Taxi Dispatch Planning via Demand and Destination Modeling

Jun Xu, Rouhollah Rahmatizadeh, Ladislau Bölöni, and Damla Turgut
Department of Computer Science, University of Central Florida, Orlando, FL
Email: {junxu,rrahmati,lboloni,turgut}@cs.ucf.edu

***Abstract*—In this paper, we focus on a taxi dispatch system with the help of auxiliary models that predict future demand and destination. We build two different neural networks for learning taxi demand and destination distribution patterns based on historical data. The trained models can predict taxi demand and destination for any area in a city at a future time. Our proposed dispatch system relies on the predictions of the previous models and is designed not only to minimize the waiting time of passengers, but also to assign the taxis to passengers in a way to minimize the idle driving distances of taxis. In order to achieve this, we balance future taxi supply-demand over the city by solving a mixed-integer program (MIP). We validate our dispatch system as well as the prediction models using a dataset of taxi trips in the New York City.**

***Index Terms*— taxi dispatch; demand prediction; destination prediction; recurrent neural network; optimization.**

I. Introduction

Traditional transportation systems with all the comfort that provides to humans, still face serious challenges due to the rapid growing traffic and inefficient dispatch operation. For instance, taxi drivers often drive for a long time to pickup a passenger and passengers often need to wait for a long time before a taxi picks them up. In addition to the wasted times, this will lead to a wide variety of problems such as more fuel consumption, traffic congestion, and air pollution [1]. To address these problems, intelligent transportation systems such as vehicle rebalance and ridesharing systems are proposed by the researchers. To make these systems more efficient, we need to understand and predict the demand of the passengers since these systems highly rely on the future demand patterns. Previous studies [2–5] have shown that historical taxi trip data can provide rich insights about how taxi demand varies from area to area and time to time. In addition, the destination prediction also plays an important role in a transportation system as it provides more detailed vehicle fleet distribution information for a dispatch center. We can also look a bit ahead and consider the future where self-driving vehicles need to autonomously decide where to look for passengers and also to balance the supply-demand ratios over the city without human help [6]. To achieve these, robust and efficient prediction models are necessary and also can be of great help to passengers, human drivers, and autonomous vehicles.

In our previous work [6], we presented a taxi demand predictor in which we make real time taxi demand predictions for the whole city. In this paper, we further extend our previous work and predict the destination of each trip. We present two different neural networks to learn historical taxi demand and destination distribution patterns. With the trained models, we can predict taxi demand and the corresponding destination for future time at any area throughout the city. In addition, we build a taxi dispatch system in which the predicted demand and destination are used for taxis supply-demand rebalance throughout the city. We optimize the taxi assignment and reallocation by solving a mixed integer programming (MIP) with the goal of minimizing the average waiting time of the passengers and the idle driving distances of the taxis.

We divide the entire city into about 1000 areas with Geohash [7] library and for each area, we encode taxi trips into specific sequential data structure for the learning models. We use Recurrent Neural Networks (RNNs) to learn the taxi demand pattern at each area. Affecting factors such as date, time steps in the day, day of the week and weather information are used to train the model. For the destination prediction model, we treat it as one input multiple possible outputs problem and use Mixture Density Networks (MDNs) [8] to model the outputs distribution. Instead of learning and predicting a destination area directly, the proposed model learns the destination pattern and outputs a distribution over all the areas in the city. We train both the demand and destination prediction models with one year taxi data in 2015 from the New York City taxi trip dataset [9]. We evaluate the performance of the prediction models and the dispatch system with taxi data in 2016 from the same dataset.

The remainder of this paper is organized as follows. Section II introduces related work on taxi demand and destination predictions as well as intelligent transportation systems. Section III describes the proposed taxi demand and destination distribution learning models. Section IV presents the proposed dispatch system. In Section V, we provide the experimental results. Section VI concludes the paper.

II. Related Work

Taxi demand prediction is a current research topic since it is one of the key strategies to improve the taxi dispatch performance as well increasing the sustainability of taxi companies. Zhao et al. [3] define a maximum predictability based on the real entropy of historical taxi demand. They prove that taxi demand is highly predictable and then propose three prediction algorithms to validate their maximum predictability theory. Moreira-Matias et al. [5] propose a prediction framework that consists of three different models and their predicted

978-1-5386-4413-3/18/$31.00 ©2018 IEEE

result is a weighted ensemble of outputs from those models. The ensemble weights are updated according to the previous prediction performance of each model. Their framework can make real-time demand prediction for the 63 taxi stands in the city of Porto, Portugal. Miao et al. [10] propose a dispatch framework for balancing taxi demand and supply throughout a city. In their work, the future taxi demand is predicted by the mean value of repeated samples from historical demand.

Taxi destination prediction is more complex than the demand prediction because it contains more uncertainty. Some well-performing models are using a small window of GPS traces to predict the destination of each trip [11]. We consider a different scenario in which we predict possible destinations for future taxi trips without relying on their GPS traces. In some dispatch systems [12] the destination estimation is sampled from a normalized distribution of destinations. However, the distribution is simply the historical average of the destinations. In this paper, we use a more powerful model based on deep neural networks that can learn highly nonlinear functions to capture patterns in the data.

