The Simpler The Better: A Unified Approach to Predicting Original Taxi Demands based on Large-Scale Online Platforms

Yongxin Tong[1], Yuqiang Chen[2], Zimu Zhou[3], Lei Chen[4],
Jie Wang[5], Qiang Yang[2,4], Jieping Ye[5], Weifeng Lv[1]
[1] SKLSDE Lab, Beihang University, Beijing, China, [2] 4Paradigm Inc., Beijing, China, [3] ETH Zurich, Zurich, Switzerland,
[4] The Hong Kong University of Science and Technology, Hong Kong SAR, China, [5] Didi Research Institute, Beijing, China
[1]{yxtong, lwf}@buaa.edu.cn, [2] chenyuqiang@4paradigm.com, [3]zimu.zhou@tik.ee.ethz.ch, [4]{leichen,qyang}@cse.ust.hk
[5]{wangjiejacob,yejieping}@didichuxing.com

ABSTRACT

Taxi-calling apps are gaining increasing popularity for their efficiency in dispatching idle taxis to passengers in need. To precisely balance the supply and the demand of taxis, online taxicab platforms need to predict the Unit Original Taxi Demand (UOTD), which refers to the number of taxi-calling requirements submitted per unit time (*e.g.*, every hour) and per unit region (*e.g.*, each POI). Predicting UOTD is non-trivial for large-scale industrial online taxicab platforms because both *accuracy* and *flexibility* are essential. Complex non-linear models such as GBRT and deep learning are generally accurate, yet require labor-intensive model redesign after scenario changes (*e.g.*, extra constraints due to new regulations). To accurately predict UOTD while remaining flexible to scenario changes, we propose LinUOTD, a unified linear regression model with more than 200 million dimensions of features. The simple model structure eliminates the need of repeated model redesign, while the high-dimensional features contribute to accurate UOTD prediction. We further design a series of optimization techniques for efficient model training and updating. Evaluations on two large-scale datasets from an industrial online taxicab platform verify that LinUOTD outperforms popular non-linear models in accuracy. We envision our experiences to adopt simple linear models with high-dimensional features in UOTD prediction as a pilot study and can shed insights upon other industrial large-scale spatio-temporal prediction problems.

KEYWORDS

Unit Original Taxi Demands; Prediction; Feature Engineering

1 INTRODUCTION

Large-scale online taxicab platforms such as Didi Chuxing [1], Uber [3] and Grab [2] are becoming increasingly popular. For instance, on Didi Chuxing, millions of taxi-calling transactions are made in Beijing alone per day. To efficiently dispatch taxis in metropolises and assign them to passengers [18, 19], it is important to predict fine-grained taxi demands and allocate taxis in advance. In a large-scale online taxicab industry, a representative indicator of taxi demand is the *Unit Original Taxi Demand* (*UOTD*), which refers to the number of taxi-calling orders submitted to the online taxicab platform per unit time (*e.g.*, every hour) and per unit region (*e.g.*, each POI). UOTD is different from the number of *pick-ups* (*PU*). The number of PU per unit time and region is a subset of UOTD because the former ignores potential passengers who eventually give up taking taxis. In contrast, UOTD reflects the complete original passenger demands for a given time and space.

Information of UOTD benefits online taxicab platforms in triple ways. (i) *Expanding potential market.* By comparing historical UOTD with the corresponding number of PU, the platforms can discover times and regions with strong taxi-calling motivation yet few final taxi rides. (ii) *Assessing incentive mechanisms.* UOTD reflects the willingness of users to travel by taxi after adopting new discount strategies and dynamic pricing. (iii) *Guiding taxi dispatching.* Predicting UOTD facilitates online taxicab platforms to allocate roaming taxis to passengers in advance. Hence predicting UOTD is a foundational issue in large-scale online taxicab industries.

Despite extensive research efforts on taxi demand prediction [14, 15, 28], none of them are applicable in predicting UOTD. These works focus on predicting the number of PU. They usually predict the number of PU based on the correlation between PU and taxi trajectories. However, taxi trajectories are not always associated with UOTD (*e.g.*, original taxi demands that are cancelled or without successful passenger pick-ups), making it impossible to extend works on PU prediction to UOTD prediction.

It is also non-trivial to tailor generic research on spatio-temporal prediction for UOTD prediction in the taxicab industry. Due to their inherent complexity, real-world prediction problems are mainly solved by high VC-dimension models [20], which consists of two paradigms: *(i)* complicated (non-linear) models with a small number of features [7, 9] and *(ii)* simple (linear) models with massive sets of features [8, 13]. The former paradigm is preferred in most spatio-temporal prediction studies, but can be cumbersome for large-scale online taxicab industries. Imagine the following example. Andy, an AI engineer of an online taxicab platform, needs to add new spatio-temporal features to a deep learning model to meet a new business strategy. He has to design extra spatio-temporal convolution neurons to reflect periodical and locally smoothing characteristics. In the fast-developing online taxicab industry, application and key factor changes due to new regulations or business strategies are common and frequent. Therefore Andy needs to repeat the labor-intensive model redesign process almost continuously. To mitigate such heavy burden, we propose to transfer *model redesign* to *feature*

KDD'17, August 13–17, 2017, Halifax, NS, Canada.

DOI: http://dx.doi.org/10.1145/3097983.3098018

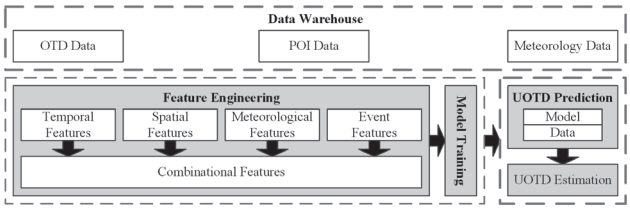


Figure 1: An overview of the framework.

redesign. Specifically, we leverage the latter paradigm, *i.e.*, a linear model with massive features, to ease integration of new information with a unified framework.

A natural question arises *whether a unified simple linear model is able to predict UOTD accurately*. We tackle this problem by integrating high-dimensional features from heterogenous datasets. Specifically, we propose LinUOTD, a unified UOTD prediction framework with a linear model and high-dimensional (200 million) features. Fig. 1 illustrates the overview of LinUOTD. We first investigate multiple real-world datasets including original taxi demands (OTD), points of interest (POI) and meteorology. We then extract four types of basic features over space, time, meteorology and event domains, and generate massive combinatorial features based on the business logics of online taxicab platforms.

The high-dimensional features also bring in an extra challenge in *training an effective model with high-dimensional features on large-scale datasets*. Our LinUOTD framework addresses this challenge with a parameter-server based distributed framework to parallelize and accelerate model training and a hash-based feature tokenization scheme to enable parallel and scalable feature engineering. In addition, LinUOTD adopts $L1$ and $L2$ regularizations and designs a spatio-temporal regularization for accurate prediction while avoiding over-fitting.

Contributions. To the best of our knowledge, this is the first effort that adopts a simple linear model with very high-dimensional (hundreds of millions) features in predicting UOTD, to meet the requirements of *accuracy* and *flexibility* in large-scale online taxicab platforms. We transform the overhead of model redesign into feature engineering, and apply a distributed learning framework to support rapid, parallel and scalable feature updating and testing. Surprisingly, evaluations on two real datasets from the largest online taxicab platform in China reveal that our approach outperforms classical non-linear models in prediction accuracy. As a pilot study, we envision our successful experiences on adopting *simple linear models with high-dimensional features* can shed light upon other large-scale industrial spatio-temporal prediction problems.

