multiplier. To predict the price multiplier based on a passenger request – that is, what the price multiplier will be if the passenger wait a while or walk away a little bit – we estimate the features at that time and location, and predict the price multiplier based on the estimated features. How to estimate and extract these features is another story, and is not the focus of this paper. As our prediction is targeted to each passenger request instead of specific time or locations, we don't have the concept of "time window" in our paper. In other words, we are not predicting the price multiplier in the unit of any pre-defined "time slot".

We use y to denote the dynamic price multiplier in a specific *EstimateFee* event, and $x \in \mathbb{R}^m$ the feature vector corresponding to y. m is the dimension of the feature vector, and x contains every feature we introduced earlier in §4. In our case, we build feature vectors of about 4,000 dimensions. With these notation, we can write our raw feature dataset as $D = \{(x_i, y_i) | i = 1, 2, \ldots, N\}$, where (x_i, y_i) represents the feature vector and the corresponding price multiplier of the i-th sample. N is the number of raw data entries in our training dataset: N is a fraction of 14,587,353 (see §3.1). Our linear regression model is trained batch-by-batch, and we also use D to represent the raw feature dataset in each batch.

Note that at this stage, all the feature variables are normalized (as described in 4.1.4) so that each dimension of x_i is within $[0, 1]$.

The target to be learned is a set of parameter vector, ω, of the same dimension of x, and an intercept b. The prediction target (i.e., the predicted price multiplier given a set of features) can be written as $p_i = \omega^T x_i + b$. The simplest form of the objective function to optimize is a squared error loss with $L1$ and $L2$ regularizations:

$$obj_{basic}(\omega) = \sum_{(x_i, y_i) \in D} (y_i - p_i)^2 + \lambda_1 ||\omega||_1 + \lambda_2 ||\omega||_2, \tag{3}$$

where the latter two terms are for $L1$ and $L2$ regularizations, with λ_1 and λ_2 as the trade-off parameter. The $L1$ regularization uses a $L1$-norm penalty term to control the sparsity of the learned parameter ω, so that there are more components of ω becoming zero or close to zero. This is beneficial for the model's interpretability, as it makes fewer features have a strong influence on the predicted dynamic price, while a lot others get a zero weight in the learned model, meaning that these features are irrelevant to the dynamic price. The use of $L2$ regularization, on the other hand, is a common practice in machine learning to avoid over-fitting, so that there will not be a huge gap between the model's performance on the training set and on the test set. It controls the $L2$-norm of ω so that any component of ω should not have an overwhelming influence on the predicted price multiplier.

Similar to [41], we also add a spatio-temporal regularization term to our optimization objective of the linear regression model. The rationale behind this regularization term is that the price multiplier should be smooth, and ***EstimateFee*** events taking place in nearby locations or close in time should have price multipliers close to each other. In a mini-batch training, we simply choose each batch containing events close in time or locations, and use the following regularization term:

$$obj_{spatio-temporal}(\omega) = \sum_{(x_i, y_i) \in D} \phi(D) var(\omega^T x_i + b | x_i \in D). \tag{4}$$

In (4), $var()$ is the variance of the predicted multipliers in the batch. $\phi(D)$ maps the batch of data to a real value, representing the closeness (in features) of all the data $(x_i, y_i) \in D$. $\phi(D)$ is defined as:

$$\phi(D) = \prod_{(x_i,y_i)\in D} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(x_i - \overline{x})^T(x_i - \overline{x})}{2\sigma}), \tag{5}$$

where $\overline{x}$ is the mean of all $x_i \in D$.

Combining (3) and (4) yields our objective function:

$$obj(\omega) = \sum_{(x_i,y_i)\in D} (y_i - p_i)^2 + \lambda_1||\omega||_1 + \lambda_2||\omega||_2 + \gamma\phi(D)var(\omega^T x_i | x_i \in D), \tag{6}$$

where γ is the trade-off parameter controlling the influence of spatio-temporal regularization.

We then use the stochastic gradient descent (SGD) algorithm to minimize the objective function based on the training data, and obtain a linear regression model with parameters ω and b.

5.2 Implementation Consideration

Applying a linear regression model on high-dimensional features may bring some scale-related problems, as it requires high computation power and storage capability. The implementation of our current linear regression model is not faced with these challenges because:

- We only combine two basic features to form composite features, and this makes the total dimensions of features less than 4,000. This is not a very high dimension and we can directly apply our linear regression model on the features.
- At this dimension it takes about 30 minutes to train the model based on the complete training set on an ordinary Intel Core-i7 personal computer. Though this seems to be a long time, in practice we never train the model based on the whole dataset at once; instead, we use mini-batch stochastic gradient descent to train the model. In other words, we gradually take one batch of data after another and incrementally train the whole model. A batch is much smaller than the whole training set, and it takes less than 1 minute to train the model based on a batch.