Given the estimated future demand and destination, different intelligent transportation systems have been proposed. Zhang et al. [4] propose a real-time taxi dispatch application. Two types of passengers are defined to model real-time taxi demand: previously left-behind, and passengers arriving shortly. A demand inference model called Dmodel is designed with hidden Markov chain to describe the state changes of passengers. Transportation system with ridesharing is also a recent popular topic with the hope of improving the utilization of taxis. Chen et al. [1] propose a system for vehicle dispatch and ridesharing. The goal is to balance the taxi supply-demand ratio while minimizing the idle mileage. Ridesharing is achieved by solving the taxis schedule with a Mixed Integer Programming (MIP). Lin et al. [13] present a dispatch system for transportation hubs with steady passenger streams. In their work, virtual demand pools, passengers walking time and ridesharing mechanism are considered. Trips pairing and taxi scheduling are done by a MatchMaking system. Similar studies based on ridesharing systems are conducted in [12], [14].

Our work focuses more on learning and predicting taxi demand and destination distribution patterns. Different from existing works, we use a recurrent neural network to capture long term dependencies in the sequence of taxi demand patterns. For the destination prediction, our model predicts the entire probability distribution over all areas in the city instead of sampling from the pre-seen destinations frequencies. Our approach gives a more realistic prediction as it takes into account the uncertainty while predicting. With the predicted results, we further build a dispatch system with the goal of balancing future taxi supply and demand over the city while minimizing the passengers' average waiting time and the taxis' idle driving distances.

III. Taxi Demand and Destination Prediction

In this section, we discuss the taxi demand prediction and the corresponding destination prediction.

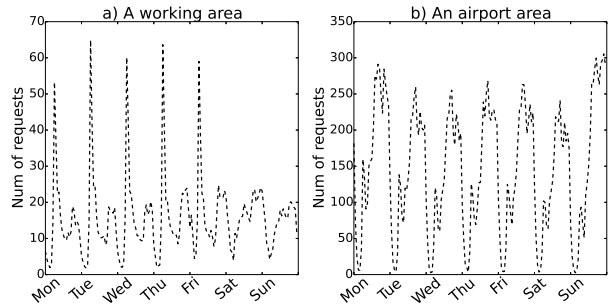

Fig. 1. Taxi demand patterns in two different areas.

A. Subareas

To predict taxi demand and destination, we first divide the entire city into small areas. Same as our previous work [15], we are using the Geohash library [7] which can divide the whole city into subareas at arbitrary precision. Each pair of $[latitude, longitude]$ can be encoded into a geohash string where neighboring areas share the same string prefix. The advantage of this encoding method is that when we sort all the encoded strings, the index of neighboring areas stay together. Later we will show that this is very helpful in our destination prediction. In this paper, we divide the entire city into around 1000 small areas with geohash precision 6 ($eachcell \leq 1.2km \times 0.61km$).

B. Predicting future demands

We propose a taxi demand predictor that can predict taxi demand in any target area of the city in the next hours, days and weeks. For any given area, the past taxi demand can be treated as a sequence. Fig. 1 shows taxi demand at two different places in New York City over a period as long as a week. We observe that in a specific area of the city, the historical taxi demand shows a predictable sequential pattern every week. Motivated by observing this pattern, we design a sequence learning model that learns the demand patterns from the sequential data.

We first divide a day into discretized time-steps $\{t_0, t_1, ...t_{max}\}$ where t_i is the ith time-step of a day. Note that the time-step length is a hyper-parameter. Second, for each area, we count the number of taxi requests in each time-step. Fig. 2 shows the input and output data structures in one time-step. For time-step t_i, the input data x_i consists of two parts: $[f_i, e_i]$. f_i represents potential affecting factors such as date, day of the week, time-step in the day and weather. We use the official historical weather information of NYC from National Oceanic and Atmospheric Administration (NOAA). e_i represents the number of pickups in each area and its length is the number of small areas in the entire city.

To build a sequence learning model, we use one of the best Recurrent Neural Networks (RNNs): Long Short Term Memory (LSTM). As shown in Fig. 3, the input data to the model at time-step t_i is $[x_{i-seq}, ...x_{i-1}, x_i]$. The meaning of seq here is that we are using previous seq time-steps data to predict next time-step data. seq is a hyper-parameter that

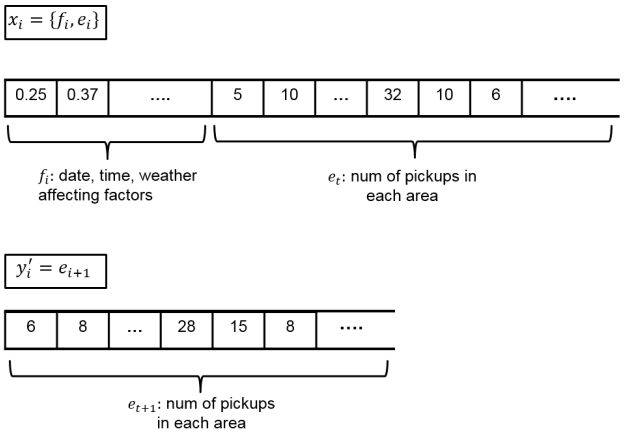

Fig. 2. Input and output data structure for demand prediction.

is set large enough to enable the network to learn long-term dependencies. Given the input data at t_i, the network predicts the output y'_i, the number of requests in each area at the next time-step. To train the network using stochastic gradient descent, we try to minimize the mean squared difference between the predicted y'_i and the ground-truth demand y_i.

