TaxiRec: Recommending Road Clusters to Taxi Drivers Using Ranking-based Extreme Learning Machines

Ran Wang[1], Chi-Yin Chow[1], Yan Lyu[1], Victor C. S. Lee[1], Sam Kwong[1], Yanhua Li[2], Jia Zeng[3]
[1]Department of Computer Science, City University of Hong Kong, Hong Kong
[2]Department of Computer Science and Engineering, University of Minnesota, Twin Cities, USA
[3]Huawei Noah's Ark Lab, Hong Kong
{ranwang3-c@my., chiychow@, yanlv2-c@my., csvlee@, cssamk@}cityu.edu.hk, yanhua@cs.umn.edu, zeng.jia@huawei.com

ABSTRACT

Utilizing large-scale GPS data to improve taxi services becomes a popular research problem in the areas of data mining, intelligent transportation, and the Internet of Things. In this paper, we utilize a large-scale GPS data set generated by over 7,000 taxis in a period of one month in Nanjing, China, and propose TaxiRec; a framework for discovering the passenger-finding potentials of road clusters, which is incorporated into a recommender system for taxi drivers to hunt passengers. In TaxiRec, we first construct the road network by defining the nodes and road segments. Then, the road network is divided into a number of road clusters through a clustering process on the mid points of the road segments. Afterwards, a set of features for each road cluster is extracted from real-life data sets, and a ranking-based extreme learning machine (ELM) model is proposed to evaluate the passenger-finding potential of each road cluster. Experimental results demonstrate the feasibility and effectiveness of the proposed framework.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications-Spatial databases and GIS; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – Information Filtering

General Terms

Algorithms, Experimentation

Keywords

Extreme learning machine, passenger-finding potential, recommender system, taxi trajectory data

1. INTRODUCTION

SIGSPATIAL'15, November 03-06, 2015, Bellevue, WA, USA
 ISBN 978-1-4503-3967-4/15/11...$15.00
DOI: http://dx.doi.org/10.1145/2820783.2820843.

Taxis have been considered as a major means of transportation in modern cities because they are the most dynamic transportation tool, which offers great convenience to our daily life. However, compared with other transportation services such as buses and subways, taxi services are more difficult to manage for both passengers and taxi drivers, due to the high flexibility of routes and operating time [5].

In some big cities, taxis are equipped with GPS sensors, which enable them to report their locations in a certain frequency. A GPS record takes down the current information of a taxi, such as the ID, time, longitude, latitude, speed, direction, and occupation, etc. Since the number of taxis in a city is large and the report frequency is high, the GPS records are generated in a large scale of tens of Gigabytes on a daily basis. At the same time, they provide reliable information on the taxi trajectories, which reflect the behavior patterns of both passengers and taxi drivers. Recent years have witnessed an increasing interest in the applications based on taxi trajectory data, such as urban planning [15], visualization [12], route prediction [13], and recommender systems [14], etc.

Recent works have shown some practical values on recommender systems, which could help passengers or taxi drivers find their targets in a more effective manner. However, existing works only focus on using the knowledge learned from the historical trajectories, such as the passengers' mobility patterns [9], the taxi drivers' pick-up behaviors [14], and the spatiotemporal distributions of taxi-passenger demand [4, 6, 7]. They neglect many objective factors such as the density of POIs or road features. Besides, most related works utilize the techniques of landmark graph, clustering, and probability, etc. Very limited ones adopt supervised learning, which can make accurate predictions on unseen samples by training a model on a set of labeled samples. Motivated by the above statements, in this paper, we propose a recommendation framework called TaxiRec, which employs a supervised learning model to recommend road clusters to taxi drivers to help them find passengers. It is noteworthy that existing recommender systems for taxi drivers focus on recommending the passenger-finding strategy in a certain region or the most efficient path to a given destination. Different from the existing works, TaxiRec treats each road cluster as an evaluation unit and recommends the most potential road clusters to taxi drivers. Accordingly, a new evaluation metric is designed. We test our model on a real-world large-scale

trajectory data set generated by over 7,000 taxis in a period of one month in Nanjing, China.