The rest of the paper is organized as follows. We describe the real-world large datasets used in our study in Sec. 2, introduce our feature engineering in Sec. 3, and elaborate on the LinUOTD model and optimization techniques to handle high-dimensional features in Sec. 4. Evaluations on two large-scale industrial datasets are presented in Sec. 5. Finally we review related work in Sec. 6 and conclude this paper in Sec. 7.

2 DATA DESCRIPTION

The section introduces the datasets used for UOTD prediction, including original taxi demand records, geographical information and meteorological data. Table 1 summarizes the statistics of the raw datasets. Note that our datasets were collected in two metropolises in China (Beijing and Hangzhou).

Table 1: A summery of the sampled datasets.

Data Category	Beijing	Hangzhou
Raw taxi demand records	23851235	12354687
Raw POI records	12398746	9854621
Raw weather records	23445698	15468796

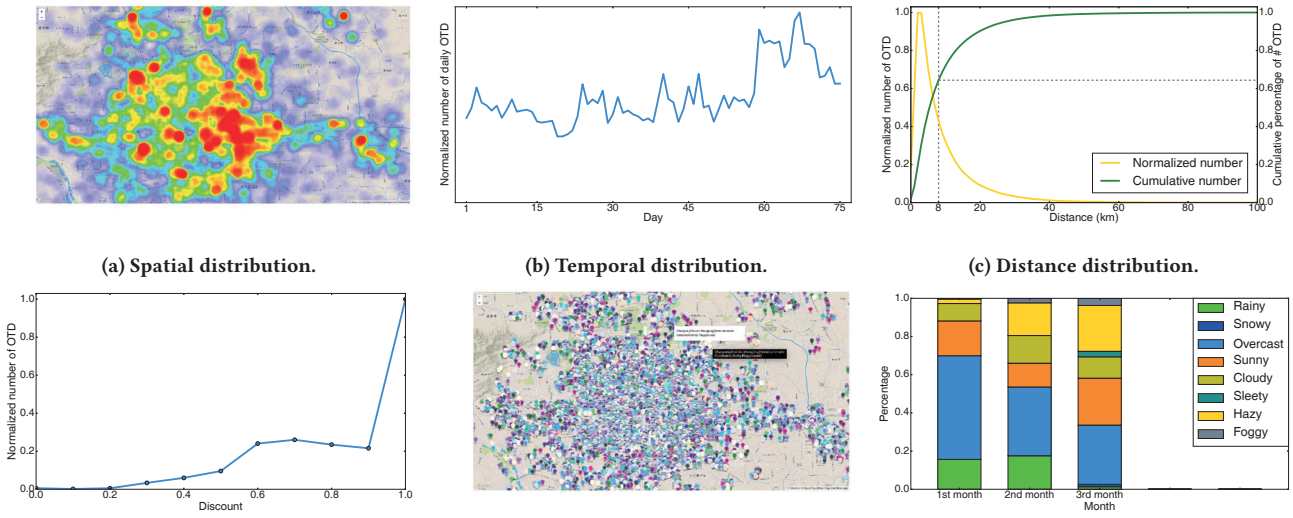
2.1 Original Taxi Demand Record Data

The original taxi demand records of the Beijing dataset are sampled in proportion from an online taxicab platform in China. The raw dataset contains 23,851,235 original taxi demand records on 75 successive days in three months in Beijing. The original taxi demand records of the Hangzhou dataset are sampled in proportion from the same online taxicab platform. The raw Hangzhou dataset contains 12,354,687 original taxi demand records on the same 75 successive days within three months. Each record in both datasets consists of a user ID, a time stamp, locations of the origin and destination, a distance estimate and the discount information. The user ID, time stamp, origin and destination are submitted by users. The distance estimate and the discount information are calculated by the taxicab platform based on the information submitted by users. User IDs and destinations are omitted since they are irrelevant to UOTD prediction [1].

Fig. 2a-Fig. 2d and Fig. 3a-Fig. 3d depict the distributions of the UOTD records from the Beijing dataset and the Hangzhou dataset, respectively. Here we take the Beijing dataset as an illustration. The heat map of the origin locations in all the original taxi demand records is shown in Fig. 2a. We observe that most original taxi demand gathers in the city center, dense residential areas and traffic hubs. For example, the red (hottest) dot in the right corner in Fig. 2a denotes the Beijing Capital International Airport. Fig. 2b plots the distribution of the normalized number of original taxi demand records per day. The number of original taxi demands fluctuates across the three months, indicating the taxi demands may be influenced by dynamic temporal factors such as weather. Fig. 2c shows the distribution (the yellow curve) and the cumulative percentage (the green curve) of the normalized number of original taxi demands with respect to different estimated distances. As shown, about 60% of the original taxi demand records are short trips within 8 km, indicating that short rides dominate the total taxi demand. Fig. 2d demonstrates the distribution of discounts, from which we see that most taxi demands get discounts within [0.6,1.0]. The above observations also hold for the Hangzhou dataset and we omit the details here.

There are also inter-city differences when comparing the corresponding distributions of UOTD between Beijing and Hangzhou. This is reasonable because of the differences in climate, economy and urban planning of the two cities. From the above observations, we decide to include data from other domains such as POI and meteorological information to predict UOTD for each city.

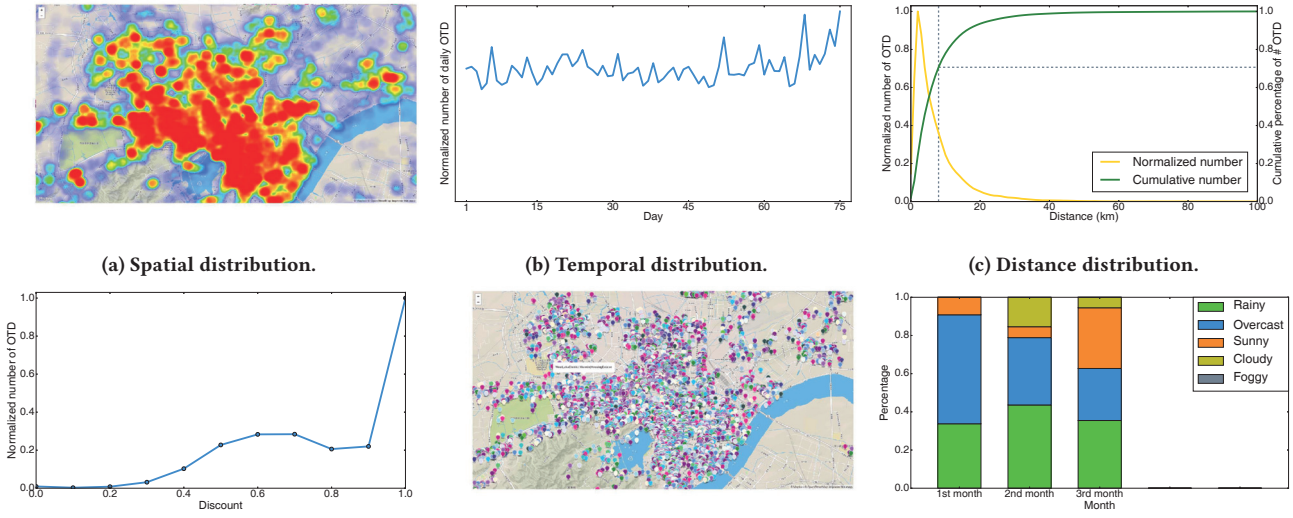
[1]Note that valuable information of the destination has been explicitly encoded in the distance estimate and the discount information.