With the trained model, we can predict taxi demands for all areas in future. Fig. 4 shows a density map of real and predicted taxi demands over the entire city. As we can see that red areas show high demand while yellow areas show lower demand. The figure illustrates that the difference between the predicted and the real demand is very small.

C. Predicting destination distributions

Destination prediction is much harder than the demand prediction because it contains much more uncertainty. Some works [11] are using a small window of GPS traces to predict the destination of each trip. We consider a different scenario in which we predict possible destinations for future taxi trips without relying on their GPS traces. We solve it as a distribution prediction problem from a statistical perspective.

Consider a trip request from one of the areas, its destination can be any area in the city. Fig. 5 (a,b) show two examples

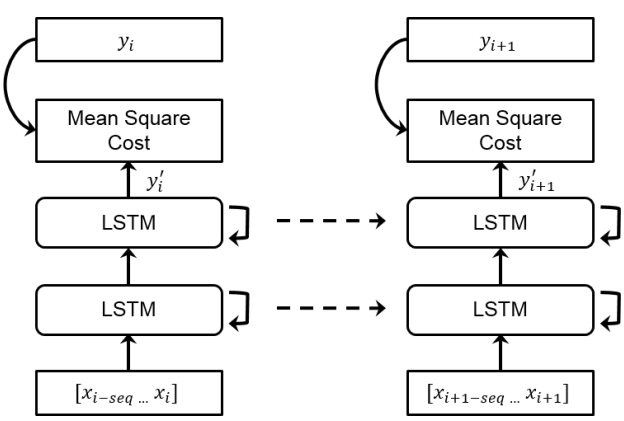

Fig. 3. Demand sequence learning model.

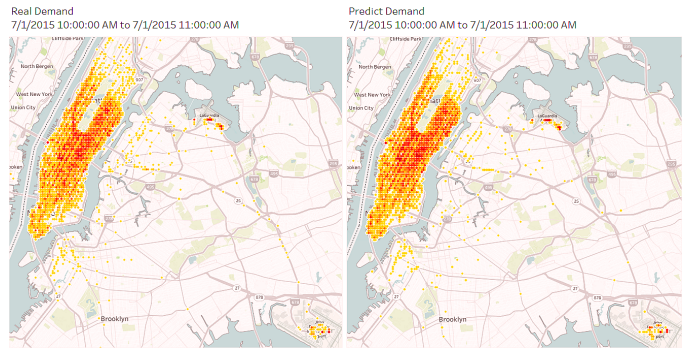

Fig. 4. The density map of real demand and the predicted demand. The figure illustrates that the difference between the prediction and the real value is very small.

of destination distributions start from two different areas. The time period is 30 $mins$. We cluster neighboring areas sorted by Geohash into different bins for better visualization. The horizontal axis represents the area index while the vertical axis represents the number of dropoffs in the area.

Fig. 5 (a,b) show interesting distribution patterns. The data distribution is multi-modal, i.e. for each input there are multiple possible outputs. This motivates us to use Mixture Density Networks (MDNs), developed by Christopher Bishop [8] that is designed to model real-valued multi-modal distributions. The idea behind MDN is to use the output of a neural network to predict the parameters of a mixture Gaussian kernels. Note that Gaussian kernels have a different set of parameters in each area. With the learned parameters, we can sample the destination predictions for each taxi trip.

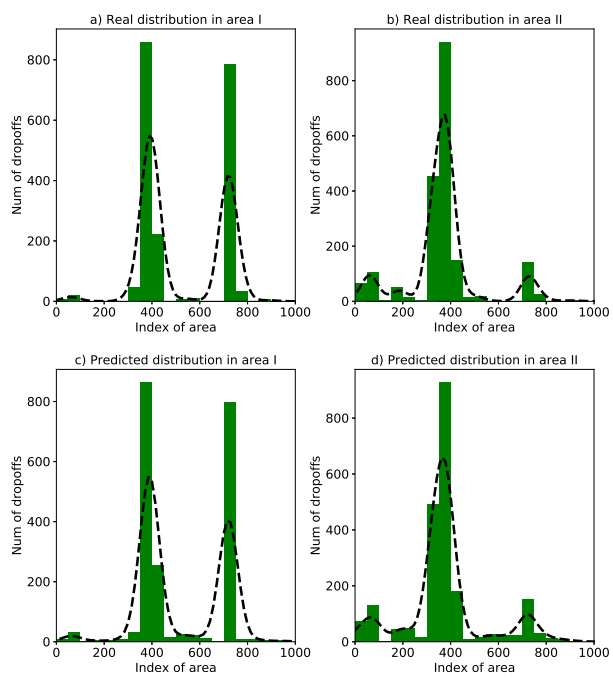

Fig. 5. Drop-off distributions of two different start areas.

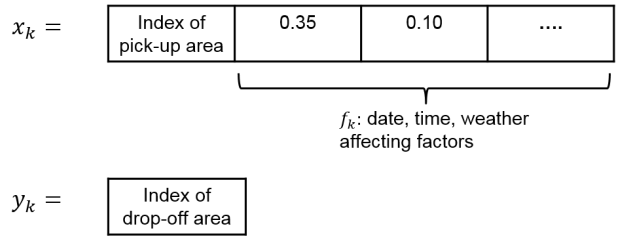

Fig. 6. Input and output data structure for destination prediction.