The remainder of this paper is organized as follows. Section 2 gives some basic definitions and designs the features. In Section 3, we propose a ranking-based extreme learning machine (ELM) model for evaluating the passenger-finding potentials of road clusters. Experimental results are analyzed in Section 4. Finally, Section 5 concludes this paper.

2. ROAD SEGMENT CLUSTERING AND PASSENGER-FINDING POTENTIAL MODELLING

2.1 Preliminaries

Road network. We use a directed graph $G(V, E)$ to model a road network, where E and V are a set of road segments and intersection of road segments, respectively. A road segment can be either one-way or bidirectional. For long road segments, each of them is divided into several road segments with a maximum length of 500 meters[1]. A trajectory is a sequence of connected and ordered road segments.

Taxi. There are three possible status for a taxi: occupied ($\mathcal{O}$), cruising ($\mathcal{C}$), and parked ($\mathcal{P}$). Intuitively, a taxi with status $\mathcal{C}$ and $\mathcal{P}$ are available for picking up passengers. An action of a taxi could be pick up a passenger ($\mathcal{P} \rightarrow \mathcal{O}$ and $\mathcal{C} \rightarrow \mathcal{O}$) or drop off a passenger ($\mathcal{O} \rightarrow \mathcal{P}$ and $\mathcal{O} \rightarrow \mathcal{C}$).

Problem definition. Given a road network, POI data, and a set of taxi trajectories, we divide a road network into road clusters and estimate the passenger-finding potential for each road cluster based on a set of features extracted from the POI data and road properties. The top-K road clusters are recommended to taxi drivers. The key challenges are (1) how to design a set of representative features that can influence the passenger-finding potential of a road cluster, and (2) how to develop an effective and efficient supervised machine learning algorithm to train the model.

2.2 Road Segment Clustering

In a road network, many road segments are usually very short (e.g., the average length of road segments in our data set is about 120 meters). Such a short road segment is not a proper evaluation unit. Since the demand for a taxi is dynamic, if the evaluation unit is too small, it is possible that the predicted demand has already changed when the taxi driver gets there. To this end, road segments should be grouped into road clusters. The mid-point of a road segment is considered as its representative point. As the most widely used one, k-means clustering technique [1] is adopted, which partitions N observations, i.e., $\{s_1, s_2, \ldots, s_N\}$, into k clusters, i.e., $\{S_1, S_2, \ldots, S_k\}$, so as to minimize the intra-cluster sum of square: $\text{argmin} \sum_{i=1}^{k} \sum_{s_j \in S_i} ||s_j - \mu_i||$, where $\mu_i = \frac{1}{k_i} \sum_{s_j \in S_i} s_j$ and k_i is the number of observations in the i-th cluster.

The clustering process needs the number of clusters (i.e., k) as an input. In practice, given a certain driving speed (*speed*), a time period (*time*) and the total length of the road segments in a city (*length*), we can find a reference for k by first computing the total driving distance $speed \times time$ as the total length of road segments in a road cluster, and then determining k as $length/(speed \times time)$. Consider our data set as an example, the average driving speed of taxis in Nanjing is 50 km/hour. Given a driving time of 20 minutes, the total driving distance is 16.67 km. Since the total length of the 126,713 road segments in Nanjing is 15,298 km, we have $15,298/16.67 \approx 918$; by taking a reasonable approximation, we set $k = 1,000$. The clustering result in the urban area is shown in Fig. 1.

[1]By statistics, most of the road segments (>85%) in our data set are shorter than 500 meters.