(a) Spatial distribution. (b) Temporal distribution. (c) Distance distribution.

(d) Discount distribution. (e) POI distribution. (f) Weather distribution.

Figure 2: Distribution of the Beijing dataset: (a) spatial distribution of origin locations; (b) normalized numbers of daily original taxi demands in three successive months; (c) distribution of estimated taxi-trip distances; (d) distribution of discounts; (e) spatial distribution of POIs; (f) monthly distribution of weather information.



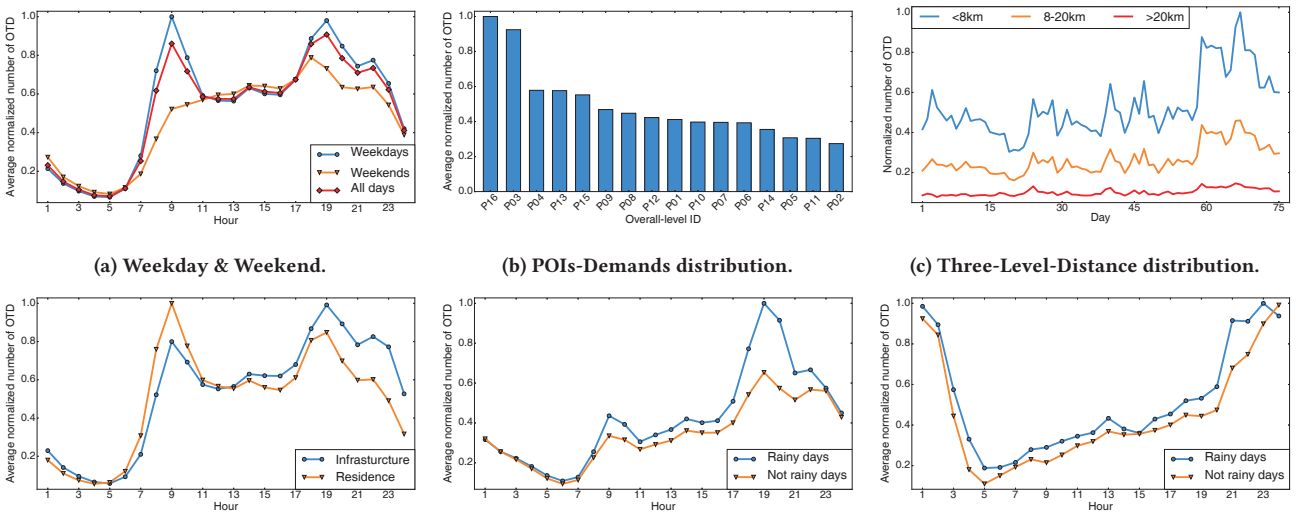
(a) Spatial distribution. (b) Temporal distribution. (c) Distance distribution.

(d) Discount distribution. (e) POI distribution. (f) Weather distribution.

Figure 3: Distribution of the Hangzhou dataset: (a) spatial distribution of origin locations; (b) normalized numbers of daily original taxi demands in three successive months; (c) distribution of estimated taxi-trip distances; (d) distribution of discounts; (e) spatial distribution of POIs; (f) monthly distribution of weather information.

2.2 POI Data

In addition to the original taxi demand record data, we also use a large-scale geographical information dataset from a major online map service provider in China. Specifically, we leverage the geographical information of Point Of Interest (POI). The Beijing POI dataset contains 55,447 distinct POIs in Beijing. Each record consists of a position, a name, an administrative district and a three-level category. For instance, *(116.49460 40.00057, Wangjing Playground, Changyang District, Entertainment:Ourdoor Activity:Playground)* is a POI record, where *(116.49460 40.00057)* represents the longitude and the latitude of the POI, *Wangjing Playground* denotes the POI name, *Changyang District* is an administrative district in Beijing, and *Entertainment:Ourdoor Activity:Playground* is the three-level category.



(a) Weekday & Weekend. (b) POIs-Demands distribution. (c) Three-Level-Distance distribution.

(d) POI-Hour distribution. (e) Weather-POI distribution (Entertainment). (f) Weather-POI distribution (Airport).

Figure 4: Distribution of features.

Table 2: Coarse-level categories of POIs.

ID	Category	Count	ID	Category	Count
P01	Tourist attraction	2385	P09	Parking lot	2764
P02	Life service	7109	P10	Entertainment	3443
P03	Residence	2634	P11	Cultural venues	1384
P04	Hotel	2234	P12	Health care	2749
P05	Sports	2406	P13	Company	2237
P06	Shopping	6245	P14	Dining	8942
P07	Organization	2236	P15	Education	2553
P08	Finance	2570	P16	Infrastructure	3556

The three-level category consists of a coarse-level (*Entertainment*), a mid-level (*Ourdoor Activity*) and a sub-level (*Playground*), respectively. In total the POIs are divided into 16 coarse-levels, 83 mid-levels, and 155 sub-levels. Table 2 lists the 16 coarse-level categories of POIs. POIs can help explain the motivations of taxi trips, and thus the spatio-temporal distributions of UOTD. For instance, shopping centers may have notable peak hours at weekends. Fig. 2e shows the spatial distribution of the POIs in Beijing. Since our goal is to predict UOTD for each POI, we preprocess the origin location of each original taxi demand record by associating its coordinates to the nearest POI. We adopt a similar approach to collect the POI dataset for Hangzhou, which contains 42,965 distinct POIs in Hangzhou. Fig. 3e shows the spatial distribution of the POIs in Hangzhou. Note that the POIs in Hangzhou share the same categories with the POIs in Beijing.

2.3 Meteorology Data

We further collect the meteorology data in Beijing during the corresponding three months from an online meteorology web of the Chinese government. Each meteorological record contains a time stamp and the information of weather condition, temperature, wind, humidity and air quality per hour. The weather condition is categorized into sleet, haze, snow, rain, clear, cloudy, fog and overcast. Fig. 2f plots the monthly weather distributions. It can be observed that there are more haze and less rain in Beijing from the first to the third month. The air quality is discretized into six levels: good, average, lightly polluted, moderately polluted, heavily polluted and severely polluted. In the meteorology dataset, 6% of the records are incomplete or empty. We complete the missing temperature, wind, and humility by averaging the values in the previous and the next hours. For missing information of weather condition and air quality, we use the same value as the ones in the previous hour. The meteorology data in Hangzhou is collected by the same approach as aforementioned, and the corresponding monthly weather distributions in Hangzhou are shown in Fig. 3f.

