Transportation Research Part C 117 (2020) 102678

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Efficient proactive vehicle relocation for on-demand mobility service with recurrent neural networks

Zengxiang Lei, Xinwu Qian, Satish V. Ukkusuri*

Lyles School of Civil Engineering, Purdue University, West Lafayette, IN, USA

ARTICLE INFO

Keywords:
Proactive relocation
Vehicle dispatching
Optimal matching
Stable matching
Deep neural network

ABSTRACT

One major challenge for on-demand mobility service (OMS) providers is to seamlessly match empty vehicles with trip requests so that the total vacant mileage is minimized. In this study, we develop an innovative data-driven approach for devising efficient vehicle relocation policy for OMS that (1) proactively relocates vehicles before the demand is observed and (2) reduces the inequality among drivers' income so that the proactive relocation policy is fair and is likely to be followed by drivers. Our approach represents the fusion of optimization and machine learning methods, which comprises three steps: First, we formulate the optimal proactive relocation as an optimal/stable matching problems and solve for global optimal solutions based on historical data. Second, the optimal solutions are then grouped and fed to train the deep learning models which consist of fully connected layers and long short-term memory networks. Low rank approximation is introduced to reduce the model complexity and improve the training performances. Finally, we use the trained model to predict the relocation policy which can be implemented in real time. We conduct comprehensive numerical experiments and sensitivity analyses to demonstrate the performances of the proposed method using New York City taxi data. The results suggest that our method will reduce empty mileage per trip by 54–70% under the optimal matching strategy, and a 25–32% reduction can also be achieved by following the stable matching strategy. We also validate that the predicted relocation policies are robust in the presence of uncertain passenger demand level and passenger trip-requesting behavior.

1. Introduction

Enabled by pervasive computing and crowd-sourcing, urban on-demand mobility services (OMS) have expanded rapidly in recent years. In 2016, OMS accounted for 15% of all intra-San Francisco trips, which was approximately 11 times higher than that of taxi trips (Castiglione et al., 2016). As for New York City (NYC), the daily OMS trips doubled from 2016 to 2018, which made up nearly 30% of all traffic in NYC Taxi (2019). On the one hand, large-scale OMS utilize massive real-time and historical information from passengers and drivers for vehicle dispatching and ride-sharing, which provides new opportunities to understand ride demand and improve system-wise coordination of vehicles. On the other hand, the fast growing OMS bring excessive empty mileage and disruption to traffic flows (Erhardt et al., 2019), which eventually generate negative externalities such as traffic congestion and emissions. To address this issue, in light of the circumstance that real-time information and control is viable, it is important to devise a global recommendation/control scheme to enhance the efficiency and sustainability of large-scale OMS.

Current operation practices of OMS are mainly reactive (Lyft, 2019; Uber, n.d.), which dispatch vehicles only based on observed

* Corresponding author. Tel.: +1 7654942296.
E-mail addresses: lei67@purdue.edu (Z. Lei), qian39@purdue.edu (X. Qian), sukkusur@purdue.edu (S.V. Ukkusuri).

https://doi.org/10.1016/j.trc.2020.102678
Received 28 November 2019; Received in revised form 17 April 2020; Accepted 21 April 2020
Available online 07 July 2020
0968-090X/ © 2020 Elsevier Ltd. All rights reserved.

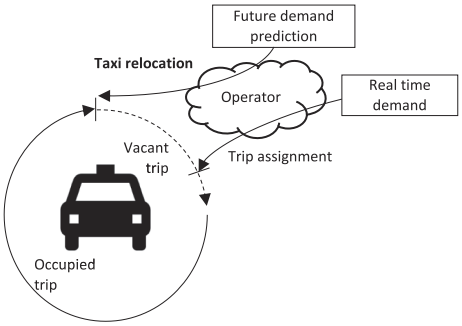


Fig. 1. Illustration of proactive vehicle relocation. After a taxi/ride-hailing vehicle drop off a passenger and becomes available, the operator will assign it to the relocation destination and the vehicle will be matched to the closest request near the relocation destination.

demand. For example, a greedy algorithm matches each request, in time sequence, with the nearest driver. Such method may work well when demand and supply are spatiotemporally aligned, nevertheless, real world observations suggest that they are highly imbalanced leading to inefficiencies (Qian et al., 2015). The inefficiency comes from the late and sub-optimal relocation decisions for rebalancing the empty vehicles of the entire system. Before a request is observed, reactive approaches rely heavily on human drivers to fulfill the rebalancing task, but such decisions are carried out based on their experience and often lead to sub-optimal relocation (Zhan et al., 2016; Zhu and Prabhakar, 2017). A more efficient way is to make the relocation decisions by leveraging the system-level information of passengers and vehicles. This motivates us to study the proactive approaches that reduce the empty mileage by properly dispatching empty vehicles before the request is realized.

Instead of waiting for more information before making dispatch decisions, proactive relocation takes effect whenever a vehicle is empty. The general mechanism of proactive relocation is shown in Fig. 1. When a vehicle becomes available, the OMS operator will assign/recommend it to relocate to the next location based on the prediction of the future demand, and the vehicle will then be assigned to the nearest request at the relocation destination. This process sends vehicles to potential places that close to new requests, thus reducing the passenger's waiting time. Furthermore, as compared with reactive approaches, proactive relocation takes future demand into account, which leads to a better matching result for the whole group of requests (Miao et al., 2016). Proactive relocation is also of great significance to the efficient operation of OMS with autonomous vehicles since in such system there is no human driver and the operator will need to derive the optimal relocation policy for the entire fleet in real time (Winter et al., 2017; Iglesias et al., 2018). The challenges for devising a well-optimized proactive relocation strategy, however, come from three aspects: (1) Prediction error and uncertainty of future demand can lead to inferior dispatching policy (Miao et al., 2017); (2) The computational cost may grow rapidly as more time steps are taken into account, which makes it intractable to solve this problem in real time; (3) Drivers may not follow the relocation decisions that decrease their income.

To address the above challenges, an innovative data-driven model is proposed to derive the zonal level proactive relocation strategy. In this model, the relocation policy is represented by the probabilities of relocating a vehicle to other zones from its current location. We first generate optimized relocation policy based on the historical trip profiles. Then the deep neural network model (DNN) that consists of fully connected layers (FC) and long short-term memory networks (LSTM) is trained to capture the spatial and temporal features of the historical relocation policies and predict the optimal relocation policies in future. Our solution addresses the first challenge by incorporating low rank approximation in DNN to improve the training efficiency and accuracy. Second, by directly predicting the relocation policy using DNN, our method requires less computational time, thus addresses the challenge (2) above. To overcome the last challenge, besides optimal strategy that minimizes the total empty mileage, here we consider the stable matching strategy that reduces the income inequality among drivers. The innovation of our study lies in that the solution framework represents the fusion of optimization methods and machine learning models, and the relocation policies derived under our framework are suitable for both OMS with human drivers as well as with autonomous vehicles.

We use New York City's yellow taxi data to validate the effectiveness of the relocation strategies obtained using our method, and quantify the robustness of the relocation performances under different demand levels and maximum waiting time. We show that both optimal matching and stable matching strategies can reduce the empty mileage, furthermore, the stable matching strategy also helps reduce the income inequality, which is the discrepancy of income among drivers.

The rest of the paper is organized as follows: Section 2 is a brief review of literature on vehicle relocation problems, mainly in the scope of OMS market. Section 3 presents our method, including optimal/stable matching, DNN for predicting relocation policy and a passenger-vehicle matching simulation for policy evaluation. In Section 4, we present the results for the numerical experiments, using 224 weekdays' rides extracted from NYC yellow taxi data. A sensitivity analysis is also presented to demonstrate the robustness of the proposed approach. Finally, we conclude our work and give future directions in Section 5.