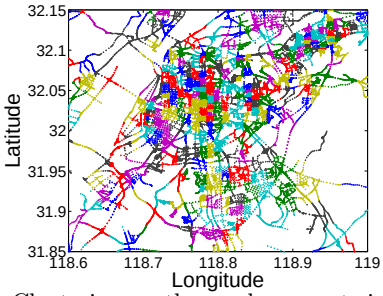


Figure 1: Clustering on the road segments in Nanjing.

Table 1: Types and numbers of road segments in Nanjing

Type	Number	Type	Number	Type	Number
Tertiary	22,020	Trunk	4,661	Secondary link	615
Secondary	17,644	Living street	4,032	Unknown	494
Primary	14,971	Path	2,316	Construction	476
Freeway	11,140	Track	1,939	Bridleway	227
Residential	10,052	Trunk link	1,555	Services	74
Minor road	9,364	Primary link	1,380	Rest area	18
Freeway link	9,010	Pedestrian	1,113	Raceway	13
Service	7,625	Tertiary link	652	Cycleway	9
Footway	4,686	Steps	622	Platform	5

2.3 Passenger-Finding Potential Modelling

Our passenger-finding potential model considers three major types of features.

Road types. The type of a road segment significantly influences the pick-up frequencies of taxis. For instance, freeway does not allow vehicles to stop, thus the pick-up frequency is zero. However, primary road with a high flow rate of visitors has a high pick-up frequency. In our work, we summarize all the road segments into 27 types as listed in Table 1, and the type of a road cluster is decided by the main type with the maximum number of road segments in it.

Road length. Although passengers are not uniformly distributed in reality, for two road segments of the same type, the longer one usually has a higher chance for a taxi to pick up passengers than the shorter one. In TaxiRec, the length of a road cluster is calculated as the sum of the lengths of all its road segments.

Points-of-interest (POIs). Another important factor that can influence the passenger-finding potential is the density of POIs near the road segment. As a consensus, taxis will have a higher chance to pick up passengers on the road that is near a larger number of POIs, and vice versa. In this paper, we extracted 66,160 POIs in Nanjing from *Weibo*[2], a popular Chinese social network web site. All the POIs are categorized into 13 types, as depicted in Table 2.

Having these preliminaries, it is possible to model the passenger-finding potentials of road clusters by the designed features. The basic idea is to treat each cluster as a sample $(\mathbf{x}_i, \mathbf{y}_i)$, which is composed of n input features $\mathbf{x}_i = [x_{i1}, \ldots, x_{in}]^{\mathrm{T}}$ ($n = 15$ here), and one output decision value $\mathbf{y}_i$ (the number of pick-up actions). Then, a regression model could be constructed on labeled samples, in order to find the relationship between $\mathbf{x}_i$ and $\mathbf{y}_i$. This model is then

[2]http://weibo.com

Table 2: Types and numbers of POIs in Nanjing

Type	Number	Type	Number
Shop	15,878	Entertainment	4,399
Life service	8,168	Education	4,339
Government & social organizations	6,516	Medical care	3,711
		Company	2,840
Restaurant	5,894	Transportation	2,342
Bank & insurance	5,352	Residence	2,313
Hotel	4,460	Attraction	1,428

used to predict the decision value of unlabeled samples.