Challenges may arise if we keep increasing the dimension of features by, for example, combining three or more basic features to form composite features and growing the dimension to a much higher magnitude. The memory may be overwhelmed in loading training data; the training time will also increase exponentially, and even training based on a batch of data cannot meet the time requirement of online model training in practice. Taking smaller batches for training is a way to solve these problems, but a fast convergence cannot be guaranteed when the batch size becomes too small. Another realistic solution is to use distributed training framework with parameter servers [13, 26, 30]. Put it simple, we can use multiple machines to perform the mini-batch SGD training process in a distributed fashion, and also use multiple machines to store and update the learned parameters. The nature of SGD (i.e., the training can be performed batch by batch) makes itself a perfect fit for distributed model training.

We don't come across these challenges because we only combine two basic features to form composite features instead of using three or more. There are multiple considerations:

- We try to combine three basic features to form composite features. Our evaluation shows that while the training time is more than doubled, the prediction accuracy does not improve significantly.
- Composite features combined from three or more basic features have a reduced interpretability. They cannot provide more information in quantifying the feature contribution and the relationship between transportation services.

6 EVALUATION AND RESULTS

In this section, we present the evaluation of our trained linear regression model, and discuss the features that contribute the most to the dynamic price multiplier.

6.1 Evaluation Setup

Our raw feature dataset contains 14,587,353 entries, and each entry has about 4,000 dimensions. We randomly choose 70% entries as the training set, and use the remaining 30% as the testing set. We do this random selection for 10 times and obtain 10 linear regression models. The average metric of these models is used for evaluation.

6.1.1 Evaluation Metric. 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, p_i, are equal to the ground-truth y_i. In predicting price multipliers, on the other hand, we don't care that much about the accuracy measure. In some cases, even though there are a slight difference between the predicted multiplier and the ground truth, it is not a problem for passengers. For example, a passenger getting a price multiplier 1.3 only wants to know if it would be possible to get a lower multiplier nearby or within a short distance, but doesn't care that much whether the lower multiplier is 1.1 or 1.2.

Instead, we use the symmetric mean absolute percentage error (sMAPE) [45], a metric based on the relative difference (or error):

$$sMAPE = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \frac{|y_i - p_i|}{y_i + p_i}. \tag{7}$$

In (7), N_{test} is the size of the testing set. A higher sMAPE means lower prediction accuracy. There are multiple considerations on choosing sMAPE as our evaluation metric:

- The sMAPE metric has the advantage of being scale-independent and easily interpretable, as it is in a percentage form. Furthermore, as the price multiplier is always greater than 0, we can avoid getting undefined values in using sMAPE (i.e., the sMAPE metric may take an undefined value when the predicted or the actual quantity is 0)
- More importantly, because of the special properties of price multipliers, other metrics such as MAE (*mean absolute error*), MSE (*mean squared error*) or RMSE (*root-mean-square error*) can be directly represented from the prediction accuracy (i.e., in what percentage we have a absolute difference being 0, 0.1, 0.2,...,0.6 between the predicted price multiplier and the ground truth), while sMAPE cannot. This is because the price multiplier only takes discrete values such as 1.0, 1.1, ..., 1.6 (and we round the predicted price multiplier to these values), and so the difference between the predicted price multiplier and the ground truth also takes discrete values. As a result, metrics such as MAE/MSE/RMSE can be calculated from the prediction accuracy directly.
- Using sMAPE to measure the relative prediction error is a common practice in evaluating forecast accuracy on, for example, human mobility pattern, taxi demand prediction and so on [40, 41, 50, 51]. Moreover, the baseline predictors we use in our evaluation (see §6.1.2) also uses the sMAPE metric. To compare our results in this paper with the baseline predictors, we also use sMAPE so that it can give a sense as to how our prediction model performs.

6.1.2 Baselines. We use two simple predictors in [18] as baselines. These two predictors try to predict the hourly average price multiplier in specific city functional areas such as business, residential and transportation areas, based on the history of hourly average price multipliers in a one-month time range. These predictors are applied in a coarser granularity: first, they only try to predict multipliers in specific

Table 3. sMAPE of baseline predictors.