3 FEATURE ENGINEERING

As discussed in Sec. 1, the key insight of this work is to adopt high-dimensional features to compensate for the expressiveness of simple linear models so that they possess the predictive ability of complex non-linear models. In this section, we select the proper feature sets for UOTD prediction via feature engineering. Feature engineering is the process of using domain knowledge of the data to create features for representing the human's understanding about influence factors of complicated problems. These understandings include the influences from basic single factors and joint influence from combinational multiple factors. Specifically, for our UOTD prediction problem, we consider two categories of features: *basic features* and *combinational features*.

3.1 Basic Features

Basic features are extracted from each individual domains.

3.1.1 Temporal Features. We exploit *Month*, *Day of month*, *Day of week*, *Hour*, *Holiday* and *Historical UOTD* as the temporal features

Table 3: The description of features.

Feature Type	Feature	Description
Temporal	Month	The month which the time interval is in
	Day of month	The ordinal number of the day in a month
	Day of week	The ordinal number of the day in a week
	Hour	The time interval in a day
	Holiday	The length of the holiday (*e.g.*, Saturday is in a two-day holiday)
	Historical UOTD	The UOTD of the same POI of the same time period in the last N days
Spatial	District	The administrative district which the POI belongs to
	POI name	The name of the POI that the location is associated with
	POI category	The three-level category of the POI
	Distance distribution	The distribution of the estimated taxi-ride distances from the POI
Meteorological	Weather condition	The description of the weather condition in a time interval
	Temperature	The temperature measured by Celsius in a time interval
	Wind	The orientation and speed of the wind in a time interval
	Humidity	The index of humidity in a time interval
	Air quality	The discretized six levels of the air quality in a time interval
Event	Discount pricing strategy	The discount pricing strategy adopted by the online taxicab platform
	Even-odd license plate plan	Traffic restrictions on the last digit of the license plate numbers
	Version of the App	The version of the taxi-calling App
Combinational	Temporal-Spatial	(Month, Spatial features) (Day of month, Spatial features) (Day of week, Spatial features) (Hour, Spatial features) (Holiday, Spatial features)
	Temporal-Temporal	(Hour, Day of week) (Hour, Day of month) (Hour, Holiday)
	Meteorological-Spatial	(Weather condition, Spatial features) (Temperature, Spatial features) (Wind, Spatial features) (Humidity, Spatial features) (Air quality, Spatial features)
	Temporal-Meteorological	(Month, Meteorological features) (Day of month, Meteorological features) (Day of week, Meteorological features) (Hour, Meteorological features) (Holiday, Meteorological features)
	Temporal-Event	(Month, Event features) (Day of month, Event features) (Day of week, Event features) (Hour, Event features) (Holiday, Event features)
	Spatial-Temporal-Meteorological	(Spatial features, Hour, Meteorological features) (Spatial features, Day of month, Meteorological features) (Spatial features, Day of week, Meteorological features) (Spatial features, Holiday, Meteorological features)

(see Table 3 for the detailed explanations). Intuitively, the taxi demands exhibit distinctive temporal characteristics. Fig. 4a plots the distribution of the normalized hourly taxi demands during weekdays, weekends, and for all days. As shown, the demands have different temporal patterns between weekdays and weekends. Specifically, there are two peaks in weekdays, representing the morning peak and the evening peak, respectively. However, at weekends, there is the only a peak in the evenings.

3.1.2 Spatial Features. As observed from Fig. 2a, the taxi demands at different locations vary in patterns and biases. Therefore, we adopt *District, POI name, POI category and Distance distribution* as the spatial features. Fig. 4b plots the normalized taxi demands for each of the 16 coarse-level categories of POIs in Table 2. Accord with our intuition, more taxi demands are seen at POIs belonging to the categories of "Infrastructure" (*e.g.*, railway stations and airports) and "Residence", where there are consistently huge volume of mobility. We also plot the average normalized taxi demands for short, medium and long taxi rides in Fig. 4c. As shown, there are relatively stable demands for long ($> 20\,km$) rides across the entire time span of our datasets, while the demands for short rides ($< 8\,km$) dramatically fluctuate over different days. Thus if the taxi demands from a POI are dominated by long rides, it is likely that the taxi demands starting from this POI are stable over time.

3.1.3 Meteorological Features. Meteorological information such as weather can be an important consideration to alter transportation modes, and accordingly, an impacting factor on taxi demands. For example, in Fig. 2b, many peaks occurred because of bad weather conditions such as heavy rain, which leads to a surge of taxi demands. In addition to *Weather condition*, we also use *Temperature*, *Wind*, *Humidity* and *Air quality* as the meteorology features for the similar reasons.

3.1.4 Event Features. We use event features of *Discount pricing strategy*, *Even-odd license plate plan* and *Version of the App* because

they can affect the incentives of taxi-calling users, and consequently, the taxi demands. Fig. 2d shows the relationship between the average discount and the number of demands. It can be observed that when the platform provides more discounts, the number of demands tends to increase in general.

3.2 Combinational Features

Feeding basic features from multiple domains into a linear model only contributes to limited improvement in model expressiveness, because linear models fail to characterize correlations among input features. To boost the expressiveness of linear models, it is vital to harness combinational features, which explicitly account for the correlations among basic features. Combinational features also bring an extra advantage to simplify model updating. New features and new correlations can be seamlessly integrated by adding the corresponding combinational features without model redesign.

In principle, all correlated features need to be combined and fed into the linear model. Since the impacts of different features aggregate in linear models, feeding diversified combinational features into the model assists in characterizing the interplays among different factors from multi-scale and multi-aspect, which is the key to improve the model's predictive ability. The complete table of combinational features is shown in [17]. We interpret several illustrative examples of combinational features in the following.

3.2.1 Temporal-Temporal Combinational Features. As shown in Fig. 4a, the influence of hours in weekdays and at weekends are different. For example, the taxi demand during in weekdays shows a sharp peak at 8:00, yet almost no notable increase in the same hour at weekends. Therefore, we combine *Hour of day* and *Day of week* as a combinational feature.

3.2.2 Temporal-Spatial Combinational Features. Since the taxi demands vary both over time and space, it is rational to jointly consider temporal-spatial features. As an example, we plot the average hourly normalized taxi demands of Residence-category POIs and Infrastructure-category POIs in Fig. 4d. It can be observed that the taxi demands at both categories of POIs show two notable peaks throughout a day (7:00 to 9:00 and 17:00 to 21:00). Nevertheless, in Residence-category POIs, the higher peak is seen from 7:00 to 9:00, when most residents leave home for work. Conversely, the taxi demand in infrastructure-category POIs dramatically arises from 17:00 to 21:00, indicating large amounts of residents calling taxis at transport infrastructure such as bus stops to return home after work. A combination of *POI category* and *Hour of day* will capture such temporal-spatial dependency of taxi demands.

3.2.3 Meteorological-Spatial Combinational Features. The rationale to combine meteorological and spatial features is that the impact of meteorological information on taxi demands varies for POIs of different functionalities. Fig. 4e and Fig. 4f show the average hourly normalized taxi demands of an entertainment place and an airport in rainy and non-rainy days. In general the influence of the rain on the airport is not obvious. However, the demands of the entertainment place are sensitive to the rain, as shown by the huge gap of taxi demands between 15:00 and 20:00 in Fig. 4e.