3. RANKING-BASED ELM MODEL

ELM [3] is an emergent technique for training single-hidden layer feedforward neural networks (SLFNs) [8]. It randomly assigns the input weights and biases, and analytically determines the output weights by the pseudo-inverse of the hidden layer output matrix. Given a training set $\mathbb{X} = \{(\mathbf{x}_i, \mathbf{y}_i)|\mathbf{x}_i \in \mathcal{R}^n, \mathbf{y}_i \in \mathcal{R}^L, i = 1, \ldots, N\}$, where $\mathbf{x}_i = [x_{i1}, \ldots, x_{in}]^{\mathrm{T}}$ is the input feature set, and $\mathbf{y}_i = [y_{i1}, \ldots, y_{iL}]^{\mathrm{T}}$ is the output decision vector. The standard SLFNs with $\widetilde{N}$ hidden nodes and activation function $g(x)$ can be mathematically modeled as $\sum_{j=1}^{\widetilde{N}} \beta_j g(\mathbf{w}_j \cdot \mathbf{x}_i + b_j) = \mathbf{o}_i, i = 1, \ldots, N$. The standard SLFNs are proved to be capable of approximating the N samples with zero error, i.e., $\mathbf{o}_i = \mathbf{y}_i, i = 1, \ldots, N$. Thus, there exist β_j, $\mathbf{w}_j$, and b_j such that $\sum_{j=1}^{\widetilde{N}} \beta_j g(\mathbf{w}_j \cdot \mathbf{x}_i + b_j) = \mathbf{y}_i, i = 1, \ldots, N$. This formulation can be written compactly as $\mathbf{H} \cdot \beta = \mathbf{Y}$, where $\mathbf{H}$ is the hidden layer output matrix, $\beta = [\beta_1, \ldots, \beta_{\tilde{N}}]^{\mathrm{T}}$, and $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^{\mathrm{T}}$. Having the above notations, the training and testing process of ELM are presented in Algorithms 1 and 2, respectively. Note that (1) when the problem is classification, L is the number of classes, and $\mathbf{y}_{ij}$ is the class membership of $\mathbf{x}_i$ in the j-th class, where $i = 1, \ldots, N$ and $j = 1, \ldots, L$; (2) when the problem is regression, $L = 1$ and $\mathbf{y}_{i1}$ is the regression value of $\mathbf{x}_i$.

Algorithm 1: Training Process of ELM

Input: Training set $\mathbb{X} = \{(\mathbf{x}_i, \mathbf{y}_i)|\mathbf{x}_i \in \mathcal{R}^n, \mathbf{y}_i \in \mathcal{R}^L, i = 1, \ldots, N\}$; activation function $g(x)$; number of hidden nodes $\tilde{N}$.
Output: Input weight $\mathbf{w}_j$; input bias b_j; output weight β.

1 Randomly assign input weight $\mathbf{w}_j$ and bias b_j where $j = 1, \ldots, \tilde{N}$;
2 Calculate the hidden layer output matrix $\mathbf{H}$;
3 Calculate the output weight $\beta = \mathbf{H}^{\dagger}\mathbf{Y}$ where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$.

Algorithm 2: Testing Process of ELM

Input: Testing set $\mathbb{T} = \{\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_i \in \mathcal{R}^n, i = 1, \ldots, M\}$; input weight $\mathbf{w}_j$; input bias b_j; output weight β.
Output: Output value $\hat{\mathbf{y}}_i$ for each testing sample $\hat{\mathbf{x}}_i$.

1 Calculate the hidden layer output matrix

$$\hat{\mathbf{H}} = \begin{bmatrix} g(\mathbf{w}_1 \cdot \hat{\mathbf{x}}_1 + b_1) & \cdots & g\left(\mathbf{w}_{\tilde{N}} \cdot \hat{\mathbf{x}}_1 + b_{\tilde{N}}\right) \\ \vdots & \ddots & \vdots \\ g(\mathbf{w}_1 \cdot \hat{\mathbf{x}}_M + b_1) & \cdots & g\left(\mathbf{w}_{\tilde{N}} \cdot \hat{\mathbf{x}}_M + b_{\tilde{N}}\right) \end{bmatrix}_{M \times \tilde{N}};$$

2 Calculate the testing output matrix $\hat{\mathbf{Y}}_{M \times L} = \hat{\mathbf{H}}_{M \times \tilde{N}} \beta_{\tilde{N} \times L}$;
3 **if** *the problem is classification* **then**
4 $\quad \hat{\mathbf{y}}_i = \mathrm{argmax}_{j=1,\ldots,L} \hat{\mathbf{Y}}_{ij}$;
5 **else if** *the problem is regression* **then**
6 $\quad \hat{\mathbf{y}}_i = \hat{\mathbf{Y}}_{i1}$.
7 **end**

Different from the traditional gradient based training methods such as back-propagation (BP), ELM treats the problem as a linear system, and does not include any iteration. It is stated in [2] that ELM has many advantages compared with the traditional methods. It not only exhibits extremely fast learning speed, but it also gives the least-square solution to the linear system.