Predictor	Business	Residential	Transportation
Markov-chain	0.0548	0.0468	0.0366
Neural network	0.0448	0.0457	0.0513

functional areas; second, the goal of prediction is the hourly average price multiplier. By comparison, the linear regression model in our paper is used to predict the *exact* price multiplier given a set of features in each *individual* passenger request, in *any location* of the city. But the baseline predictors can still give us a sense of the sMAPE metric.

The first baseline predictor is a Markov-chain predictor, predicting the price multiplier by training a 3-order Markov predictor. Another baseline predictor is a neural network predictor, predicting the price multiplier by training a two-layer neural network, involving both the temporal and weather features. Tab. 3 shows the sMAPE of these baseline predictors in selected business, residential and transportation areas. Similar to Fig. 2, the criteria of selecting these typical functional areas could be found in [18, 19] and is not discussed here.

6.2 Overall Results

We train a linear regression model based on the training set and calculate the sMAPE on the testing set. We train for 10 times, and the average sMAPE is 0.0431. This sMAPE could be considered a much better result than our baseline predictors for the following two reasons:

- The sMAPE of our linear regression model is already lower than baseline predictors, except in the case of using Markov-chain predictor in transportation area. The specific city functional areas chosen in [18] exhibit, to some extent, certain regularity in either the demand or the price multiplier, but in our paper the linear regression model is for any location in the city, where the price multipliers are more random.
- The sMAPE of our linear regression model is averaged among every single ***EstimateFee*** event, instead of from the predicted hourly average price multiplier.

We also investigate the absolute value of the difference between the predicted price multiplier and the ground truth based on the testing set. The percentage of predicting exactly the ground truth multiplier is 40.28%, and the percentage of having a difference of 0.1 is 40.89%. The percentages of having a difference of 0.2 to 0.6 are 15.32%, 2.25%, 0.86%, 0.38% and 0.02%, respectively. These percentages indicate that our linear regression model can have very good prediction (i.e., having a difference less than or equal to 0.1) in about 81.17% cases.

The need to involve composite features to improve the expressiveness of the linear regression model can also be seen from the sMAPE. We train a linear regression model with only basic features, and its sMAPE turns out to be 0.0672, 55.92% higher than the sMAPE with composite features. Then we train a model with basic features and composite features from the same dataset (see §4.2.1), the sMAPE is 0.0576 and this shows that using multi-source urban data also significantly improves the prediction accuracy.

Besides, we also test the performance improvement of using $L2$ and spatio-temporal regularization in model training. For $L2$ regularization, its goal is to avoid over-fitting, and we compare the prediction accuracy on the training and test set with and without $L2$ regularization. Tab. 5 shows the sMAPE and prediction accuracy (within 0.1-difference in price multiplier), and these figures show that using $L2$ regularization indeed reduces the difference of prediction accuracy between the training and test set. For spatio-temporal regularization, we also compare the prediction accuracy with and without

Table 4. The top-20 features/dimensions ranked by coefficients.

Rank	Feature (:dimension)	Coef	Datasets
1	(*historical price multipliers: 1 hour*)	0.400409	RoD
2	(*historical price multipliers: 1 hour, pressure*)	0.391540	RoD+Weather
3	(*isHoliday, weather condition: light rain*)	-0.132702	RoD+Weather
4	(*historical price multipliers: 1 hour, temperature*)	0.131557	RoD+Weather
5	(*day of week: Friday, weather condition: light rain*)	0.129897	RoD+Weather
6	(*historical price multipliers: 1 hour, humidity*)	0.126101	RoD+Weather
7	(*hour of day: 7pm, weather condition: sprinkles*)	0.124791	RoD+Weather
8	(*hour of day: 8am, day of week: Monday*)	0.124486	RoD
9	(*historical price multipliers: 2 hour, taxi count*)	-0.119318	RoD+Taxi
10	(*hour of day: 7am, day of week: Monday*)	0.115868	RoD
11	(*historical price multipliers: 2 hour, full taxi count*)	-0.112601	RoD+Taxi
12	(*historical price multipliers: 1 hour, month: Oct.*)	0.108534	RoD
13	(*hour of day: 5pm, full taxi ratio*)	0.107540	RoD+Taxi
14	(*historical price multipliers: 3 hour*)	0.106606	RoD
15	(*historical price multipliers: 3 hour, pressure*)	0.103188	RoD+Weather
16	(*historical price multipliers: 1 hour, visibility*)	0.100780	RoD+Weather
17	(*historical price multipliers: 1 hour, day of week: Tuesday*)	0.099416	RoD
18	(*historical price multipliers: 1 hour, isHoliday*)	-0.099234	RoD
19	(*historical price multipliers: 1 hour, full taxi ratio*)	0.098980	RoD+Taxi
20	(*hour of day: 12pm, weather condition: sprinkles*)	0.098235	RoD+Weather