To build a distribution learning model, we first extract each trip information from historical taxi dataset, which contains time-stamp, pickup location, and dropoff location. Then we encode it into a data structure shown in Fig. 6. One of the advantages of using Geohash library is that the encoded neighboring areas will stay close when we sort them. Each trip is converted into a pair of data point $[x_k, y_k]$, where k represents the trip index in the datset. x_k consists of the pickup area and the corresponding factors such as time step since the beginning of the day, day of the week and weather. y_k represents the destination area of this trip.

Fig. 7 shows the distribution learning model. The goal is to learn the parameters of mixture of Gaussians for each area. As shown in Fig. 7, we feed x_k to a fully connected neural network. The expected output is a vector of distribution parameters with length $3 \times M$. M is the number of Gaussian kernels, which is a hyper-parameter. Each Gaussian kernel consist of 3 variables $[\omega, \mu, \sigma]$. ω is the mixing coefficient, μ is the mean and σ is the standard deviation.

A suitable loss function is to minimize the logarithm of the likelihood of the distribution of the training data:

$$Cost = -ln\left\{\sum_{m=1}^{M} w_m(x)\phi_m\{y, \mu_m(x), \sigma_m(x)\}\right\} \quad (1)$$

where $\phi_m\{y, \mu_m(x), \sigma_m(x)\}$ is the mth Gaussian kernel. It

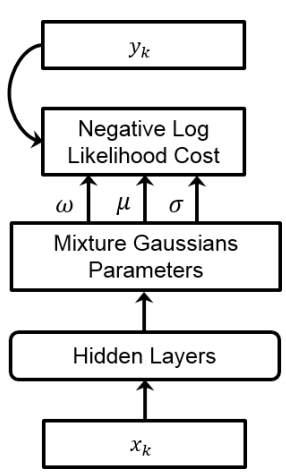

Fig. 7. Destination distribution learning model.

can be represented as:

$$\phi_m\{y, \mu, \sigma\} = \frac{1}{2\pi\sigma_m(x)} \exp\left\{-\frac{|y - \mu_m(x)|^2}{2\sigma_m^2(x)}\right\} \quad (2)$$

For each pair $[x_k, y_k]$ in the training dataset, we can calculate the cost based on the predicted distribution versus the actual value, and then attempt to minimize the sum of all the costs combined. Fig. 5 (c,d) shows the corresponding predicted distributions. As we can see, there are some differences compared to the real distributions in Fig. 5 (a,b) but overall the differences are small.

IV. Dispatching Model

In this section, we discuss how we do the taxi dispatch using the demand and destination predictive models we trained in the previous section.

Similar to the current mobile app based taxi services, we consider a scenario where passengers can send real time taxi requests to the system with start and destination locations. Goals of our dispatch system are given as follows:

- Serve all the taxi requests.
- Minimize idle driving distances of taxis.
- Minimize passengers average waiting time (time between sending request and pickup).

As we discussed in section III, we divide the whole city into a list of small areas all_areas. We also divide a whole day into time-steps $\{t_0, t_1, ...t_{max}\}$ such that t_i means the ith time-step of a day. Besides, we generate a distance matrix between each pair of areas $(a_i, a_j), i, j \in N$, where $N = len(all_areas)$ is the total number of areas, a_i is the pickup area, and a_j is the dropoff area. The trip distance is shown by d_{ij} and is the average distance for all trips between a_i and a_j from historical taxi data. For pairs of areas (a_i, a_j) that there are no recorded trips between them, we use a bidirectional search to find intermediate areas between them and then the shortest path $argmin(d_{ip} + d_{pq} + d_{qj})$ is returned as the distance d_{ij}. Here p and q are intermediate areas indexes. Algorithm 1 shows the process of generating distance matrix.

Given the distance matrix, for each taxi request received by the system, the corresponding travel time can be estimated. Based on this information, we build the dispatch system. For any area a, we represent the taxi demand at time-step t_i as D_i^a. Similarly, the available number of taxis is represented as P_i^a which consists of 2 parts:

$$P_i^a = idle_i^a + arrival_i^a \quad (3)$$

where $idle_i^a$ represents original idle taxis in area a at time step t_i, $arrival_i^a$ represents arriving available taxis to area a at time step t_i.