3.2.4 Other Combinational Features. In addition to the above combinational features of two basic features, it is also feasible and necessary to compose combinational features among multiple basic features, *e.g.*, by combining POI, hour and weather, as a temporal-spatial-meteorological feature (see Table 3). Note that the inclusion of a complex combinational feature (*e.g.*, hour-POI-weather) does not necessarily mean the exclusion of the subsets of the combinational feature (*e.g.*, hour-POI). This is because the training samples for complex features are usually sparse, making them difficult to train effectively. In contrast, it is beneficial to keep a feature hierarchy of the complex feature (*i.e.*, its subsets) to train at multiple scales. After training subset features on relatively abundant training samples, it is plausible to effectively train complex combinational features even with sparse data. Following the above principles and the selection criteria from the understanding of the business logic of online taxicab platforms, we finally choose over 100 features consisting of 200 million dimensions.

4 MODEL AND OPTIMIZATION

In this section, we present our LinUOTD model, which is a linear regression model with high-dimensional features and a spatio-temporal regularizer (Sec. 4.1). To efficiently train our LinUOTD model, we adopt a parameter-server based distributed learning framework and a hash-based tokenization method (Sec. 4.2).

4.1 UOTD Prediction Model

Let y be the UOTD in a specific hour at a given POI, and $\boldsymbol{x} \in \mathbb{R}^m$ be the feature vector corresponding to y. Note that $\boldsymbol{x}$ is a very high dimension vector with more than two hundred million dimensions. Then, the raw data is reorganized to the set $D = \{(\boldsymbol{x}_i, y_i)|i = 1, 2, ..., N\}$, where $(\boldsymbol{x}_i, y_i)$ represents the UOTD and the corresponding feature vector of the i-th sample.

Our UOTD prediction model, LinUOTD, is a linear regression model with very high dimension features, which can be formulated as $p_i = \boldsymbol{w}'\boldsymbol{x}_i$, where $\boldsymbol{w}$ is the parameter vector to be learned, and p_i is the prediction result. The objective function is shown in (1), which is a squared error loss with $L1$ and $L2$ regularizations.

$$\text{obj}_{linear}(\boldsymbol{w}) = \sum_{i=1}^{N}(y_i - p_i)^2 + \lambda_1 \|\boldsymbol{w}\|_1 + \lambda_2 \|\boldsymbol{w}\|_2 \quad (1)$$

where λ_1 and λ_2 are the trade-off parameters for $L1$ and $L2$ regularizations, respectively.

Note that real-world UOTD records close in space or time tend to be similar. This smoothness requirement on the UOTD prediction results leads us to design the following objective function to reflect the spatio-temporal regularization:

$$\text{obj}_{spatio\text{-}temporal}(\boldsymbol{w}) = \sum_{\boldsymbol{X} \subseteq \boldsymbol{D}} \phi(\boldsymbol{X}) var(\{\boldsymbol{w}'\boldsymbol{x}|\boldsymbol{x} \in \boldsymbol{X}\}) \quad (2)$$

where $var()$ denotes the variance, $\boldsymbol{X}$ is a subset sampled from $\boldsymbol{D}$, and $\phi(\boldsymbol{X})$ maps subsets of POIs and times to a real value which controls the regularization of prediction variance of instances $\boldsymbol{x}$ in $\boldsymbol{X}$. In the following discussions, for the simplicity of notation, we use $\boldsymbol{X}$ to both represent the subset of $\boldsymbol{D}$, and the feature matrix consists of all $\boldsymbol{x} \in \boldsymbol{X}$. For the aforementioned spatio-temporal regularization, we devise $\phi(\boldsymbol{X})$ based on the radial basis function

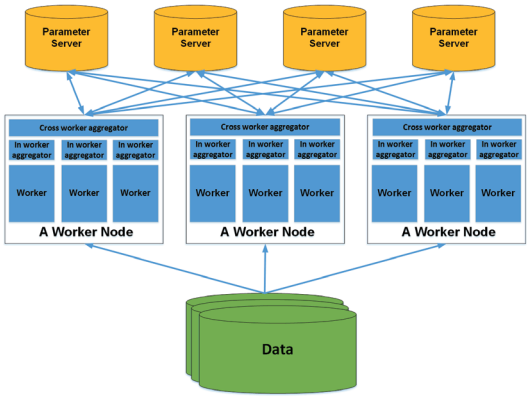


Figure 5: The architecture of the parameter-server based distributed learning framework.

kernel:

$$\phi(X) = \prod_{\boldsymbol{x} \in X} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(\boldsymbol{x}-\overline{X})'(\boldsymbol{x}-\overline{X})}{2\sigma}} \tag{3}$$

where $\overline{X}$ is the mean of data in X: $\overline{X} = \frac{1}{|X|}\sum_{\boldsymbol{x}\in X}\boldsymbol{x}$

Putting the above two regularizations together, the final objective function for LinUOTD is defined as:

$$\begin{aligned} \text{obj}_{LinUOTD}(\boldsymbol{w}) = &\sum_{i=1}^{N}(y_i - p_i)^2 + \lambda_1 \|\boldsymbol{w}\|_1 + \lambda_2 \|\boldsymbol{w}\|_2 \\ &+\gamma \sum_{X \subseteq D} \phi(X) var(\{\boldsymbol{w}'\boldsymbol{x} | \boldsymbol{x} \in X\}) \end{aligned} \tag{4}$$

where γ is the trade-off parameter which globally controls the influence of spatio-temporal regularization.

To minimize the objective function $\text{obj}_{LinUOTD}$ in (4), we adopt stochastic gradient descent with regulatized dual average (RDA) [22] and AdaGrad [26], as the FTRL algorithm in [13].

Without considering $L1$ and $L2$ regularization, we can minimize (4) by using minibatch based stochastic gradient descent:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \boldsymbol{g}_t, \tag{5}$$

where t indicates the number of iterations and η_t is the learning rate at tth iterationm and $\boldsymbol{g}_t$ represents the gradient at the tth iteration. We sample a minibatch $X \subseteq D$ from data at iteration t, which is represented in a matrix form $X_t = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l)'$, and we define $\overline{X_t} = (\overline{X_t}, \overline{X_t}, \ldots, \overline{X_t})'$. The derivative of the objective function in (4) without $L1$, $L2$ loss is $\boldsymbol{g}_t = X_t'(\boldsymbol{p}_t - \boldsymbol{y}_t) + \gamma\phi(X)(X_t'X - \overline{X_t}'\overline{X_t})\boldsymbol{w}$, where $\boldsymbol{p}_t$ and $\boldsymbol{y}_t$ are the prediction vector and label vector for data in X_t respectively.