Table 5. Prediction accuracy on the training and test set with/without $L2$ regularization.

	with $L2$ regularization	without $L2$ regularization
on training set	0.0425 and 83.32%	0.0418 and 84.75%
on test set	0.0431 and 81.17%	0.0472 and 76.25%

the regularization term. As shown above, the sMAPE and prediction accuracy are 0.0431 and 81.17% with spatio-temporal regularization. The corresponding figures are 0.0464 and 77.87% without this regularization term. The comparison shows that using spatio-temporal regularization can lead to better performance.

Additionally, we also train a neural network model with all the basic features shown in Tab. 2 extracted from the multi-source urban datasets. As mentioned in §5, a non-linear model does not require composite features to characterize the correlation between features, we thus only involve basic features that account for about 130 dimensions. Our neural network model has a four-layer structure, and evaluation results show that the sMAPE is 0.0403 and that the model has a good prediction in 85.68% cases. These results meet our expectation: compared with our linear regression model, a neural network model can provide a slightly higher prediction accuracy with a small dimension of features, but it is not interpretable, takes longer time for training, and is not flexible enough when adding new features.

The overall results indicate that our linear regression model can achieve a much better performance than baseline predictors including a neural network predictor, and this can be explained by the use of composite features and of multi-source urban data. In the next subsection we will unveil the contribution of different features quantitatively.

6.3 Feature Contribution

Following the feature engineering in §4, we include more than 370 features in our linear regression model, and the dimension of features reaches about 4,000. Each dimension (of features) has a corresponding weight in our trained linear regression model, and the absolute value of the weight quantifies the contribution of this dimension/feature to the predicted dynamic price multiplier. In Tab. 4 we show the top-20 features and dimensions, according to the absolute value of their respective coefficients. We also highlight the source of dataset(s) of these features.

Concerning the source of dataset(s) of these features. The RoD data is undoubtedly the most influential dataset to produce features: each of the top-20 features are from RoD data. Among these 20 features, 9 of them are related to the weather data, and 4 of them are related to the taxi data. None of these features is relevant to our POI data or public transportation distribution data. So, from a very high-level perspective, the dynamic price multiplier is mostly influenced by the RoD data itself, the weather data and the taxi data.

The presence of the weather and taxi data as the most influential datasets can be understood intuitively. The weather condition influences passengers' demand: in bad weather more people are inclined to find a taxi or car in a shorter waiting time at the expense of a higher price. The taxi service, as a similar competitor to the RoD service, also plays an important role: when there are more available taxis around for hailing, fewer people open the RoD app and search for available cars.

Concerning individual features. The historical price multipliers (in the last 1, 2 or 3 hours) are the most important features, appearing in 13 out of top-20 features. These features are either the historical price multipliers themselves, or composite features with RoD features (e.g., *month*, *day of week*, *isHoliday*), weather features (e.g., *pressure*, *temperature*, *humidity* and *visibility*) or taxi features (e.g.,*taxi count*, *full taxi count* and *full taxi ratio*). Regarding the coefficients, the *historical price multiplier in the last 1 hour* has a coefficient 0.400409, more than twice other coefficients. The strong correlation between historical and current price multipliers also indicate that the variation of dynamic prices follows certain pre-defined, rational rules (though unknown), making it possible and meaningful to predict price multipliers.

The influence of weather features is also clear when investigating individual features. For example, when there is higher temperature, the price multiplier rises. This is, possibly, due to the fact that more passengers tend to request for services when it is hotter. Similarly, when there is rain (e.g., light rain or sprinkles), the price multiplier is higher, possibly because there are higher demand. Most importantly, the influence is quantified, and it is possible for one to compare the influence of different features directly.

We also observe that there is indeed competition between taxi and RoD service, but the competition is not as fierce as it is expected. A taxi feature itself does not have a strong influence on the price multiplier: it only affects the price multiplier strongly when associated with RoD features. Qualitatively speaking, when there are fewer taxis (i.e., lower *taxi count* and *full taxi count*) and fewer available taxis (i.e., higher *full taxi ratio*) around, the price multiplier of RoD service rises, and vice versa. The RoD features associated with taxi features are the *historical price multipliers* and *hour of day*. Specifically, the feature ranked #13 indicates that only during the evening rush hour (i.e., 5pm to 6pm) the competition between taxi and RoD service is apparent. During other hours of the day, the competition is not strong, and the reasons may be that (a) passengers' demand is so high that taxis and RoD cars together are still not enough to meet the demand during, for example, the time after 6pm in evening rush hour and (b) when the demand is not high, passengers may choose between taxis and RoD services according to their own preference instead of according to the supply of taxis or RoD cars around.