At time-step t_i, we assign available taxis to requests in two steps. First, for each area, we sort taxi requests D_i^a by receiving time, then a greedy assignment with available taxis P_i^a is conducted. There is a possibility that some requests can not be assigned due to limited number of available taxis in the same time-step. Since we sort all the taxi requests by receiving

Algorithm 1: Distance matrix generation

```
all_areas, list of all areas
N = len(all_areas)
all_trips, from dataset, grouped by trip start area
Distance matrix Dis = [N][N]
for (a_i, a_j) in all_areas do
    trips = gettrips(a_i, a_j, all_trips)
    Dis[a_i][a_j] = mean(trips.distance)
end
/* inferring distances
for (a_i, a_j) in all_areas do
    if Dis[a_i][a_j] == NULL then
        /* no recorded trips */
        candidatepaths =
        bidirectionsearch(a_i, a_j, Dis)
        Dis[a_i][a_j] = min(candidatepaths)
    end
end
return Dis
```

Algorithm 2: Taxi dispatching

```
1  all_areas, list of all areas
2  Distance matrix Dis
3  Future rebalance time steps lookahead
4  Demand prediction model M0
5  Destination prediction model M1
6  for t_i ∈ [begin, end] do
7      P_i = idle_i + arrival_i
8      /* greedy assign taxis to requests
9      remains_i = sort_assign(D_i, P_i)
10     update(idle, arrival)
11     s_i = rebalance(t_i, remains_i, idle, arrival)
12     if s_i ≠ NULL then
13         assign(s_i, P_i)
14         /* rebalance for future
15     end
16 end
17 Function rebalance(t, remains, idle, arrival):
18     for t_j ∈ (t, t + lookahead] do
19         D'_j ← M0.predict(t_j) /* demand
20         Des ← M1.predict(t_j, D'_j) /* destination
21         update(arrival) ← (Des, Dis)
22         P'_j = idle_j + arrival_j
23         D'_j = D'_j + remains
24         remains = sort_assign(D'_j, P'_j)
25         update(idle, arrival)
26     end
27     solution = MIP(remains, P')
28     return solution
29 end
```

time, a high priority is given to earlier requests and they will be fulfilled in the next time-step.

Second, we rebalance the taxis according to our prediction of future demand within a number of $lookahead$ time steps. For a future time step t_j at area a, where $t_j \in (t_i, t_{i+lookahead}]$, we use the predicted taxi demand ${D'_j}^a$ and the available taxis ${P'_j}^a$ to model the taxi assignment process. After this, a taxi rebalance process is conducted to optimize the supply-demand curves in each area over the entire city.

With the predicted taxi demand and destination, we optimize the taxi assignment and rebalance by solving a Mixed Integer Programming (MIP). The objective of the MIP is to minimize the total idle driving distances while serving all the coming requests. The details of dispatch process is shown in Algorithm 2.

In Algorithm 2, the dispatch process at each time step is shown in lines 6-16. For each area, we first assign available taxis to requests in a greedy fashion. Then, the unassigned requests together with updated available taxis are passed to the $rebalance$ function. During the rebalance process, the system first uses the predicted future demand and destination to model the greedy assignment according to future predictions. After that, all the remaining unassigned requests and the available taxis in the system are the targets to be matched. We optimize the matching by solving a MIP with the goal of minimizing the total idle driving distances. Finally the MIP solution is returned and a real taxis rebalance is conducted to serve the future demand.

V. Experimental Study

In this section, we evaluate our taxi demand and destination prediction models as well as the performance of the dispatch system.

A. Experimental setup

We validate the performance of the proposed network model with the New York City taxi trip dataset [9]. There are two kinds of taxi cabs in NYC: the yellow cabs, which operate mostly in Manhattan, and the green cabs, which operate mostly in the suburbs. The dataset contains daily recorded taxi trips executed by more than 15000 taxis for the whole city. The total number of taxi trips varies everyday. We list one week data in 2016 as an example, from Monday to Sunday: $[374305, 395678, 408184, 432087, 453192, 480818, 418237]$. We train our taxi demand and destination prediction models with one year taxi data in 2015 and validate the prediction models as well as the dispatch system with taxi data in February 2016.

For the taxi demand patterns learning model, we use a LSTM-based recurrent neural network. We discretize the requests into time-steps of 10 $mins$ for each area in 2015. The training data shape is $(365 * 144, 144, 997 + 10)$ in which $365 * 144$ is the total number of time-steps, 144 in the second dimension is the sequence length (one day or 24×6), 997 in the last dimension is number of areas in the whole city and 10 is the number of affecting factors including date, time step

of the day, day of the week and weather information.

For the destination distribution learning model, we use a feed-forward neural network. We use a time step length 30 $mins$ to learn the destination distribution for each area since it helps to make the data more predictable compared to 10 $mins$. In the dispatch system, we use the same mixture Gaussian parameters every 30 $mins$ at one area.

For the dispatch system, we initialize the taxi distribution based on a sum of historical requests in each area. We use the generated distance matrix and a fixed taxi speed of 64 km/h to estimate the traveling time between pairs of areas in the city. Table I includes the list of parameters in the experiments.

TABLE I
EXPERIMENTAL PARAMETERS

Area/grid size	$\leq 1.2km \times 0.61km$
Dispatch system time step unit	1 min
Taxi speed	64 kph (40 mph)
Number of areas	997
Number of hidden layers	2
Number of Gaussian kernels	5

B. Performance metrics and baselines

1) Demand prediction: To systematically examine the performance of our prediction approach, we include results with the widely used prediction error metric called symmetric Mean Absolute Percentage Error (sMAPE) [6]. From the statistical perspective, sMAPE describes a percentile prediction error and can be defined as follows:

$$sMAPE_i = \frac{1}{N}\sum_{n=1}^{N}\frac{|Y_{n,i} - \widehat{Y}_{n,i}|}{Y_{n,i} + \widehat{Y}_{n,i} + c} \tag{4}$$

Herein i is the time step t_i and N is the total number of areas in the city. $Y_{n,i}$ represents the real taxi demand in area a_n at time-step t_i while $\widehat{Y}_{n,i}$ is the predicted taxi demand. The constant c in Eq. 4 is a small number ($c = 1$ in this application) to avoid division by zero when both $Y_{n,i}$ and $\widehat{Y}_{n,i}$ are 0.