However, it is argued in [11] that with the randomly assigned input weights, ELM usually suffers from the unstable problem, which shows a weak robustness to many problems. Wang et al. [11] made a study on the random weights between input and hidden layers, which shows that the random mapping can outperform some kernel mappings in many cases. Later, they made an analysis on the ELM approximate error based on the random weights [10], but the unstable problem is still hard to overcome.

In this paper, we propose a ranking-based ELM model for the road cluster evaluation problem (Algorithm 3). In ELM, different input weights will generate different results. However, it is difficult to evaluate the quality of the weights in the training process. A possible solution is to integrate the results of a set of ELMs, in order to eliminate or weaken the unstable factor of the random mechanism. Besides, taxi drivers are always interested in the ranking order of the road clusters, rather than the actual number of pick-up actions. Thus, it is important to treat it as a ranking process, rather than a traditional regression problem.

Algorithm 3: Ranking-based ELM for Road Cluster Evaluation

Input: Training set with labeled road clusters $\mathbb{X} = \{(\mathbf{x}_i, \mathbf{y}_i)|\mathbf{x}_i \in \mathcal{R}^n, \mathbf{y}_i \in \mathcal{R}, i = 1, \ldots, N\}$; testing set with unlabeled road clusters $\mathbb{T} = \{\hat{\mathbf{x}}_i|\hat{\mathbf{x}}_i \in \mathcal{R}^n, i = 1, \ldots, M\}$; activation function $g(x)$; number of hidden nodes $\tilde{N}$; number of ELMs C.
Output: Ranking order of all the clusters.

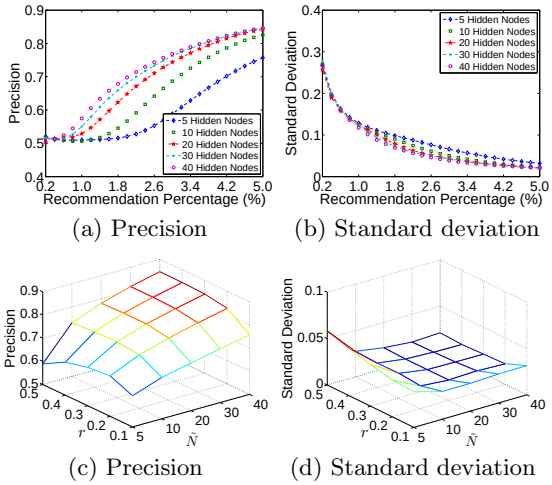
1 **for** $c = 1, \ldots, C$ **do**
2 $\quad$ Call Algorithm 1 on $\mathbb{X}$ with $g(x)$ and $\tilde{N}$, get an ELM with $\mathbf{w}_j$, b_j and β;
3 $\quad$ Call Algorithm 2 (regression mode) on $\mathbb{T}$ with $\mathbf{w}_j$, b_j and β, get the output value $\hat{\mathbf{y}}_i$ for each $\hat{\mathbf{x}}_i$;
4 $\quad$ Re-denoted all the training and testing clusters as $\{(\mathbf{x}_i, \mathbf{y}_i)|\mathbf{x}_i \in \mathcal{R}^n, \mathbf{y}_i \in \mathcal{R}, i = 1, \ldots, N + M\}$, and rank them with descending order of $\mathbf{y}_i$, where the ranking result for $\mathbf{x}_i$ is denoted as $\mathbf{r}_{ic}$;
5 **end**
6 **for** *each cluster* $\mathbf{x}_i, i = 1, \ldots, N + M$ **do**
7 $\quad$ Calculate the average ranking result as $\mathbf{r}_i = \sum_{c=1}^{C} \mathbf{r}_{ic}/C$;
8 **end**
9 Sort all the clusters with ascending order of $\mathbf{r}_i$;
10 **return** the ranking order.