2. Literature

This section provides a review of the literature in devising proactive relocation policies for OMS systems. Many of these studies focus on the relocation problems for one-way carsharing systems (Smith et al., 2013; Kek et al., 2009; Weikl and Bogenberger, 2013;

Huang, 2018) and autonomous on-demand mobility systems (AOMS) (Pavone et al., 2012; Zhang and Pavone, 2016; Chuah et al., 2018; Zhang et al., 2016). Some studies are related to traditional taxis and e-hailing services (Sayarshad and Oliver Gao, 2018; Miao et al., 2016; Miao et al., 2017; Iglesias et al., 2018), in which the proactive relocation is also called as ”non-myopic dispatching” and ”rebalancing”. Despite the operational differences between different OMS systems, there are two streams of literature in terms of the methodology. The first method corresponds to the steady-state approach, where it is assumed that the system reaches a steady-state that is stable over time. Then it devises the relocation policy based on this steady-state. The second method belongs to the category of real-time optimization, where the decision maker identifies the optimal matching strategy by solving the optimization problem with all information revealed.

2.1. Steady-state method

Pavone et al. (2012) utilized a fluid model to describe the passenger arrivals and vehicle departures, then based on this model they derived a station-wise rebalancing policy to make all areas reach an equilibrium in which there are excess vehicles and no waiting customers. Smith et al. (2013) extended this model by adding the flows of the drivers who perform the relocation. Zhang and Pavone (2016) presented a queueing-theoretical model for autonomous OMS system and then solved the relocation policy that minimizes the number of rebalancing vehicles using linear programming. Sayarshad and Oliver Gao (2018) modeled the non-cooperative OMS as a multi-server queue system, then developed a dynamic pricing scheme to reduce the average travel distance for serving one passenger. Chuah et al. (2018) introduced control of passenger queues and solved the optimal rebalancing policy incorporating waiting time constraints in the queuing system.

2.2. Real-time optimization

Kek et al. (2009) formulated the vehicle relocation problem as a mixed integer programming problem (MILP) and heuristically converted the result to operating policies. Weikl and Bogenberger (2013) proposed a two-step relocation algorithm for a free-floating car-sharing system with the first step identifying periodically repetitive spatial–temporal demand patterns and the second step solving the relocation problem with the demand patterns. Miao et al. (2016) proposed a model predictive control (MPC) framework for solving taxi dispatching problem. Zhang et al. (2016) introduced MPC into the autonomous OMS system and solve the relocation actions with future demand and car arrivals. Huang (2018) extended the MINLP formulation with a behavior model for predicting future demand influenced by relocation schedules. There are recent works that adopted deep learning techniques in vehicle relocation. Iglesias et al. utilized LSTM (Iglesias et al., 2018) to predict the OD matrix between regions, then computed rebalancing strategies based on MPC. Wen et al. (2017) built a deep Q network to decide how to relocate vehicles between zones for each time step.

2.3. Contributions

These two streams represent two general methods to solve the vehicle relocation problems. The first characterizes the steady-state of the OMS system by a set of parameters, then solves the optimal strategy based on these parameters. The drawback of this stream lies in its unrealistic assumption about the existence of the steady-state as travel demand and traffic conditions are unstable and changing over time. The second method strictly formulates the optimization problem to solve the optimal relocation policy. By incorporating the prediction of future demand and vehicle arrivals, this method can provide non-myopic relocation instructions. Nevertheless, the limitation comes from the computational cost and demand uncertainty. For a large and complex OMS system, solving such optimization in real time can be computationally expensive when more time steps are taken into consideration and the demand fluctuates would also influence the quality of the relocation policy. Based on these insights, we summarize the contributions of this work as follows.

- Instead of solving optimization in real time, we propose a machine learning framework to directly predict optimized relocation policy based on the real-time demand.
- We suggest a way to use low rank approximation in the deep learning model to cut down the output dimensions, which greatly reduces the model complexity and improves the training accuracy.
- We consider two types of relocation strategies: the optimal strategy that minimizes the total empty mileage, and the stable matching strategy (Bai et al., 2014; Kümmel et al., 2016) that reduces the total empty mileage while balancing the drivers' income.
- We perform extensive numerical experiments and sensitivity analyses based on the real world data to show the effectiveness and the robustness of our approach.

3. Methodology

3.1. Overview

This study focus on devising an optimized zonal level relocation policy over discretized time steps. Each zone is a small area serving as the unit for demand aggregation and relocation execution. The objective is to reduce the empty mileage generated by OMS.

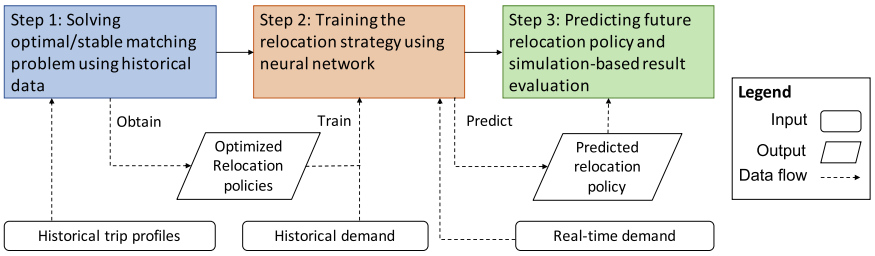


Fig. 2. Overall framework.

To achieve this goal, we consider two different strategies: the optimal strategy that minimizes the total empty mileage, and the stable matching strategy that reduces the empty mileage while balancing drivers' income. Given n zones, the relocation policy of time t, denoted as P^t, is a $n \times n$ matrix, in which each element p_{ij} means the possibility to relocate to zone j starting from zone i. Once a vehicle finishes its last trip at time step t, the operator would assign/recommend it with the relocation destination based on the corresponding transition probability in P^t.

The framework of our method is shown in Fig. 2. We first solve the relocation optimal/stable matching problems to obtain the optimized relocation policies based on the historical trip data. The results of the first step then serve as the training target for the neural network to learn the relationship between travel demand and corresponding relocation policies. In this step, to reduce the number of parameters in the fully connected layer of the neural network, we adopt low rank approximation (LRA) to project the relocation policies to lower dimensions. Finally, we evaluate our methods by a faithful simulation that matches vehicles and passengers based on the relocation policies.

3.2. Optimal matching

We introduce optimal matching based on the developed model in Zhan et al. (2016), which matches the available taxis and requests within one time step to minimize the total matching costs. For time step t, given the set of available taxi $\mathcal{V}_t$ and requests $\mathcal{R}_t$. The optimal matching is formulated as follows.

$$\begin{aligned} \min \quad & \sum_{i,j} m_{ij} w_{ij} \\ \text{s. t.} \quad & m_{ij}(d_{ij} - v_{max} t_{ij}) \leqslant 0, \quad v_i \in \mathcal{V}_t,\ r_j \in \mathcal{R}_t \\ & \sum_{j} m_{ij} = 1, \quad v_i \in \mathcal{V}_t \\ & m_{ij} \in \{0, 1\} \end{aligned} \tag{1}$$

where m_{ij} is the indicator variable of matching result; $m_{ij} = 1$ when taxi v_i is matched to request r_j, otherwise $m_{ij} = 0$. w_{ij} is the matching cost, d_{ij} is the distance between the v_i and r_j, t_{ij} is the available time for taxi v_i to pickup r_j. The first constraint ensures a matched taxi could reach its request within the available time. The second constraint states that one request can only be served once. Here we want to minimize the empty mileage, we set $w_{ij} = d_{ij}$.

Here we assume that each vehicle receives two consecutive requests with no less than 10 min. This allows us to discretize time into 10 min' intervals without considering a vehicle that departures and arrivals within the same time interval. In addition, we assume that for each time step, there exists a matching that all passengers can be served. This eliminates the case of keeping unserved requests into the next time step. Unmatched vehicles will be kept in the same zone waiting for the matching in the next time step. For each time step t, V_t is the combination of unmatched vehicle from the last time step and the arrived occupied vehicles within the time step t.