Other top features not explicitly explained above are self-explanatory: they are effective in increasing the price multipliers by increasing passengers' demand (e.g., bad weather, rush hours, busy days) and/or decreasing the supply of cars around (e.g., bad weather, holidays).

6.4 Discussions

There are two interesting facts in Tab. 4: the absence of features from POI and public transportation distribution data. In this subsection, we provide some discussions.

The relationship between RoD service and public transportation. The distribution of public transportation such as bus & metro does not have a strong influence on the dynamic prices in RoD service. In fact, for those features, either basic or composite, associated with bus & metro features, the largest absolute value of coefficient is about 0.01, and most coefficients are even smaller than 0.0001 – far smaller than the coefficients of top features listed in Tab. 4.

The reason behind this, as we consider, is that public transportation such as bus & metro have different customer base with RoD service. For a potential passenger of RoD service, when he fails to get a car or when he considers the price multiplier is too high, he may turn to taxis (as validated by the competition between taxis and RoD service in Tab 4) or choose to wait for a while, instead of seeking for buses around. Similarly, for people who usually take buses, they seldom resort to taxis or RoD services unless faced with extreme conditions such as in a hurry to go to a meeting or to the airport.

The influence of POI features. The POI features we use in our study is the *POI counts* – the number of POIs of 14 different categories around the location. Intuitively, these POI features should have some influence on the predicted price multipliers, as the dynamic price and its variation is definitely related to the location as shown from Fig. 2 to 6. Interestingly, we don't find any features relevant to POI features in Tab. 4. As a matter of fact, for those features relevant to POI features, the largest absolute value of coefficient is as small as 0.001, and there are indeed a lot of zero coefficients.

We hypothesize that the reasons are two-fold. Firstly, the location information is already partly revealed by taxi features (both basic and composite features), though implicitly. For example, the number of taxis/full taxis visiting a location, the average speed of taxis driving around a location and the number of passengers getting on/off taxis around a location all help to describe a picture about the supply, demand and traffic condition that can characterize the location.

Another reason is from the inability of POI counts to reveal location information. We point out in §3.4 that using the category of the nearest POI to characterize the location is not enough, and now our results suggest that using POI counts is also not a good idea. An example around the airport terminal can clearly illustrate this. A passenger is standing in the airport terminal and requests for a ride. Clearly the "transportation facility" property is the reason why s/he is here. The POI counts, on the other hand, may not suggest this. The number of "transportation facility" POIs may be only one – the terminal; but there may be a number of shops, restaurants or hotels around, and the number may far exceed that of "transportation facility". In other words, the *POI counts* features emphasize the number of POIs, but in some cases we also need to differentiate POIs by their "importance".

We verify our second hypothesis by investigating the check-in data of a location-based service (similar to Foursquare) in China. The data includes the check-in record in Beijing during 2012. If a place receives more check-ins, it is safe, to some extent, to claim that this place is important or at least, more frequently visited. For the location of an ***EstimateFee*** event, we find out the place around the location that is most frequently visited, and use this place's number of check-ins and category as our POI-related features. We add these features and train the linear regression model accordingly. The resulting sMAPE is 0.0429 and is very close to our earlier results in §6.2; for those features related to the newly-added POI check-in features, the largest absolute value of coefficient is 0.02 – 20 times larger than that of POI counts features.

Another way to verify our second hypothesis is to use the TF-IDF statistics to characterize the "importance" of POIs from a different perspective. The *POI counts* feature uses the number of POIs of different categories – for the i-th POI category, we use p_i to denote the number of POIs around. For the whole city, we use N to denote the total number of POIs, and use N_i to denote the total number of POIs of the i-th category. We can then calculate the TF-IDF of the i-th POI category around the location one sends out an *EstimateFee* request:

$$p_{tfidf,i} = p_i * log(\frac{N}{1+N_i}). \tag{8}$$

Instead of using p_i as the POI features (i.e., the *POI counts* feature), now we use $p_{tfidf,i}$ as the POI features. In this way, different categories are not weighted equally - the more common a category of POI is, the more its weight is diminished. There are certainly other more complex definitions of the TF-IDF value, but here we use this simple one for the purpose of evaluating the effectiveness of using TF-IDF. When replacing the *POI counts* features with the TF-IDF values, our evaluation shows that the sMAPE is 0.0430 (very close to the original one) but again features relevant to the TF-IDF features have the largest weight about 17.5 times than features related to *POI counts* have.