4. EXPERIMENTAL ANALYSIS

Existing recommender systems for taxi drivers focus on recommending the passenger-finding strategy in a certain region or the most efficient path to a given destination. To the best of our knowledge, TaxiRec is first work treating each road cluster as an evaluation unit. Our trajectory data are collected from over 7,000 taxis in one month in Nanjing, China, road network data is obtained from OpenStreetMap, and POI data are crawled from Weibo.

Experiment design. In the experiment, 48 data sets are tested, representing the 48 time intervals in a day (i.e., (1) 0:00 to 0:30, (2) 0:30 to 1:00, ... (48) 23:30 to 0:00). All the experiments are conducted on different days separately and the average results are recorded. More specifically, we realize Algorithm 3 with randomly selected training clusters. Since the random mechanism may cause some unstable problems, we repeat the experiment 20 times, then observe the average value and standard deviation. The activation function in ELM is set as sigmoid function $g(x) = \frac{1}{1+\exp(x)}$, and the number of ELMs in Algorithm 3 is $C = 10$. The experiments are performed under MATLAB 7.9.0, which are executed on a computer with an Intel Core 2 Duo CPU, 4GB memory, and 32-bit Windows 7 system.

Performance metric. Traditional recommender systems usually adopt the precision and recall to evaluate the performance. Precision defines the fraction of the recom-



(a) Precision (b) Standard deviation

(c) Precision (d) Standard deviation

Figure 2: Average result of the 48 time intervals. (a) to (b) with $r = 0.5$. (c) to (d) with $p = 5\%, K = 50$.

mended objects that are relevant, and recall defines the fraction of relevant objects that are recommended. After all the road clusters are ranked into an order by TaxiRec, the top-K clusters are selected from this order and are recommended to the taxi driver. The recommendation precision and recall are defined as

$$\text{Precision} = \frac{|\text{Real top } K \text{ clusters} \cap \text{Ranked top } K \text{ clusters}|}{|\text{Ranked top } K \text{ clusters}|}$$

$$\text{Recall} = \frac{|\text{Real top } K \text{ clusters} \cap \text{Ranked top } K \text{ clusters}|}{|\text{Real top } K \text{ clusters}|},$$

where K is smaller than the number of clusters. In our problem, $|\text{Real top } K \text{ clusters}| = |\text{Ranked top } K \text{ clusters}| = K$. Thus, precision and recall give the same result; hence, we will omit the results for recall.

Results analysis. We first fix the ratio of training samples as $r = 0.5$, and observe the results with $K = \{2, 4, 6, \ldots, 50\}$, which account for the recommendation percentage of p=$\{0.2\%,0.4\%,0.6\%,\ldots,5\%\}$. Afterwards, we fix the recommendation percentage as $p = 5\%$ ($K = 50$), and observe the impacts of parameters $(r, \tilde{N})$, where $r = \{0.1, 0.2, 0.3, 0.4, 0.5\}$ is the ratio of training clusters, and $\tilde{N} = \{5, 10, 20, 30, 40\}$ is the number of hidden nodes in ELM. Fig. 2 depicts the average precision and standard deviation of the 48 time intervals. Basically, we have three key observations. (1) A higher recommendation percentage (p) results in a higher precision and a lower standard deviation. Since when p is higher, the number of evaluated clusters is larger, which gives a higher chance to recommend the potential ones. Obviously, when $p = 100\%$, the precision will be 1, and the standard deviation will be 0. (2) Both r and $\tilde{N}$ can largely influence the result. When p is fixed, a higher r and a larger $\tilde{N}$ can help improve the precision and reduce the standard deviation. (3) The improvement becomes very limited when the value of $\tilde{N}$ reaches a certain level. For instance, the result of $\tilde{N} = 10$ is much better than that of $\tilde{N} = 5$, but the result of $\tilde{N} = 40$ is just slightly better than that of $\tilde{N} = 30$. Since a larger $\tilde{N}$ increases the time complexity of ELM, it is important to realize a trade-off between the precision and efficiency.