By adding dummy requests to make the number of requests equal the number of available vehicles, as shown in Fig. 3, the above

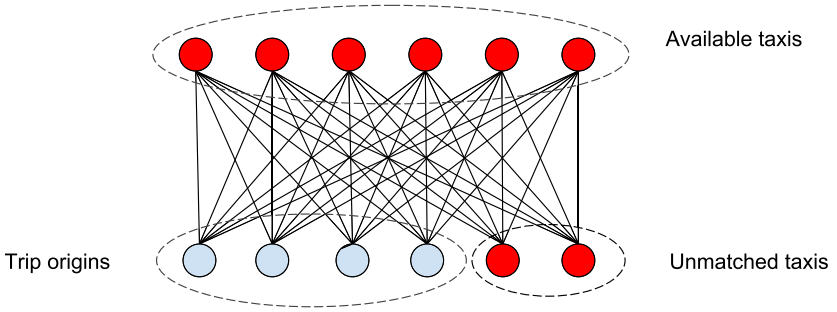


Fig. 3. Optimal matching as a minimum weight bipartite perfect matching problem in bipartite graph.

problem is equivalent to minimal weight perfect matching and can be solved by the Hungarian algorithm (Kuhn, 1955).

3.3. Stable matching

Consider searching passengers as a non-cooperative game between drivers, in which each driver has his own preference to passengers and each passenger is served by the first arrived drivers, a stable matching between drivers and passengers is a matching solution that no driver can be better off by changing their assigned passenger (Bai et al., 2014). In other words, stable matching creates an assignment that all drivers should satisfy. Note the stable matching is not unique, here we are interested in one solution obtained by the Gale-Shapley algorithm (Dubins and Freedman, 1981). This solution is known to be optimal for one side, which means every driver is matched with the closest request that they can secure (Bai et al., 2014; Kümmel et al., 2016).

Besides the two assumptions in Section 3.2, here we add two more assumptions to perform stable matching. First, we assume drivers only consider distance for deciding which request is better for the sake of simplicity. Second, since we do not have the request time (from NYC taxi records we only know the pickup time), we also assume the request time is X minutes earlier than the pickup time, where X is a random variable following a truncated normal distribution with non-negative value. We use this request time to deal with the condition that drivers can reach the passenger before the request is realized. To be clear, if multiple vehicles can reach the passenger before his/her request time, we assign this passenger to the driver with the smallest relocation mileage.

A practical issue arises as some requests may lie in remote areas (e.g. requests from the airports) and can not be reached by all vehicles, which may lead to unserved passengers in the result of the Gale-Shapley algorithm. By rural hospital theory (Gale and Sotomayor, 1985), those unserved requests in one stable matching, would still be unserved in all results of stable matching, thus violate the assumption that all trips are served. Therefore, we consider a variety of stable matching problem called stable matching with covering constraints (SMC) (Mnich and Schlotter, 2017). For a vehicle v_i and request r_j, we call (v_i, r_j) as one blocking pair when v_i prefer r_j and v_i can reach r_j earlier than his/her competitors. The objective of the SMC is to find a matching with minimum number of blocking pairs under the constraint that all requests being served.

This SMC is in NP-hard as shown in Mnich and Schlotter (2017), so we develop an approximation algorithm (Algorithm 1) to obtain reasonable results. Note when all passengers can be reached by all vehicles, this algorithm is equivalent to the Gale-Shapley algorithm because in this case two while loops can be merged into one; Otherwise, this algorithm ensures there is no blocking pair between unmatched vehicles and requests. The maximum number of the blocking pair is $(|\mathcal{R}|-1)|\mathcal{R}|$, which is much lower than upper bound $|\mathcal{V}||\mathcal{R}|$, where $|\mathcal{R}|$ is the number of requests, $|\mathcal{V}|$ is the number of empty vehicles and $|\mathcal{V}| \approx 4|\mathcal{R}|$ based on the observation from NYC yellow taxi data.

Algorithm 1. Approximation algorithm for stable matching between vehicles and requests

Input: Set of requests; Set of available vehicles
Initialize all requests and vehicles and their corresponding preference lists
Running Hopcroft-Karp algorithm (Hopcroft and Karp, 1973) (maximum bipartite matching) to obtain the first matching
while ∃ free driver v who still has a request to propose **do**
r = first request on v's list to whom v has not yet proposed
if v can pickup r earlier than r's current assignment v' **then**
v' becomes free
Assign v to r
end if
end while
while ∃ driver v is matched with r' but has better request to propose **do**
r = first request on v's list to whom v has not yet proposed
if v can reach r earlier than its assignment v', and v' can serve r' in time **then**
Assign v' to r'
Assign v to r
end if
end while

3.4. Low rank prediction with deep neural network

Based on the relocation trips calculated from the optimal/stable matching, we obtain the relocation policy P^t for each time step, which is a $n \times n$ matrix that represents the possibilities of relocating from one zone to others. A toy example that illustrates the calculation of policy matrix P^t can be found in Fig. 4.

We then use P^t as the training target of the neural network to develop a model that predicts a well-optimized relocation policy based on the information of real-time travel demand.

3.4.1. Low rank approximation

Note the number of elements in the relocation matrices are n^2. In this regard, the output dimension will be significantly increased with a growing number of zones in the study area. To overcome the high dimensional output space, we first introduce low rank approximation (LRA), a widely used method in data compression (Markovsky, 2008), for the purpose of dimensionality and noise

VehicleID	Departure time	Origin Zone	Destination Zone
1	8:06	1	2
2	8:06	3	1
3	8:08	3	2
4	8:09	1	1
5	8:09	2	2
6	8:09	1	2
2	8:12	2	1

Reolocation policy for 8:05-8:10:

$$P^{8:05-8:10} = \begin{bmatrix} 0.333 & 0.667 & 0 \\ 0 & 1 & 0 \\ 0.5 & 0.5 & 0 \end{bmatrix}$$

Fig. 4. A toy example for the calculation of relocation matrix during 8:05–8:10 (the number of zones $n = 3$).

reduction (Deerwester et al., 1990). The form of the LRA problem is as follows.

$$\begin{aligned} &\min \quad \|M - \widehat{M}\| \\ &\text{s. t.} \quad rank(\widehat{M}) \leqslant k \end{aligned}$$

where $M = [vec(P^0), vec(P^1), vec(P^2), ..., vec(P^T)]$ and *vec* reorganizes the matrix into a column vector. The shape of M is $n^2 \times T$. In this way, we have the columns of M capturing the temporal dynamics of relocation action P^t, and the rows capturing the spatial information of relocation action P^t.

The above problem can be solved exactly using singular value decomposition (SVD). Given $M = USV'$ is the SVD result, we have the optimal solution $\widehat{M} = U_k S_k V_k' = [U_k S_k \vec{v}_1', U_k S_k \vec{v}_2', ..., U_k S_k \vec{v}_T']$, where U_k and V_k are respectively the first k columns of U and V, and S_k is the $k \times k$ diagonal matrix with the elements be the first k singular values of matrix M, $\vec{v}_t$ is the t-th column of V_k. Instead of directly predicting P^t with n^2 elements, we can predict $S_k \vec{v}_t$, which only has k elements. Furthermore, by the definition of SVD, we have $U_k' U_k = I_{k \times k}$, so left multiplying U_k will not influence the $L2 - norm$, that is $\|S_k \vec{v}_t'\|_2 = \|U_k S_k \vec{v}_t'\|_2 \approx \|vec(P^t)\|_2$, so the mean squared error (MSE) of $S_k v_t'$ is approximately equal to the MSE of $vec(P^t)$.

3.4.2. Deep neural network

The purpose of the deep neural network is to explore the spatial and temporal dependencies of system status and the low rank approximation of future optimal relocation policy P^{t+1}. As shown in Fig. 5, the model input S^t is a vector of passenger demand between each pair of OD i, j for time interval t. The first fully connected layer FC1 is therefore responsible for understanding the spatial correlations among the trip ODs and detecting low level spatial features. Given n zones, the input and output size is n^2 and l_{FC1} respectively.