We use RDA to handle the regularizations. The dual of (5) is

$$\boldsymbol{w}_{t+1} = argmin_{\boldsymbol{w}}(\sum_{s=1}^{t}\boldsymbol{g}_s\boldsymbol{w} + \frac{1}{2}\sum_{s=1}^{t}\sigma_s \|\boldsymbol{w} - \boldsymbol{w}_s\|_2^2 + \lambda_1 \|\boldsymbol{w}\|_1) + \lambda_2 \|\boldsymbol{w}\|_2) \tag{6}$$

and the closed-form solver of (6) is

$$w_{t+1,i} = \begin{cases} 0 & |z_{t,i}| \le \lambda_1 \\ -(\frac{\beta+\sqrt{n_i}}{\alpha} + \lambda_2)^{-1}(z_i - \lambda_1 sgn(z_i)) & \text{otherwise} \end{cases} \tag{7}$$

Algorithm 1: Training Process

input: training data D, parameters λ_1, λ_2

For all i, initialize $z_i = n_i = 0$;
foreach *train iteration* **do**
  foreach *minibatch* $X \subseteq D$ **do**
    Pull parameters $\boldsymbol{w}$ for all instances in X from parameter servers;
    Calcualte predition $\boldsymbol{p} = X\boldsymbol{w}$;
    Calculate gradient $\boldsymbol{g} = X'(\boldsymbol{p} - \boldsymbol{y}) + \gamma\phi(X)(X'X - \overline{X}'\overline{X})\boldsymbol{w}$;
    Push $\boldsymbol{g}$ to parameters servers (Call *Parameter updating*);

Following AdaGrad, we adapt per-coordinate learning rates to accelerate the learning process. For the i-th coordinate of $\boldsymbol{g}_t$, $\eta_{t,i}$ is calculated by

$$\eta_{t,i} = \frac{\alpha}{\beta + \sum_{s=1}^{t} g_{s,i}^2} \tag{8}$$

We refer readers to [13] for the deductions of the above formulas.

In the following, we present our optimization techniques in implementing the above algorithms for high-dimensional feature learning on massive datasets.

4.2 Implementation and Optimization

Despite the simple structure of LinUOTD, the 200 million dimensional features make it infeasible to train LinUOTD on a single machine. To efficiently learn the high-dimensional features, we exploit a parameter-server based distributed learning framework and a hash-based tokenization method, which we detail in sequel.

4.2.1 System Optimization. Fig. 5 illustrates the architecture of the parameter-server based distributed framework to train LinUOTD. The framework consists of multiple parameter servers and work nodes, and is fit for parallel machine learning on massive datasets. All the model parameters are stored evenly and distributively among the parameter servers, while the training data are dispatched to each work node when the training process starts.

During the training process, each work node runs multiple parallel workers (threads), which analyze the training samples in minibatches, fetch the corresponding parameters from the parameter servers via a global feature hashing function, and calculate the prediction values and the gradients. Algorithm 1 summarizes the gradient calculation process at each worker. Afterwards, the gradients of the same parameter will be aggregated first via in-worker aggregation and then be transferred among work nodes for cross-worker aggregation. Finally, all the newly calculated gradients will be pushed to the corresponding parameter servers, and each parameter server will update the parameters accordingly using the gradients received. Algorithm 2 details the process of parameter updating at each parameter server based on the formulas in Sec. 4.1. The above iteration continues till the end of the training process.

4.2.2 Feature Engineering Optimization. Feature tokenization is important in practical machine learning systems. By tokenization,

Algorithm 2: Parameter Updating

input : gradient g

index $I = \{i | g_i \neq 0\}$;
foreach $i \in I$ **do**
  $\sigma_i = \frac{1}{\alpha}(\sqrt{n_i + g_i^2} - \sqrt{n_i})$;
  $z_i = z_i + g_i - \sigma_i w_i$;
  $n_i = n_i + g_i^2$;
  if $|z_i| < \lambda_1$ **then**
    $w_i = 0$;
  else
    $w_i = -(\frac{\beta + \sqrt{n_i}}{\alpha} + \lambda_2)^{-1}(z_i - \lambda_1 sgn(z_i))$;

each feature, which can be either a value, a string or their combinations, is allocated a unique ID. Tokenization can be a bottleneck in efficiency for large-scale distributed machine learning systems. Since allocating a globally unique feature ID requires all the threads to share a lock, tokenization makes parallel computation difficult and inefficient. In fact, in industrial machine learning systems, tokenization sometimes takes longer time than training.

To break the bottleneck due to tokenization, we propose a hash-based tokenization method, which assigns an ID to each feature in a lock-free manner, while ensuring the global uniqueness of the IDs. Specifically, each discrete feature is hashed to a 64-bit space by hashing schemes such as murmurhash and cityhash. Hence the rate of hash collision remains low even for billion-dimensional features. Hash-based tokenization brings two advantages:

- Parallelism. Classical tokenization schemes cannot be parallelized due to the need of a shared lock among all threads. Conversely, hash-based tokenization is lock-free, thus enabling parallel computation.
- Scalability. To tokenize new data feature dimensions, hash-based tokenization simply needs to hash the new dimensions without re-allocating IDs.

The above hash-based tokenization is essential in our distributed parallel learning framework, which substantially accelerates the process of data preparation.

5 EXPERIMENTAL STUDY

This section presents the evaluations of our LinUOTD scheme.

5.1 The Experimental Setup

We evaluate the performance of our scheme on both the Beijing and the Hangzhou datasets. We chronologically order each dataset and use the first 3/4 for training and the remaining 1/4 for testing. In the experiments, we denote our approach as LinUOTD (Linear regression for UOTD prediction).

5.1.1 Baselines. We compare our LinUOTD method with the following state-of-the-arts as baselines.

- **Historical Average (HA):** predicting UOTD using the average of the historical UOTD during the same periods, *e.g.*, exploiting all UOTD data from 7:00-8:00 of all historical Fridays to predict the UOTD from 7:00-8:00 on a Friday.

Table 4: The performance of different methods.

Dataset	Method	ER	SMAPE	RMLSE
Beijing	HA	0.96957864	0.44033822	0.52884659
	ARIMA	0.89574376	0.42708392	0.50064628
	Markov	0.81039261	0.37087309	0.65547612
	GBRT	0.73525391	0.43042413	0.42926168
	NN	0.81226708	0.43515638	0.43978603
	HP-MSI	0.72515736	0.38083785	0.44228373
	LinUOTD	**0.6466814**	**0.35701066**	**0.40665828**
Hangzhou	HA	0.70616373	0.45098107	0.55787302
	ARIMA	3.16414193	0.46414572	0.59576175
	Markov	0.83794771	0.44441837	0.83023651
	GBRT	0.52536404	0.54445512	0.50110505
	NN	0.61526469	0.56586680	0.50200963
	HP-MSI	0.63366671	**0.43352982**	0.51046835
	LinUOTD	**0.54730029**	0.44870624	**0.49750043**

- **Auto-Regressive Integrated Moving Average (ARIMA):** predicting UOTD using the well-known time-series model.
- **Markov Model (Markov):** predicting UOTD by training a 3-order Markov predictor based on the UOTD of the 15 most recent corresponding periods, as in [28].
- **Gradient Boosted Regression Tree (GBRT):** predicting UOTD using non-parametric regression, which is one of the most effective statistical learning models for prediction.
- **Neural Network (NN):** predicting UOTD by training a neural network using the UOTD of 15 most recent corresponding periods (*e.g.*, the UOTD from 7:00-8:00 of 15 latest Fridays to predict the UOTD from 7:00-8:00 on a Friday) and all the basic features, as in [15, 28].
- **HP-MSI:** predicting UOTD adopting the state-of-the-art in predicting the number of bikes to be rent from or returned to each bike station [12].