5. CONCLUSION

In this paper, we have designed TaxiRec; a framework aims to recommend road clusters with the highest passenger-finding potentials to taxi drivers. In TaxiRec, we first conduct a clustering process to divide a road network into smaller road clusters. Then, a feature set is designed for each road cluster, which reflects its basic properties with regard to various factors (i.e., its main road type, its total length, the numbers of different types of POIs in the cluster). Afterwards, the training clusters are randomly selected for labeling (counting the number of pick-up actions during a certain time interval). These clusters are used to train an ELM regression model, which predicts the numbers of pick-up actions for the unlabeled clusters. Finally, we rank all the clusters in terms of their regression values and recommend top-K road clusters to taxi drivers. We have evaluated the performance of TaxiRec through extensive experiments based on a trajectory data set of over 7,000 taxis in Nanjing for one month, the road network data retrieved from OpenStreetMap, and the POI data crawled from Weibo. Experimental results show the feasibility and effectiveness of TaxiRec.

6. REFERENCES

[1] J. A. Hartigan and M. A. Wong. Algorithm AS 136: A k-means clustering algorithm. *Applied Statistics*, pages 100–108, 1979.

[2] G. B. Huang, H. Zhou, X. Ding, and R. Zhang. Extreme learning machine for regression and multiclass classification. *IEEE TSMCB*, (99):1–17, 2010.

[3] G. B. Huang, Q. Y. Zhu, and C. K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

[4] B. Li, D. Zhang, L. Sun, C. Chen, S. Li, G. Qi, and Q. Yang. Hunting or waiting? Discovering passenger-finding strategies from a large-scale real-world taxi dataset. In *IEEE PerCom Workshops*, 2011.

[5] S. Liu, J. Pu, Q. Luo, H. Qu, L. M. Ni, and R. Krishnan. VAIT: A visual analytics system for metropolitan transportation. *IEEE TITS*, 14(4):1586–1596, 2013.

[6] S. Ma, Y. Zheng, and O. Wolfson. T-Share: A large-scale dynamic taxi ridesharing service. In *IEEE ICDE*, 2013.

[7] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas. On predicting the taxi-passenger demand: A real-time approach. In *Progress in Artificial Intelligence*, pages 54–65. Springer, 2013.

[8] A. Nigrin. *Neural networks for pattern recognition*. The MIT press, 1993.

[9] G. Qi, G. Pan, S. Li, Z. Wu, D. Zhang, L. Sun, and L. T. Yang. How long a passenger waits for a vacant taxi? Large-scale taxi trace mining for smart cities. In *IEEE GreenCom*, 2013.

[10] R. Wang, S. Kwong, and D. D. Wang. An analysis of ELM approximate error based on random weight matrix. *IJUFKS*, 21(supp02):1–12, 2013.

[11] R. Wang, S. Kwong, and X. Wang. A study on random weights between input and hidden layers in extreme learning machine. *Soft Computing*, 16(9):1465–1475, 2012.

[12] Z. Wang, M. Lu, X. Yuan, J. Zhang, and H. v. d. Wetering. Visual traffic jam analysis based on trajectory data. *IEEE TVCG*, 19(12):2159–2168, 2013.

[13] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang. T-Drive: Driving directions based on taxi trajectories. In *ACM SIGSPATIAL*, 2010.

[14] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie. T-Finder: A recommender system for finding passengers and vacant taxis. *IEEE TKDE*, 25(10):2390–2403, 2013.

[15] Y. Zheng, Y. Liu, J. Yuan, and X. Xie. Urban computing with taxicabs. In *ACM UbiComp*, 2011.