However, having a single slice of input S^t suffers the issue of partial observability. It lacks the proper 'velocity' information to tell if the OMS market is shifting from off-peak to peak-time period or is approaching the end of peak hours, where the relocation policy shall differ significantly under these circumstances. Consequently, we introduce the LSTM network (Hochreiter and Schmidhuber, 1997) to capture the temporal dependencies among the extracted spatial features from a collection of sequential observations $S^t, S^{t+1}, ..., S^{t+T}$. In particular, we leverage a single LSTM layer which takes the input directly from the output of the last fully connected layer for each observation of the sequential input, and we take the output of the last hidden state of the LSTM layer that is unrolled over last $T = 10$ time steps as the extracted spatio-temporal features of the input state (our OD matrix). The number of hidden units of the LSTM layer is set to l_{LSTM}. Finally, the last hidden state h^{t+T} is passed through a fully connected layer FC2 with $l_{LSTM} \times k$ connections to obtain the low rank approximation $S_k \vec{v}_{t+T+1}'$ of the future relocation policy P^{t+T+1}. The optimization objective is to minimize the MSE of the predicted $S_k \vec{v}_{t+T+1}$ and true $S_k \vec{v}_{t+T+1}$. The minimization is performed using ADAM (Kingma and Ba, 2014). Early stopping is introduced to prevent the overfitting issue of the neural network.

We then estimate P^{t+T+1} by $U_k S_k \vec{v}_{t+T+1}'$ and standardize it by filtering negative values and scaling each row of P^{t+T+1} from 0 to 1.

3.5. Passenger-vehicle matching simulation

The trained neural network model is then used to predict the relocation policy in the future. To evaluate the predicted policy, we develop a simulation program that matches drivers and trip requests in real time. Each driver, denoted as v, is represented by the tuple $(v_{id}, v_t, v_{lon}, v_{lat}, v_{zone})$, where v_{id} is the ID of v, v_t is the available time, v_{lon} and v_{lat} capture the geocoordinates, v_{zone} is the zone index of the location. We denote each request r as the tuple $(r_t, r_t^o, r_{lon}^o, r_{lat}^o, r_{zone}^o, r_t^d, r_{lon}^d, r_{lat}^d, r_{zone}^d, r_{cost})$, where r_t is the request time, r_t^o is the departure time, r_{lon}^o and r_{lat}^o are the geocoordinates of the departure location, and r_{zone}^o is the corresponding zone index, r_t^d is the arrival time, r_{lon}^d and r_{lat}^d are the geocoordinates of the arrival location, r_{zone}^d is the zone index of the arrival location and r_{cost} is the fare of r.

This simulation consists of two parts: A relocation system which (1) gives recommendations to drivers d when they finish the previous trip and (2) updates relocation policy based on the real time demand. A trip assignment system that assigns taxis to the pending requests. We maintain a set of passenger queue Z_i, taxi lists Q_i for all relocation zones i, and use relocation strategy as a black box f. Within each zone, the trip assignment follows the rule of batch matching, where available passengers and drivers are matched by solving the optimal matching Problem 1. Here we set the time window for batch matching to 30s. For inter-zone relocation, the empty distance is calculated as the combination of the relocation distance (distance from the last drop-off location to the relocation

zone's centroid) and the pickup distance (distance from the zone's centroid to the pickup location). For vehicles that are not relocated to different zones, the empty mileage is directly calculated as the distance between the passengers and vehicles. For unmatched rides, the passenger will wait for the next time step when the current time is earlier than his/her departure time. Otherwise, the passenger will be served following the first come first serve (FCFS) rules, where passengers are assigned in the order of the request time to the closest available vehicle, where we define "available vehicles" as the vehicles that can reach the passenger before the passenger's departure time. If there are still unserved passengers after the FCFS process, we consider it as a failed match and accumulate the mismatch count by 1. The complete simulation algorithm is summarized in Algorithm 2. As a final remark of the matching simulation, we note that the cost of batching matching will always be superior to that of the FCFS matching due to the cost for proactively relocating vehicles and the additional mileage for making inter-zonal trips. For example, if a vehicle is relocated from zone i to zone j but finally assigned to a request in zone k, where zone k is in the middle of zones i and j. Then the empty mileage is calculated as the summation of the distance for relocating to zone j first and then moving back to zone k. In this regard, the predicted proactive relocation is deemed to achieve a lower cost if the vehicles are relocated to the appropriate places.

Algorithm 2. Passenger-vehicle matching simulation

Input: System states for different time steps S^1, S^2, ...,S^T; Set of new requests in different time steps R^1, R^2, ...,R^T; Set of arrival vehicles in different time steps V^0, V^1, ...,V^T
Output: Matching result $\mathcal{M}$, Total vacant distance D, Total mismatch counts m
Initialize Q_i using V^0
for t from 1 to the maximal time step T **do**
Update relocation strategy $P^t = f(S^t)$
for vehicle v in V^t **do**
Choose destination i based on P^t and v_{zone}
Add v to Q_i^t
end for
for each passenger r in R^t **do**
Add r to $Z_{r^o_{zone}}$
end if
each zone Z_i
Solve optimal matching (1), obtain pairs of vehicle v^* and request r.
$D \leftarrow D + d_{r,v^*}$
$v_t^* \leftarrow r_t^d$, $v_{lon}^* \leftarrow r_{lon}^d$, $v_{lat}^* \leftarrow r_{lat}^d$, $v_{zone}^* \leftarrow r_{zone}^d$
Add v^* to $V^{v_t^*}$, delete r
end for
for each request r satisfying $r_t^o == t$ **do**
if A vehicle can pickup the passenger in time **then**
Assign closest vehicle v^*
$D \leftarrow D + d_{r,v^*}$
$v_t^* \leftarrow r_t^d$, $v_{lon}^* \leftarrow r_{lon}^d$, $v_{lat}^* \leftarrow r_{lat}^d$, $v_{zone}^* \leftarrow r_{zone}^d$
Add v^* to $V^{v_t^*}$, delete r
else
$m \leftarrow m + 1$
Delete r
end if
end for
end for

4. Experiment results

4.1. Experiment settings

We demonstrate the performance of our proactive relocation methods for both optimal and stable matching problems using 2013 New York City taxi trip data on weekdays (Donovan and Work, 2016). The reason for using 2013 data instead of more recent ones is because that the 2013 data provide encoded taxi ID for each trip, which allows us to track trip sequences of each taxicab so that we can obtain the baseline relocation performance and compare it with our results. Besides taxi ID, the data also contain timestamps and geolocations of trip pickups and drop-offs as well as trip fare. Due to the lack of fine-grained trajectory data, we estimate the distance of relocation trips based on the Manhattan distance. Data preprocessing is performed to exclude trip records during anomaly events (e.g. daily number of trips deviates the mean for over 1.5 times of interquartile), and we remove trip records with distances less than 200 m and the duration less than 1 min. This leaves us with 471,389 daily trip records for 224 weekdays. We focus on two study periods, 7:00–9:00 (AM period) and 18:00–20:00 (PM period), which correspond to the trip prime time where efficient relocation is expected. We consider drivers (taxi IDs) who appear during 6:00–9:00 and 17:00–20:00 as the active drivers during these two time periods.