These results suggest that the *POI counts* features indeed fail to express the POI's importance, and that the check-in data or the TF-IDF statistics of *POI counts* can rectify this problem to some extent. Both methods make the corresponding features have higher influence on the dynamic prices – in other words, they describe the location information more accurately. But they are two directions to characterize the "importance" of POIs:

- Using check-in data: POIs that are more frequently-visited are more important;
- Using TF-IDF statistics: The categories of POIs that appear more common on the map are less important.

For the check-in data, we don't include them in our earlier results in §6.2 because it dates back to as early as 2012 and POIs in Beijing may have changed a lot in 4 years. In other words, the check-in data is meaningful, but may not be accurate enough to be included in our overall results. We also don't include the TF-IDF statistics in our main results, as it is only one direction to characterize the "importance" of POIs, and the improvement is still not enough. We plan to find out another way to combine these two directions together and generate a much more accurate description of location information, and this is left as the future work.

7 CONCLUSION

We focus on the prediction of dynamic prices in emerging RoD services such as Uber and Didi. Predicting the dynamic prices can help passengers to obtain more information and make decisions (i.e., whether to take a ride) at ease. We use a linear regression model with high-dimensional features extracted from multi-source urban data to predict the price multiplier. On one hand, the use of multi-source data helps to make the prediction more accurate; on the other hand, we are able to quantify the relationship between RoD service and multiple means of transportation.

We train a linear regression model with more than 370 features, and the dimension of features reaches about 4,000. The model can achieve a very good prediction in about 81.17% cases, with a sMAPE of 0.0431, much better than our baseline predictors. The sMAPE metric also shows that combining features from multi-source urban datasets is beneficial. We also quantify the contribution of features with the help of the linear regression model. Results show that the historical price multipliers of the last several hours are the most important features that contribute to the price multiplier. Weather condition and the availability of taxis around are also key features, accounting for a significant fraction in top features.

Regarding the relationship between different means of transportation, our results show that there is indeed competition between taxis and RoD services, but it is not as fierce as expected. Only in very special circumstances (e.g., during evening rush hour) the availability of taxis plays an important role in the dynamic prices of RoD services. Contrarily, the influence of public transportation such as bus & metro is negligible – showing that RoD service may have different customer base with public transportation and there is little competition between them.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Project of China (No. 2017YFB1002000), the National Science Foundation of China (No. 61602067, 61472212, 61572059, 61202426, 71531001), the Fundamental Research Funds for the Central Universities (No. 2018CDQYJSJ0024) and Frontier Interdisciplinary Research Fund for the Central Universities (No. 106112017CDJQJ188828). Suiming Guo and Chao Chen contributed equally on this work.

REFERENCES

[1] Aimee Picchi. 2016. Uber vs. Taxi: Which Is Cheaper? Retrieved Jan 25, 2018 from http://bit.ly/2DMgrMc

[2] AMap. 2017. API of AMap Service. http://bit.ly/2n8YRbZ

[3] Shunsuke Aoki, Kaoru Sezaki, Nicholas Jing Yuan, and Xing Xie. 2017. An Early Event Detection Technique with Bus GPS Data. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '17)*. ACM, 49:1–49:4.

[4] Omar Besbes and Assaf Zeevi. 2009. Dynamic Pricing Without Knowing the Demand Function: Risk Bounds and Near-Optimal Algorithms. *Operations Research* 57, 6 (2009), 1407–1420.

[5] Yong Cao, Thomas S. Gruca, and Bruce R. Klemz. 2003. Internet Pricing, Price Satisfaction, and Customer Satisfaction. *International Journal of Electronic Commerce* 8, 2 (2003), 31–50.

[6] Chao Chen, Shuhai Jiao, Shu Zhang, Weichen Liu, Liang Feng, and Yasha Wang. 2018. TripImputor: Real-time Imputing Taxi Trip Purpose Leveraging Multi-sourced Urban Data. *IEEE Transactions on Intelligent Transportation Systems* PP, 99 (2018), 1–13.

[7] 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.

[8] Chao Chen, Daqing Zhang, Xiaojuan Ma, Bin Guo, Leye Wang, Yasha Wang, and Edwin Sha. 2017. CrowdDeliver: Planning City-wide Package Delivery Paths Leveraging the Crowds of Taxis. *IEEE Transactions on Intelligent Transportation Systems* 18, 6 (2017), 1478–1496.