2) Destination prediction: For the destination prediction, we show the classification accuracy by using the number of correct predictions divided by the total number of requests at time step t_i.

$$Accuracy_i = \frac{Corrects_i}{Corrects_i + Incorrects_i} \tag{5}$$

3) Dispatch system: for our dispatch system, we show the performance in terms of three metrics:

- Passengers' average waiting time.
- Taxis' average idle driving distances.
- Dispatch algorithm time complexity (Computational time for solving the MIP).

C. Performance results

First, we report demand prediction error sMAPE and destination prediction accuracy over the entire city (all prediction areas). Second, we report metrics that characterize the dispatch system.

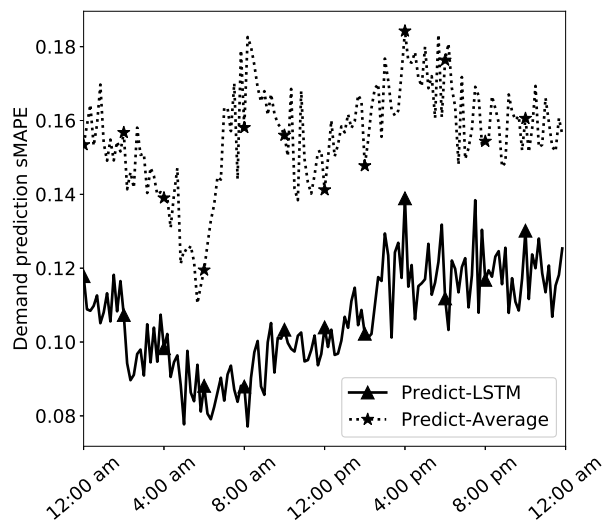

Fig. 8. Demand prediction performance according to sMAPE.

1) Prediction performance: We show the prediction performance in terms of the metrics formulated in Eq. 4 and Eq. 5. As a comparison, we also show the performance of methods based on the average of historical data. For instance, for each area, if it is 10:00 am on Monday, the predicted demand would be the average of demands there at 10:00 am in the past 5 Mondays. The corresponding destinations would be the normalized average of all destinations (starting from same area) at 10:00 am of past 5 Mondays.

For the demand prediction, we respectively use *Predict-LSTM, Predict-Average* to represent our LSTM-based approach and the historical data average-based approach. For the destination prediction, we respectively use *Predict-GM, Predict-Average* to represent our Gaussian mixture-based approach and the normalized average-based approach.

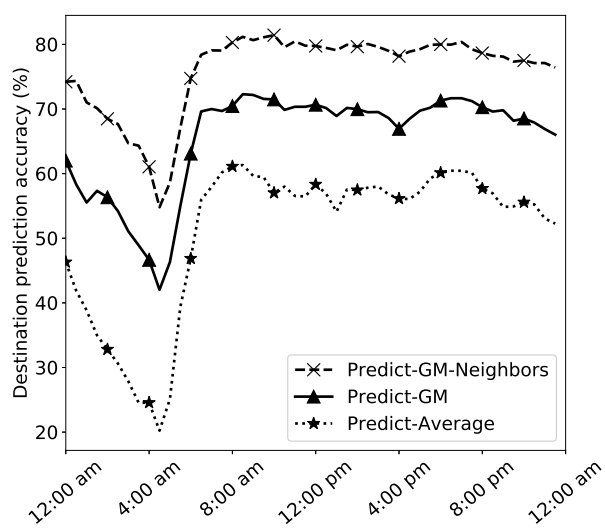

Fig. 9. Destination prediction precision.