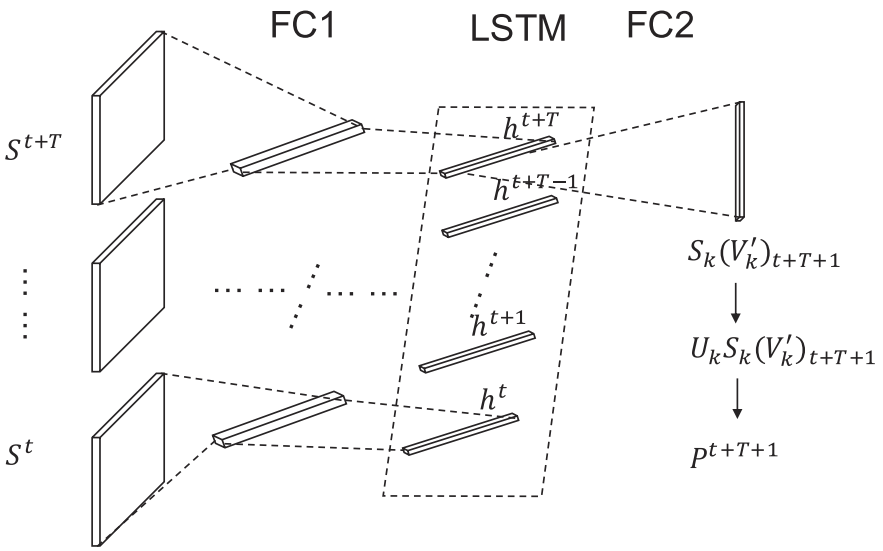


Fig. 5. Structure of the deep neural network.

With the processed data, we generate the relocation policy for 40 zip-code zones which cover around 95% of the pickups and drop-offs within study periods. We merge tiny zones to their neighbors, remove the central park area and increase the size of the zones adjacent to central park to better capture pickups/drop-offs nearby central park. The detailed information of the selected study periods and study area is shown in Fig. 6 and Table 1. Based on the summary statistics in Table 1, we observe that most pickups and drop-offs are located inside Manhattan. We also find notable asymmetry between the number of pickups and drop-offs in most of the zones for both time periods. This presents the evidence of unbalanced supply and demand and states the need for efficient taxi relocation.

Based on the historical taxi trip data, we calculate the optimal solutions of the optimal matching and stable matching problems for every 10 min with the average travel speed for relocation trips as 20 mph. The results are then grouped for training the deep learning

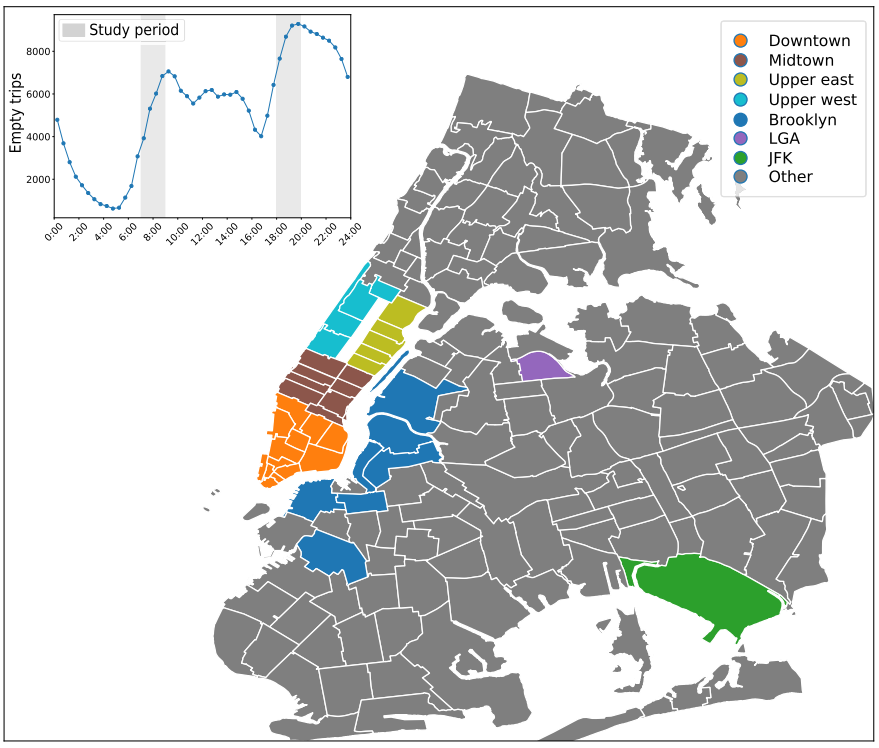


Fig. 6. Study area and study time period.

Table 1
Distribution of pickups and drop-offs.

Region	AM (7:00–9:00)		PM (18:00–20:00)	
	Pickup	Dropoff	Pickup	Dropoff
Downtown	0.334	0.314	0.367	0.391
Midtown	0.368	0.435	0.406	0.351
Upper east	0.125	0.084	0.084	0.093
Upper west	0.041	0.046	0.042	0.039
Brooklyn	0.054	0.041	0.039	0.059
LGA	0.005	0.013	0.015	0.004
JFK	0.026	0.014	0.023	0.008
SUM	0.954	0.946	0.977	0.945

model, which predicts the optimal relocation policy for the next 5-min interval. For the stable matching problem, we assume that passengers will request their rides X minutes before the actual time of pickups, where X represents the waiting time of each passenger and is assumed to follow a truncated normal distribution to ensure nonnegative waiting time.

Finally, the matching cost is measured as the empty mileage traveled to pick up each passenger trip. Besides the matching cost, we also introduce the Gini index to characterize the fairness of the matching results among individual drivers. Let c_i be the hourly income for driver i, we have

$$c_i = \frac{\text{Total income}}{\text{End of the study period} - \text{First observed time}} \tag{2}$$

where the first observed time is the end time of the trip that closest to the study period (e.g. for a vehicle with trip records as 6:10–6:30, 6:40–6:50, 7:10–7:23 for AM period, we select 6:50 as the first observed time). The Gini index is then calculated as

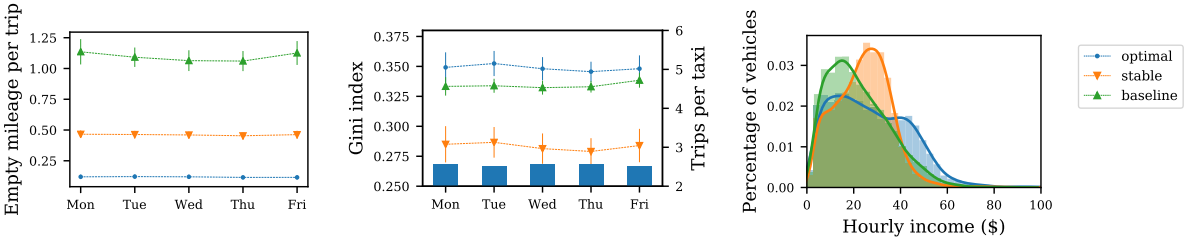
$$G = \frac{\sum_i \sum_j \left| c_i - c_j \right|}{2n^2 \bar{c}} \tag{3}$$

where n is the number of active drivers and $\bar{c}$ is the average income of all active drivers. Large G reflects high income variation among drivers and represents a high degree of income inequality.

4.2. Matching results

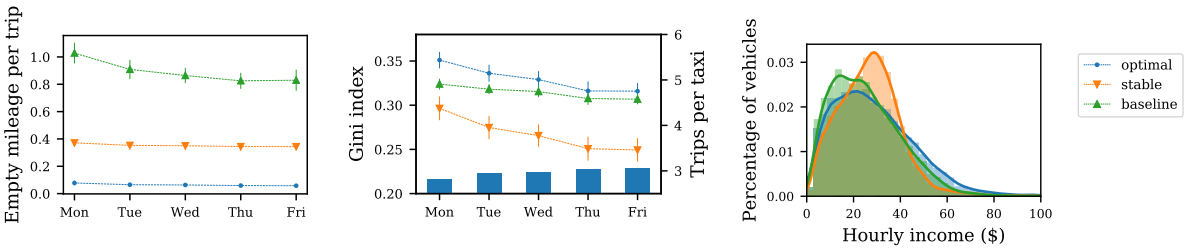
We first present the performances for solving the optimal matching and stable matching problems using historical taxi trip data. We also calculate the same metrics from the real relocation trips performed by human drivers as the baseline. The results are shown in Figs. 7 and 8.