5.1.2 Metrics. We use three metrics: *Error Rate (ER)*, *Symmetric Mean Absolute Percent Error (SMAPE)* and *Root Mean Squared Logarithmic Error (RMLSE)* for evaluation.

$$ER = \frac{\sum_{i=1}^{N} |p_i - y_i|}{\sum_{i=1}^{N} y_i}$$

$$SMAPE = \frac{2}{N} \sum_{i=1}^{N} \frac{|p_i - y_i|}{p_i + y_i + 1}$$

$$RMLSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (log(p_i + 1) - log(y_i + 1))^2}$$

where N is the number of testing data. p_i and y_i are the estimation and the ground truth of the i^{th} instance, respectively.

5.2 Overall Results

Table 4 summarizes the results of all the compared methods with respect to the three evaluation metrics. From the results, we make the following observations. *(i)* As expected, the naive HA performs poorly on both datasets. However, sometimes ARIMA and Markov are even worse than the naive HA method. A possible reason might be that time-series methods ignore the spatial variations of UOTD. Consequently, they tend to yield unstable prediction accuracies for different regions and thus unsatisfactory overall performance on large-scale datasets. *(ii)* NN and GBRT are two competitive methods. The reasons might be that both methods are supervised

Table 5: The top 10 features ranked by mutual information.

Rank	Feature	MI
1	(Day of week, Hour of day, POI name, POI category, Weather condition, Temperature, Air quality)	1.560136
2	(Day of week, Hour of day, POI name, POI category)	1.279322
3	(Temperature, POI name, District, POI category)	0.893714
4	(Hour of day, POI name, District, POI category)	0.834471
5	(Day of week, POI name, District, POI category)	0.531203
6	(Air quality, POI name, District, POI category)	0.496533
7	(Holiday, POI name, District, POI category)	0.435790
8	(POI name, POI category)	0.353553
9	(UOTD of 1 day ago)	0.299916
10	(UOTD of 7 days ago)	0.299857

non-linear models and able to extract spatio-temporal features from multiple heterogeneous data sources. *(iii)* Methods tailored for spatio-temporal prediction (HP-MSI and our LinUOTD) achieve the best overall performance. Our LinUOTD outperforms HP-MSI in almost all the metrics on the two datasets. The only exception is the SMAPE metric on the Hangzhou dataset, where HP-MSI yields slightly lower SMAPE. In summary, LinUOTD generally outperforms all the baselines (both time-series based approaches and complex, non-linear models) by adopting a simple linear regression model, indicating that properly selected massive feature sets can compensate for the simplicity of models in UOTD prediction.

5.3 Feature Contribution Analysis

To evaluate the effectiveness of both the basic and the combinational features, we list the top 10 features with respect to *Mutual Information (MI)* [5] in Table 5. It can be observed that the top 7 features are all combinational ones, demonstrating the necessity to include various cross-domain features. In particular, the most contributive feature is a multi-scale combination of temporal (*Day of week*, *Hour of day*), spatial (*POI name*, *POI category*), and meteorological (*Weather condition*, *Temperature*, *Air quality*) features.

The last two top features are basic temporal features, which accords with the intuition that UOTD is highly correlated to the latest historical UOTD from different time scales. Concretely, UOTD tends to vary smoothly within a short duration (*e.g.*, compared with the UOTD 1 day ago), and exhibits certain periodic patterns (*e.g.*, compared with the UOTD 7 days ago).

5.4 Prototype System

To ease researchers/engineers to utilize the UOTD prediction results for further analysis and strategy making, we implement a prototype system for users to navigate, query and visualize the results of UOTD prediction. Fig. 6 shows a screen shot of the prototype system. We use flatty (a template based on bootstrap) to build the front-end basic page and the mapbox API (a larger provider of custom online maps) for map related elements on the page. For the back-end, we use flask (a python microframework for web application) to process the requests. From the user side, the prototype works as follows. After the user submits the keyword for a POI and the current time (see the top), relevant POIs are shown (see the left), and the locations of the POIs are also marked on the map.

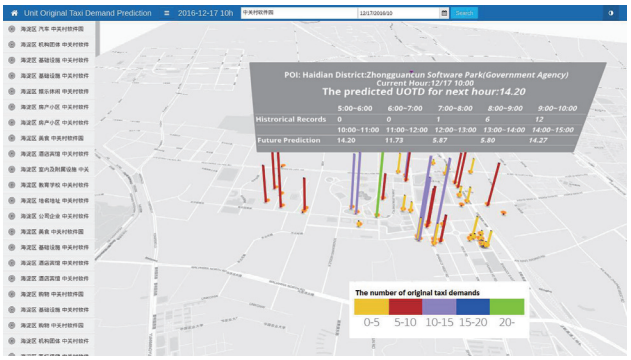


Figure 6: Prototype System Demo.

The user can also see the 3D views by selecting "View 3D". The predictions for the next hour for the POIs will be indicated by a bar of different heights and colors. As the user navigates the POIs by placing his/her cursor near a certain POI, the details of the UOTD prediction will be shown in a gray table, which includes the predictions for the next five hours and the historical UOTD records for the last five hours.

6 RELATED WORK

The design of LinUOTD is closely related to the following two categories of research.

6.1 Prediction of Taxi Demands

In this subsection, we review mainstream schemes on taxi demand prediction, which is one of the most important research topics in urban computing[23, 29, 30]. Depending on whether the prediction model requires taxi trajectories[24], we discuss *trajectory-based* prediction and *trajectory-free* prediction, respectively.

Trajectory-based Prediction. Zhang *et al.* [27] propose a framework combining clustering and time-series forecasting based on historical taxi trajectories to predict taxi demands in urban areas. Moreira-Matias *et al.* [14] design a model to predict the number of future services at a given taxi stand, where the GPS traces and event signals are transformed into a time series of interest as both a learning base and a streaming test framework. Yuan *et al.* [25] recommend places to pick up passengers quickly leveraging historical GPS trajectories of taxicabs. Li *et al.* [21] devise an improved ARIMA-based prediction model to forecast the spatio-temporal variations of passengers at a given hotspot using a large-scale GPS trace dataset. Zhao *et al.* [28] analyze over 14 million taxi pick-up samples in NYC and show a high predictability of the taxi demand. Moreover, Sun *et al.* [16] propose a predictive query to predict the aggregation number of objects according to the historical data of moving taxicabs. In addition, Anwar *et al.* [4] combine the trajectories of taxicabs and flight arrival data to predict the unmet taxi demands, which means the gap between taxi demands and potential supply of taxicabs at airport. The above trajectory-based schemes are not directly applicable in UOTD prediction, because trajectory information is not always associated with UOTD information, *e.g.*, the original taxi demands that have been cancelled.

Trajectory-free Prediction. Li *et al.* [12] propose to estimate the overall demand in a bike-sharing system, which can be easily

extended to predict taxi demands. In our work, we take the above trajectory-free prediction schemes as baselines, and show that our LinUOTD scheme outperforms the baseline in prediction accuracy.