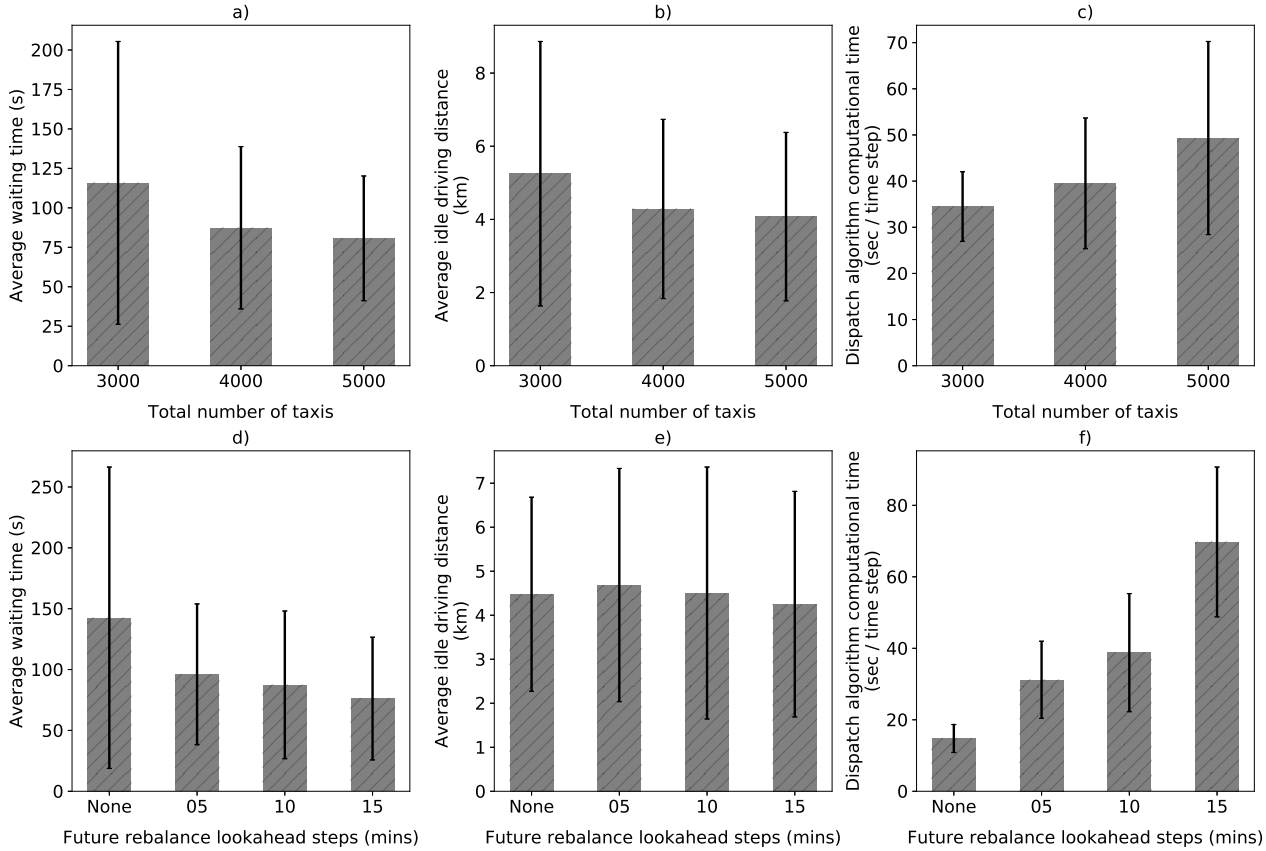

Fig. 10. Performance on passengers average waiting time, taxi average idle driving distances and the time complexity for finding optimal solution of taxi assignment. For figures a-c, 10 time steps $lookahead$ parameter is used. For figures d-f, 4000 total number of taxis is used.

Fig. 8 shows the prediction sMAPE over the entire city. Similar to the results in [6], the LSTM-based model always has a better prediction performance compared to the model based on historical data average. Besides, note that during busy hours around 8:00 am in the morning, the LSTM-based model still performs very well while the data average based model becomes much worse. This is because the LSTM-based model takes into account long term dependencies of past information while predicting the future.

Fig. 9 reports the destination prediction performance. It can be seen that Gaussian mixture model performs better than the normalized average-based method. Besides, since our prediction is sampled from a continuous distribution, if we accept neighboring areas as correct predictions, we achieve better results as shown in Fig. 9 (Prediction-GM-Neighbors). Note that predicting neighboring areas as destination is still acceptable and useful for our taxi dispatch application since the taxi is very close to the target area. On the other hand, considering Fig. 8 and Fig. 9 we observe that interestingly both of them have a small value at around 4:00 am. This however means that demand prediction achieves the best performance (lowest error) while destination prediction hits its worst accuracy during that time period. The reason could be that the total number of requests is low and mainly from some hot areas such as airports and bars, but the corresponding destinations can be anywhere around the town and are not that predictable.

2) Performance of Dispatch System: We evaluate the performance of our dispatch system from two perspectives. The first one is the performance metrics based on different total number of taxis in the system and the second one is the performance metrics based on different future rebalance $lookahead$ time steps.

Fig. 10 (a-c) show the error bars with standard deviation for average waiting time of each passenger, average idle driving distances of taxis, and the computational time of solving the MIP based on different total number of taxis in the system. As we can see, both the average waiting time and the average idle driving distances gradually decrease as there are more taxis running in the system. Large standard deviation is shown when total number of taxis is 3000. On the other hand, the goal of the MIP is to find the rebalance solution while minimizing total idle driving distances. The time complexity increases as the total number of taxis grows.

Fig. 10 (d-f) show three performance metrics on different future rebalance look ahead time steps. As we can see, by increasing the $lookahead$ time steps, shorter average waiting time for passengers can be obtained. Large standard deviation

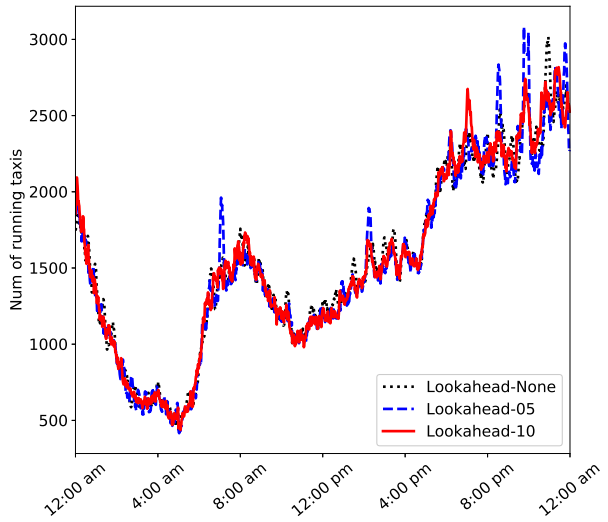

Fig. 11. Number of real-time running taxis in the system throughout a day. The total number of taxis in the system is 4000 for each of the method.