Figs. 7a and 8a show the comparison of the empty mileage per trip among optimal matching, stable matching and actual relocations (baseline) performed by human drivers. It can be observed that both optimal matching and stable matching can significantly reduce the empty mileage in all days. For AM period, it takes 1.095 miles on average for taxi drivers to reach the next pickup location while stable matching can reduce this cost to 0.461 miles and the relocation by optimal matching only costs 0.118 miles. Considering that the average trip length is 1.632 miles for AM period, the ratios of empty mileage to total vehicle mileage traveled (VMT) are 40.2% for human drivers, 6.7% for optimal matching and 22.0% for stable matching respectively. These sufficiently demonstrate the effectiveness of both stable matching and optimal matching in reducing empty mileage, and the results of optimal matching also echo the findings reported in (Zhan et al., 2016). Similarly, for PM period, we observe that human drivers introduce on average 0.892 empty miles for serving each passenger trip while the cost of stable matching are 0.352 miles and 0.065



(a) Empty mileage (b) Gini index of individual income (c) Distribution of individual income on November 12 (Tuesday)

Fig. 7. AM matching result. Error bars represent one standard deviation.



(a) Empty mileage (b) Gini index of individual income (c) Distribution of individual income on November 12 (Tuesday)

Fig. 8. PM matching result. Error bars represent one standard deviation.

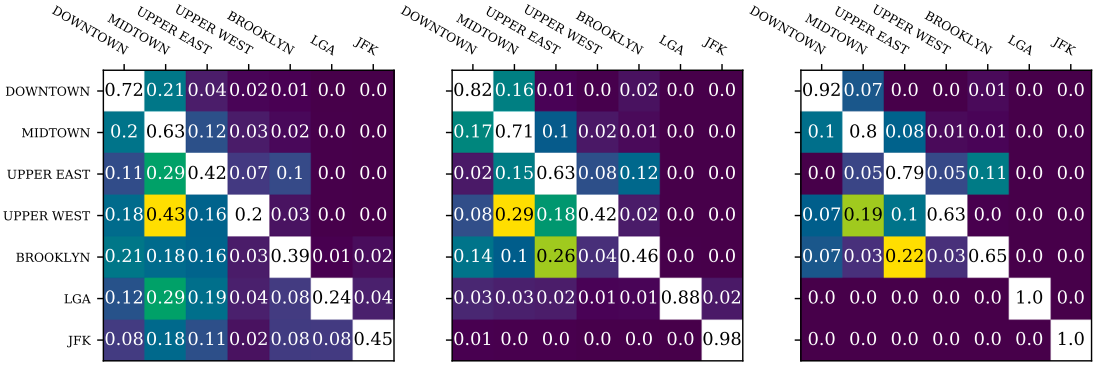
miles for optimal matching. In terms of total VMT, human drivers spend 35.1% of total mileage for relocation, but optimal matching only requires 3.8% and stable matching costs 17.6%. These results reveal the huge potential for reducing empty mileage if drivers and passengers are matched following the optimal matching or stable matching policy.

Figs. 7b and 8b compare the Gini index of individual drivers' income. We note that the optimal matching policy consistently yields the highest Gini index and the stable matching policy produces the lowest Gini index. The results indicate that, though achieving the most reduction of empty mileage, the optimal matching intensifies the income discrepancy among taxi drivers, with the Gini index being 5.7% higher than that of human drivers and 23.2% higher than that of stable matching. The resulting relocation policy is therefore unlikely to be followed by drivers who seek to maximize their revenue. On the other hand, stable matching provides more stable solutions for the OMS market with revenue maximizing drivers and will lead to a more effective reduction in empty trip mileage. We further show the distribution of individual income on one selected day in Figs. 7c and 8c. Compared with the relocation results by human drivers, the optimal matching leads to more drivers with higher income (over 40 $/h) but also a notable portion of drivers with relatively lower income, which eventually leads to higher values of the Gini index. On the contrary, the distribution of hourly income under stable matching gives a higher median and mode, which demonstrates the effectiveness of stable matching in balancing the income among individual drivers.

In Fig. 8b we also observe that matching results of Monday evening have a much higher Gini index than other weekdays'. This is because of the relatively low demand in Monday evening. Since the Gini index takes the average income in the denominator and low demand leads to low average income, it is unsurprising that the results of Monday evening have higher Gini index.

To gain further insights on the differences among different relocation strategies, we aggregate and average the relocation results into the 7×7 matrix to represent the fraction of relocation trips between large zones (shown in Fig. 6) and the results are shown in Figs. 9 and 10. It can be observed that, except for JFK, over 40% of human drivers choose downtown and midtown Manhattan as their relocation destination regardless of their last drop-off locations. Compared to the results in Table 1 where more than 70% of pickups are located in downtown and midtown Manhattan, this suggests that the relocations performed by human drivers are likely driven by their knowledge of passenger demand distribution. This also signifies a high degree of impatience for human drivers, where they are observed to make much more frequent relocations outside the drop-off areas and we believe the primary reason is due to lack of information. Instead, according to the results of stable matching and optimal matching, the most efficient relocation strategy for reducing empty mileage is to wait at the drop-off location in high demand areas and relocate vehicles in low-demand zones based on the distribution of passenger demand. For example, while human drivers prefer to relocate to midtown, both stable matching and optimal matching recommend distributing more empty vehicles to the upper east area during the AM peak period.

Finally, we also observe that the relocation decisions at LGA and JFK airports differ significantly from those in Manhattan. First,



(a) Baseline (b) Stable (c) Optimal

Fig. 9. Aggregated AM relocation policy.

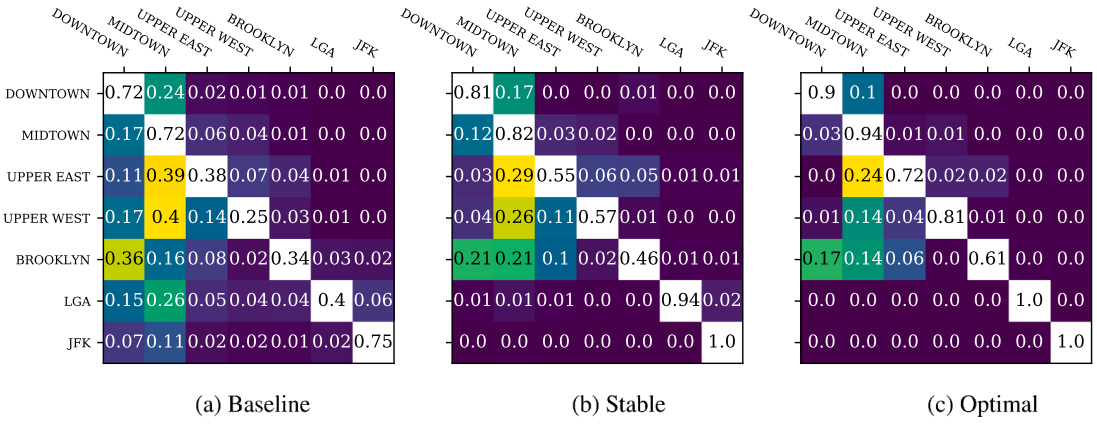


Fig. 10. Aggregated PM relocation policy.

nearly no human drivers who drop off the passengers in Manhattan are observed to relocate to the two airports. Here our result suggests that it is sufficient to serve all passenger demand at the airports with incoming taxi trips. Second, both stable matching and optimal matching relocate few vehicles that arrive at the airports. Meanwhile, around 50% of human drivers choose to return to Manhattan after dropping off passengers at LGA for both AM and PM periods. And the percentage of returning drivers at JFK is 40% during AM period and 23% during PM period. The large number of relocation trips from the airports to Manhattan is likely one of the major sources of empty travel miles considering the distance between Manhattan and the two airports. In our dataset, the relocation trips from/to the airports contribute to 19.77% of the total empty mileage during the AM period and 18.9% during the PM period. This result suggests the necessity to design better guidelines and incentives for taxis and FHVs at the airports, so that both empty mileage and emissions can be reduced. Moreover, we have seen recent evidence that many airports started to change their pickup policies for for-hire vehicles. For example, LAX adopted a new service called LAX-it (Los Angles World Airports, 2020) to handle the waiting of taxis and FHVs by setting a specific waiting area near the airport. The shuttle service is in place to transport passengers to the designated taxi and OMS pickup area, which can reduce the drivers' waiting time and help OMS drivers to park and wait at the airport.