[9] Huihui Chen, Bin Guo, Zhiwen Yu, Liming Chen, and Xiaojun Ma. 2017. A Generic Framework for Constraint-Driven Data Selection in Mobile Crowd Photographing. *IEEE Internet of Things Journal* 4, 1 (2017), 284–296.

[10] 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.

[11] 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.

[12] Peter Cohen, Robert Hahn, Jonathan Hall, Steven Levitt, and Robert Metcalfe. 2016. Using Big Data to Estimate Consumer Surplus: The Case of Uber. Retrieved Jan 25, 2018 from http://bit.ly/2pqXiWo

[13] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc' aurelio Ranzato, Andrew Senior Senior, Paul Tucker, Ke Yang, Quoc V. Le, and Andrew Y. Ng. 2012. Large Scale Distributed Deep Networks. In *Advances in Neural Information Processing Systems 25*. 1223–1231.

[14] Tobias Franke, Paul Lukowicz, and Ulf Blanke. 2015. Smart crowds in smart cities: real life, city scale deployments of a smartphone based participatory crowd management platform. *Journal of Internet Services and Applications* 6, 1 (2015), 27.

[15] Jerome H. Friedman. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics* 29, 5 (2001), 1189–1232. http://www.jstor.org/stable/2699986

[16] Jon Froehlich, Joachim Neumann, and Nuria Oliver. 2009. Sensing and Predicting the Pulse of the City Through Shared Bicycling. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence (IJCAI '09)*. Morgan Kaufmann Publishers Inc., 1420–1426.

[17] Bin Guo, Zhu Wang, Zhiwen Yu, Yu Wang, Neil Y. Yen, Runhe Huang, and Xingshe Zhou. 2015. Mobile Crowd Sensing and Computing: The Review of an Emerging Human-Powered Sensing Paradigm. *ACM Comput. Surv.* 48, 1, Article 7 (2015), 31 pages.

[18] 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 the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous '17)*. ACM, 1–10.

[19] Suiming Guo, Chao Chen, Yaxiao Liu, Ke Xu, and Dah Ming Chiu. 2018. Modelling Passengers' Reaction to Dynamic Prices in Ride-on-demand Services: A Search for the Best Fare. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4 (2018), 136:1–136:23.

[20] 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.

[21] 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.

[22] Jonathan Hall, Cory Kendrick, and Chris Nosko. 2015. The effects of Uber's surge pricing: a case study. Retrieved Jan 25, 2018 from http://bit.ly/2kayk9O

[23] Wen He, Kai Hwang, and Deyi Li. 2014. Intelligent Carpool Routing for Urban Ridesharing by Mining GPS Trajectories. *IEEE Transactions on Intelligent Transportation Systems* 15, 5 (2014), 2286–2296.

[24] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, and Joaquin Quiñonero Candela. 2014. Practical Lessons from Predicting Clicks on Ads at Facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising (ADKDD'14)*. ACM, Article 5, 5:1–5:9 pages.

[25] G. E. Hinton and R. R. Salakhutdinov. 2006. Reducing the Dimensionality of Data with Neural Networks. *Science* 313, 5786 (2006), 504–507.

[26] Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B. Gibbons, Garth A Gibson, Greg Ganger, and Eric P Xing. 2013. More Effective Distributed ML via a Stale Synchronous Parallel Parameter Server. In *Advances in Neural Information Processing Systems 26*. 1223–1231.

[27] Jacob Davidson. 2014. Uber Has Pretty Much Destroyed Regular Taxis in San Francisco. Retrieved Jan 25, 2018 from http://ti.me/1vegHWv

[28] Michael L. Kasavana and A. J. Singh. 2001. Online Auctions. *Journal of Hospitality & Leisure Marketing* 9, 3-4 (2001), 127–140.

[29] 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.

[30] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling Distributed Machine Learning with the Parameter Server. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI '14)*. 583–598.

[31] Xiaolong Li, Gang Pan, Zhaohui Wu, et al. 2012. Prediction of urban human mobility using large-scale taxi traces and its applications. *Frontiers of Computer Science* 6, 1 (2012), 111–121.

[32] Lloyd Alter. 2017. Is Uber Killing Transit? Retrieved Jan 25, 2018 from http://bit.ly/2DMf1S3

[33] Shuo Ma, Yu Zheng, and Ouri Wolfson. 2013. T-share: A large-scale dynamic taxi ridesharing service. In *2013 IEEE 29th International Conference on Data Engineering (ICDE '13)*. 410–421.