6.2 Distributed Machine Learning with Parameter-Servers

Industrial applications usually adopt distributed machine learning frameworks to process with massive features on big datasets. Traditional distributed machine learning methods focus on "data parallelism", where each computing node needs to store the duplicates of all the parameters and models, which leads to enormous overhead in communication and storage. With the emergence of Distbelief model [6], the parameter-server based distributed framework is attracting increasing research interest [10, 11]. Our work applies a parameter-server based framework, yet differs from existing works [6, 10, 11, 13] in two system optimization techniques: *(i)* We design a hash-based feature tokenization scheme to enable parallel and scalable data preprocessing. *(ii)* We improve the efficiency of communication via request aggregation.

7 CONCLUSION

In this paper, we propose LinUOTD, a unified approach to predicting unit original taxi demands (UOTD) for large-scale online taxicab platforms. LinUOTD is a linear regression model with over 200 million dimensional features. The simple model structure facilitates easy model modification, while the high-dimensional features guarantee accurate prediction performance. We design a spatio-temporal regularization scheme, a distributed learning framework and a hash-based tokenization method to enable effective, parallel and scalable feature learning in a high-dimensional feature space on massive datasets. Extensive evaluations on two large-scale datasets from an industrial online taxicab platform validate the effectiveness of our approach. We envision our experiences of a simple linear model with massive features in UOTD prediction can serve as an insightful reference for various practical spatio-temporal prediction problems with both accuracy and flexibility requirements.

ACKNOWLEDGMENT

We are grateful to anonymous reviewers for their constructive comments on this work. Yongxin Tong is supported in part by National Grand Fundamental Research 973 Program of China under Grant 2014CB340300, NSFC Grant No. 61502021 and 71531001, and SKLSDE (BUAA) Open Program SKLSDE-2016ZX-13. Lei Chen is supported in part by Hong Kong RGC Project N HKUST637/13, NSFC Grant No. 61328202, NSFC Guang Dong Grant No. U1301253, HKUST-SSTSP FP305, Microsoft Research Asia Gift Grant, Google Faculty Award 2013 and ITS170. Qiang Yang is supported in part by Hong Kong CERG projects 621013, 16211214 and 16209715. Weifeng Lv is supported by NSFC Grant No. 61421003. Weifeng Lv is the corresponding author.

REFERENCES

[1] *Didi Chuxing*. https://en.wikipedia.org/wiki/Didi_Chuxing.
[2] *Grab*. https://en.wikipedia.org/wiki/Grab_(application).
[3] *Uber*. https://en.wikipedia.org/wiki/Uber_(company).
[4] A Anwar, M. Volkov, and D. Rus. 2013. Changinow: A mobile application for efficient taxi allocation at airports. In *16th International IEEE Conference on Intelligent Transportation Systems, The Hague, The Netherlands*. 694–701.
[5] T Cover and J. Thomas. 2006. *Elements of information theory (2. ed.)*.
[6] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, and Q. Le. 2012. Large scale distributed deep networks. In *Advances in neural information processing systems*. 1232–1240.
[7] J. Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001).
[8] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, and S. Bowers. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, New York City, New York, USA*. 5:1–5:9.
[9] G. Hinton and R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* (2006), 504–507.
[10] Q. Ho, J. Cipar, H. Cui, S. Lee, J. Kim, P. Gibbons, G. Gibson, G. Ganger, and E. Xing. 2013. More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in neural information processing systems*. 1223–1231.
[11] M. Li, D. Andersen, J. Park, A. Smola, A. Ahmed, V. Josifovski, J. Long, E. Shekita, and B. Su. 2014. Scaling distributed machine learning with the parameter server. In *11th USENIX Symposium on Operating Systems Design and Implementation, Broomfield, CO, USA*. 583–598.
[12] Y. Li, Y. Zheng, Y. Zhang, and L. Chen. 2015. Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, Bellevue, WA, USA*. 33:1–33:10.
[13] B. McMahan, G Holt, D Sculley, M. Young, D. Ebner, J. Grady, L. Nie, T. Phillips, E. Davydov, D. Golovin, S. Chikkerur, D. Liu, M. Wattenberg, A. Hrafnkelsson, T. Boulos, and J. Kubica. 2013. Ad click prediction: a view from the trenches. In *The 19th ACM International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA*. 1222–1230.
[14] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. 2013. Predicting taxi-passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems* (2013), 1393–1402.
[15] N. Mukai and N. Yoden. 2012. Taxi demand forecasting based on taxi probe data by neural network. In *Intelligent Interactive Multimedia: Systems and Services*. 589–597.
[16] J. Sun, D. Papadias, Y. Tao, and B. Liu. 2004. Querying about the past, the present, and the future in spatio-temporal. In *Proceedings of the 20th International Conference on Data Engineering, Boston, MA, USA*. 202–213.
[17] Y. Tong, Y. Chen, Z. Zhou, L. Chen, J. Wang, Q. Yang, and J. Ye. 2017. The simpler the better: a unified approach to predicting original taxi demands on large-scale online platforms (Techinical report). http://www.cse.ust.hk/~yxtong/tr_prediction.pdf. (2017).
[18] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu. 2016. Online Minimum Matching in Real-Time Spatial Data: Experiments and Analysis. In *Proceedings of the 42nd International Conference on Very Large Databases, New Delhi, India*. 109–118.
[19] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen. 2016. Online mobile Micro-Task Allocation in spatial crowdsourcing. In *32nd IEEE International Conference on Data Engineering, Helsinki, Finland*. 49–60.
[20] V. Vapnik and V. Vapnik. 1998. *Statistical learning theory*. Wiley New York.
[21] Li X., Pan G., Wu Z., Qi G., S. Li, D Zhang, W. Zhang, and Z. Wang. 2012. Prediction of urban human mobility using large-scale taxi traces and its applications. *Frontiers of Computer Science in China* (2012), 111–121.
[22] L. Xiao. 2010. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research* (2010), 2543–2596.
[23] J. Yuan, Y. Zheng, X. Xie, and G. Sun. 2011. Driving with knowledge from the physical world. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA*. 316–324.
[24] J. Yuan, Y. Zheng, C. Zhang, W. Xie, G. Xie, X.and Sun, and Y. Huang. 2010. T-drive: driving directions based on taxi trajectories. In *18th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, San Jose, CA, USA, Proceedings*. 99–108.
[25] J. Yuan, Y. Zheng, L. Zhang, X. Xie, and G. Sun. Where to find my next passenger. In *Proceedings of the 13th international conference on Ubiquitous computing, Beijing, China*. 109–118.
[26] M. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint* (2012).
[27] K. Zhang, Z. Feng, S. Chen, K. Huang, and G. Wang. 2016. A Framework for Passengers Demand Prediction and Recommendation. In *IEEE International Conference on Services Computing, San Francisco, CA, USA*. 340–347.
[28] K. Zhao, D. Khryashchev, J. Freire, C. Silva, and H. Vo. 2016. Predicting taxi demand at high spatial resolution: approaching the limit of predictability. In *2016 IEEE International Conference on Big Data, Washington DC, USA*. 833–842.
[29] Y. Zheng, L. Capra, O. Wolfson, and H. Yang. 2014. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology* (2014), 38:1–38:55.
[30] Y. Zheng, Y. Liu, J. Yuan, and X. Xie. 2011. Urban computing with taxicabs. In *Proceedings of the 13th international conference on Ubiquitous computing, Beijing, China*. 89–98.