4.3. Training results

The results from optimal matching and stable matching produce the set of relocation policies over 224 weekdays. We then use the derived relocation policy from the first 200 days to train the deep learning model. Since deep learning model usually requires a large sample size to achieve satisfactory training results, we enlarge the training set via the moving window approach with the step size of 1 min. For example, we calculate the relocation policy for "8:00–8:05", "8:01–8:06", ···, "8:25–8:30", with their corresponding training input being the OD flows during "7:55–8:00","7:56–8:01", ···, "8:20–8:25". If there is no relocation trip within 5 min, we use the interpolation of the average value in other time periods to fill the corresponding value in the relocation matrix. LRA is then performed to reduce the dimension of relocation matrix from n^2 ($n = 40$ in our case) to a lower dimension k ($k \ll n$). The input and output are grouped for every 10 consecutive time steps. Thereby the final training data for the deep learning model contains 23200 samples (116 for each day), with the input size being $10 \times 40 \times 40$ and the output shape being $k \times 1$. Mini-batch gradient descent is used to improve training stability. 5-fold cross-validation is used for different combinations of parameters. For each fold, we implement the early stopping strategy which stops training after 5 consecutive epochs without sufficient improvement ($>10^{-4}$) in MSE. After tuning hyperparameters, we set the learning parameters as (Dimension k, Batch size, Learning rate, FC1 size, LSTM units) = (80, 32, 0.0001, 800, 256) for the DNN models.

Fig. 11 shows the training curve with the chosen hyperparameters, where the training MSE is smoothed by every epoch. The

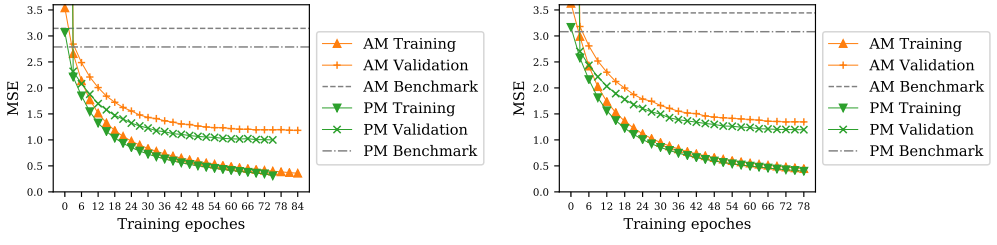


Fig. 11. Training curves for the deep learning model.

Table 2
Summary of the training results.

Model	AM (7:00–9:00)		PM (18:00–20:00)	
	MSE	MAE	MSE	MAE
Optimal	1.1007	0.0503	0.9262	0.0502
Stable	1.2635	0.0590	1.1197	0.0559

training epochs are found to be 86 and 76 for optimal matching models for AM and PM, and 80 for stable matching models. In each figure, the dashed lines present the MSE calculated from directly using the mean value of the training targets as the prediction result. This gives the benchmark results where the relocation decisions are static. The gap between the gray lines and the MSE of the validation results show how much variance can be explained by the trained deep learning model. It can be observed that the trained model reduces the MSE from around 3 to 1.2 compared with the benchmark, which means it can capture around 60% of the variance. In addition to MSE, we also calculate the maximum absolute error (MAE), which is more tangible as it shows the maximal element-wise error of the predicted relocation policy. The results are shown in Table 2. The MAE is observed to be smaller than 6%. Note in Fig. 9 and Fig. 10 the maximum element-wise difference between each pair of strategy is much greater than this value, which indicates the MAE is small enough to distinguish different relocation strategies.

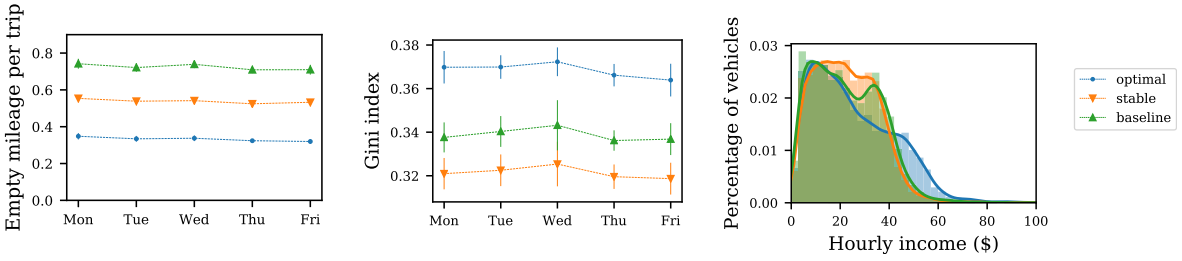
4.4. Simulation results

We next use the trained deep learning model to perform the simulation of taxi relocation using the data of the rest 24 weekdays. We consider three scenarios: (1) the optimal scenario in which we use the optimal matching policies; (2) the stable scenario in which stable matching policies are used; (3) the baseline scenario in which we directly follow the relocation policies adopted by real-world taxi drivers during the same time period of the simulation data. For the optimal and stable matching scenario, we further compare our method (DNN) with three other benchmark policies, including

1. The one week policy which uses the solved optimal/stable relocation policy at the same time period from the previous week (One week).
2. The average policy which uses the average solved optimal/stable relocation policy at the same time period during the first 200 days (Average).
3. The oracle policy which directly uses the solved optimal/stable relocation policy of the same time period (Oracle).

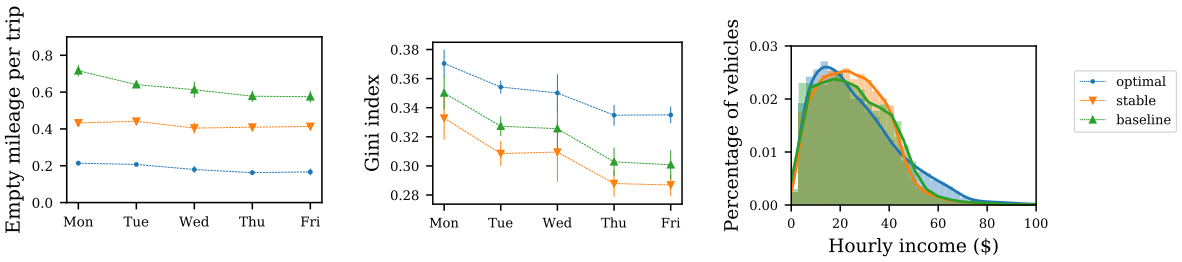
We consider that the request time follows the normal distribution and the relocation policy results in a set of probabilistic events where a taxi at zone i may be relocated to zone j based on the relocation matrix. In light of the randomness introduced by the relocation policy and trip request time, all the following results are obtained as the average of 50 simulation runs, where we generate 5 samples of trip request time following the request time distribution and for each sample, we run the simulation with 10 different random seeds.