can be seen if there is no future rebalance ($None$) at all. The reason is that some requests are assigned to taxis in far areas due to no taxi availability nearby. A slightly increase in the average of taxi idle driving distances is shown when introducing future rebalance mechanism. It is reasonable and note that it also gradually decreases to a level less than the non-rebalance method ($None$). The reason could be that the result of the $lookahead$ rebalance avoids most long distance pickups. The computational time on different $lookahead$ time steps is shown in Fig. 10-f. Compared to Fig. 10-c, future rebalance is more time consuming than just increasing the total number of taxis. The reason is that in future rebalance, more taxis need to be assigned to other areas. Due to the limited computational power, the maximum $lookahead$ number in our experiment is 15 time steps. A better result could be obtained if we use a longer $lookahead$ time steps. But note that as the $lookahead$ grows, the prediction performance on future taxi demand also decreases.

Lastly, we show the real-time number of running taxis in the system throughout a day in Fig. 11. We do not see a big difference among them except a slight increase in busy hours when introducing future rebalance. The reason could be that the number of running taxis in the system highly relies on the real-time demand while only a small number of taxis are involved in the future rebalance.

VI. Conclusion

In this paper, we first propose two learning models to capture the patterns of historical taxi demand and destination distributions in each area over a city. The trained models can make real-time prediction of taxi demands and the destinations for the whole city. We also build a dispatch system in which the predicted demands and destinations are used for the taxis reallocation towards the future supply-demand balance in the city. The optimal taxi assignment and reallocation strategy is obtained by solving a mixed-integer program (MIP). We validate our dispatch system with taxi trip data of 2016 in New York City. Experimental results show that the proposed dispatch system can decrease the average waiting time of the passengers and the average idle driving distances of the taxis.

We believe that our approach has a great potential to optimize the distribution of taxis throughout the city so that the number of taxis required is minimized. Finally, analyzing passenger behaviors can result in finding some insights that lead to making the transportation more efficient. For instance, it might help to develop ride-sharing strategies or using some shuttles in high demand routes. A well-performing demand and destination prediction model is the key step towards reaching an integrated and efficient transportation system.

References

[1] X. Chen, F. Miao, G. J. Pappas, and V. Preciado, "Hierarchical data-driven vehicle dispatch and ride-sharing," in *Proc. of IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec 2017, pp. 4458–4463.

[2] K. Zhang, Z. Feng, S. Chen, K. Huang, and G. Wang, "A framework for passengers demand prediction and recommendation," in *Proc. of IEEE SCC'16*, June 2016, pp. 340–347.

[3] K. Zhao, D. Khryashchev, J. Freire, C. Silva, and H. Vo, "Predicting taxi demand at high spatial resolution: Approaching the limit of predictability," in *Proc. of IEEE BigData'16*, December 2016, pp. 833–842.

[4] D. Zhang, T. He, S. Lin, S. Munir, and J. A. Stankovic, "Taxi-passenger-demand modeling based on big data from a roving sensor network," *IEEE Transactions on Big Data*, vol. 3, no. 3, pp. 362–374, September 2017.

[5] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, September 2013.

[6] J. Xu, R. Rahmatizadeh, L. Bölöni, and D. Turgut, "Real-time prediction of taxi demand using recurrent neural networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 8, pp. 2572–2581, August 2018.

[7] G. Niemeyer. (2008) Tips & tricks about geohash. [Online]. Available: http://geohash.org/site/tips.html

[8] C. M. Bishop, "Mixture density networks," Aston University, Tech. Rep., 1994.

[9] NYC Taxi Limousine Commission. Taxi and limousine commission (tlc) trip record data. [Online]. Available: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

[10] F. Miao, S. Han, S. Lin, J. A. Stankovic, D. Zhang, S. Munir, H. Huang, T. He, and G. J. Pappas, "Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 463–478, April 2016.

[11] A. de Brébisson, É. Simon, A. Auvolat, P. Vincent, and Y. Bengio, "Artificial neural networks applied to taxi destination prediction," *CoRR*, vol. abs/1508.00021, 2015.

[12] J. Alonso-Mora, A. Wallar, and D. Rus, "Predictive routing for autonomous mobility-on-demand systems with ride-sharing," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, September 2017, pp. 3583–3590.

[13] J. Lin, S. Sasidharan, S. Ma, and O. Wolfson, "A model of multimodal ridesharing and its analysis," in *Proc. of IEEE International Conference on Mobile Data Management (MDM)*, June 2016, pp. 164–173.

[14] H. Zheng and J. Wu, "Online to offline business: Urban taxi dispatching with passenger-driver matching stability," in *Proc. of IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 816–825.

[15] J. Xu, R. Rahmatizadeh, L. Bölöni, and D. Turgut, "A sequence learning model with recurrent neural networks for taxi demand prediction," in *Proc. of IEEE LCN*, October 2017, pp. 261–268.