[34] R Preston McAfee and Vera Te Velde. 2006. Dynamic pricing in the airline industry. *forthcoming in Handbook on Economics and Information Systems, Ed: TJ Hendershott, Elsevier* (2006). http://bit.ly/2ChavdL

[35] H. Brendan McMahan, Gary Holt, D. Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, Sharat Chikkerur, Dan Liu, Martin Wattenberg, Arnar Mar Hrafnkelsson, Tom Boulos, and Jeremy Kubica. 2013. Ad Click Prediction: A View from the Trenches. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '13)*. ACM, 1222–1230.

[36] Pedro M.d'Orey, Ricardo Fernandes, and Michel Ferreira. 2012. Empirical Evaluation of a Dynamic and Distributed Taxi-sharing System. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. 140–146.

[37] Anastasios Noulas, Salvatore Scellato, Renaud Lambiotte, Massimiliano Pontil, and Cecilia Mascolo. 2012. A Tale of Many Cities: Universal Patterns in Human Urban Mobility. *PLOS ONE* 7, 5 (2012), 1–10.

[38] Salvatore Scellato, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Andrew T. Campbell. 2011. NextPlace: A Spatio-temporal Prediction Framework for Pervasive Systems. In *Proceedings of the 9th International Conference on Pervasive Computing (Pervasive '11)*. Springer-Verlag, 152–169.

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

[40] Xuan Song, Hiroshi Kanasugi, and Ryosuke Shibasaki. 2016. DeepTransport: Prediction and Simulation of Human Mobility and Transportation Mode at a Citywide Level. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI '16)*. 2618–2624.

[41] Yongxin Tong, Yuqiang Chen, Zimu Zhou, Lei Chen, Jie Wang, Qiang Yang, and Jieping Ye. 2017. The Simpler The Better: A Unified Approach to Predicting Original Taxi Demands on Large-Scale Online Platforms. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17)*. ACM, 1653–1662.

[42] Jingyuan Wang, Fei Gao, Peng Cui, Chao Li, and Zhang Xiong. 2014. Discovering urban spatio-temporal structure from time-evolving traffic networks. In *Proceedings of the 16th Asia-Pacific Web Conference*. Springer International Publishing, 93–104.

[43] Jingyuan Wang, Qian Gu, Junjie Wu, Guannan Liu, and Zhang Xiong. 2016. Traffic Speed Prediction and Congestion Source Exploration: A Deep Learning Method. In *Proceedings of the 16th IEEE International Conference on Data Mining (ICDM)*. IEEE, 499–508.

[44] Leye Wang, Daqing Zhang, Yasha Wang, Chao Chen, Xiao Han, and Abdallah M'hamed. 2016. Sparse mobile crowdsensing: challenges and opportunities. *IEEE Communications Magazine* 54, 7 (2016), 161–167.

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

[46] Haoyi Xiong, Daqing Zhang, Guanling Chen, Leye Wang, Vincent Gauthier, and Laura E. Barnes. 2016. iCrowd: Near-Optimal Task Allocation for Piggyback Crowdsensing. *IEEE Transactions on Mobile Computing* 15, 8 (2016), 2010–2022.

[47] Chuang-Wen You, Nicholas D. Lane, Fanglin Chen, Rui Wang, Zhenyu Chen, Thomas J. Bao, Martha Montes-de Oca, Yuting Cheng, Mu Lin, Lorenzo Torresani, and Andrew T. Campbell. 2013. CarSafe App: Alerting Drowsy and Distracted Drivers Using Dual Cameras on Smartphones. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '13)*. ACM, 13–26.

[48] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. 2010. T-drive: Driving Directions Based on Taxi Trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '10)*. ACM, 99–108.

[49] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI '17)*. AAAI, 1655–1661.

[50] 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 *Proceedings of the 2016 IEEE International Conference on Big Data (Big Data '16)*. 833–842.

[51] Kai Zhao, Sasu Tarkoma, Siyuan Liu, and Huy Vo. 2016. Urban Human Mobility Data Mining: An Overview. In *Proceedings of the 2016 IEEE International Conference on Big Data (Big Data '16)*. 1911–1920.

[52] Yu Zheng, Yanchi Liu, Jing Yuan, and Xing Xie. 2011. Urban computing with taxicabs. In *Proceedings of Ubicomp '11*. 89–98.

Received February 2018; revised July 2018; accepted September 2018