After running the simulations for all three scenarios, the mismatch counts are found to be zero, which suggests that all passengers can be served by following the relocation policy. We then take the average of all random seeds and summarize the results in Figs. 12 and 13. We observe that the tendency of the matching cost and Gini index from the simulation results largely agree with the matching results. This finding supports the ability of the deep learning model in learning the key philosophy behind the optimal matching and stable matching strategies. However, the overall reduction in empty mileage or improvements in the fairness of income are less effective as compared to the previously reported matching results. This is expected as previous matching results were identified as the global optimal solutions of the corresponding optimization problems. In general, the learned optimal matching strategy achieves the highest reduction in empty mileage but it also leads to greater income inequality. On the other hand, the learned stable matching policy is observed to reduce empty mileage and reduce income inequality simultaneously. In terms of the one day's (Nov 12, Tuesday)



(a) Empty mileage (b) Gini index of individual income (c) Distribution of individual income

Fig. 12. AM simulation results.



(a) Empty mileage (b) Gini index of individual income (c) Distribution of individual income

Fig. 13. PM simulation results.

income distributions, from Figs. 12c and 13c one can still observe that (1) there is a group of high come (over 40 dollars/h) drivers in the optimal matching strategy, and (2) higher median and mode of the income distribution are obtained under the stable matching strategy. We would like to point out that the income distribution generated under the baseline scenario is different from that performed by the human drivers in the trip data, which is shown in Figs. 7c and 8c. This is due to that the simulation assigns vehicles to passengers based on batching matching while this rule is not followed by taxis in the real world due to the lack of coordination and perfect information. We further present a more detailed comparison between the matching results and the simulation results in Table 3. Compared with the result of the optimal matching, the simulated optimal strategy has higher empty mileage, which is unsurprising because we consider the proactive relocation distance to the zones' centroids in the simulation and this prolongs the total relocation mileage. Similarly, the simulated stable strategy also presents higher empty mileage as compared to the stable matching result. However, the empty mileage per trip of the baseline simulation is around 30% lower than the real-world observations. This reflects how the local batch matching alone, as the common practice in the current OMS markets (Uber, n.d.), may contribute to reducing total empty mileage. Moreover, when compared with the baseline scenario, we observed that the simulated optimal matching strategy can save 54.1% empty mileage per trip for AM and 70.3% for PM; the simulated stable matching strategy can save 25.6% empty mileage per trip for AM and 32.8% for PM. In terms of the Gini index, it is found that the simulated results have higher Gini indexes than their corresponding matching results. This is because the batch matching essentially resembles to the optimal matching and represents system optimal solution, which may sacrifice the benefits of individual drivers and hence increases the income inequality. Such an observation is also consistent with our calculated results in Figs. 7 and 8. Finally, we report the comparisons between DNN and three other benchmark policies, and the results are shown in Fig. 14, Fig. 15 and Table 4. Within each scenario, it can be observed that DNN in most of the cases is better than other benchmark methods. For optimal scenario, it is found that, on average, the DNN model results in better relocation performances as compared to the two other benchmark methods based on historical observations, and achieves similar performance when compared to the oracle policy. In particular, when compared to the strategy using the solved optimal policy from the previous week, the DNN can save around 34.42 miles during AM period and 167.26 miles during PM for each day, which means that DNN can save 201.68*365*5/7 = 52580 miles per year during peak hours. This gap is tripled when compared to the average strategy. For the stable scenario, it can be observed that DNN outperforms the average policy in both AM and PM periods in terms of empty mileage and income inequality. However, the other heuristic policy which uses the solved stable policy in the previous week (one week policy) may achieve slightly better results than DNN for the AM period. Note in the training results shown in Table 2, the DNN model for stable matching in AM has the highest MSE and MAE. This implies under the current setting, the stable matching policy for AM is harder to learn than other cases. The issue can be improved by performing feature engineering and revising the network structure specifically for AM. For PM, we observe that DNN is better than the one week policy and reaches a similar performance to the oracle policy. Adding the differences between DNN and the one week policy for AM and PM periods, we found that the DNN still costs 17.51 fewer miles than the one week policy for each day. These results suggest our method in general outperforms other benchmark methods. It should be noted that when compared with the baseline scenario, the simple methods based on historical observations for optimal/stable matching may also achieve good relocation performances. For example, it is found that using the average optimal relocation policy at the same time period during the first

Table 3
Comparison between matching results and simulation results.

Scenario		Matching result		Simulation result	
		Empty mileage per trip	Gini index	Empty mileage per trip	Gini index
Optimal	AM	0.1183 (−89.2%)	0.3487	0.3325 (−54.1%)	0.3683
	PM	0.0647 (−92.7%)	0.3298	0.1851 (−70.3%)	0.3487
Stable	AM	0.4614 (−57.9%)	0.2831	0.5382 (−25.6%)	0.3213
	PM	0.3528 (−60.4%)	0.2676	0.4195 (−32.8%)	0.3050
Baseline	AM	1.0950	0.3341	0.7243	0.3388
	PM	0.8917	0.3145	0.6238	0.3210

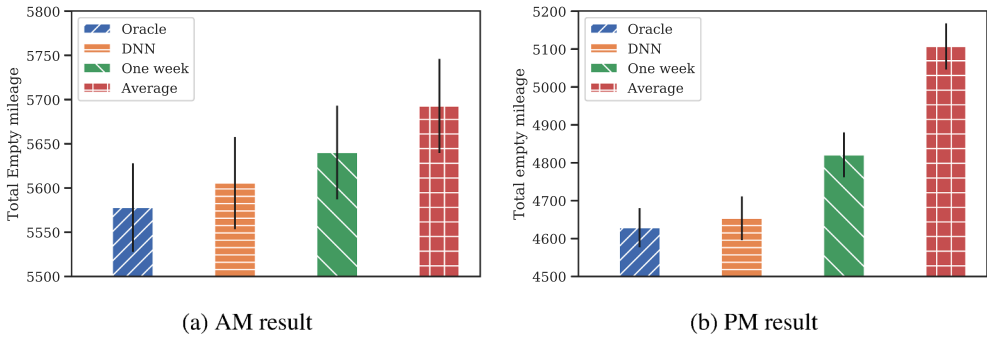


Fig. 14. The comparison of total empty mileage between the DNN and other benchmark policies for optimal matching.

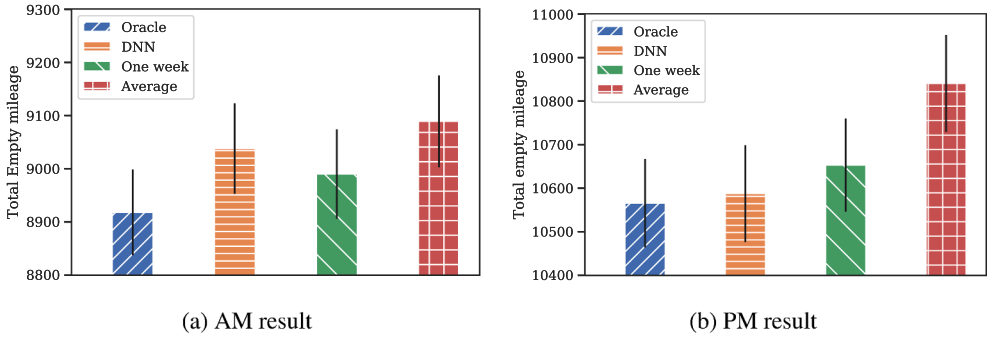


Fig. 15. The comparison of total empty mileage between the DNN and other benchmark policies for stable matching.

Table 4
Mean and standard deviation of the total mileage and income Gini index by using the DNN and other benchmark policies.

Scenario	Model	AM			PM		
		Mean	Standard deviation	Gini index	Mean	Standard deviation	Gini index
Optimal	DNN	5605.57	51.97	0.3684	4653.36	57.56	0.3484
	One week	5639.99	52.92	0.3680	4820.62	58.88	0.3480
	Average	5692.58	53.13	0.3675	5106.50	60.57	0.3483
	Oracle	5577.82	50.01	0.3690	4628.47	51.40	0.3509
Stable	DNN	9037.66	85.05	0.3328	10587.68	106.74	0.3149
	One week	8989.90	84.02	0.3320	10652.95	111.01	0.3155
	Average	9089.02	86.19	0.3329	10840.27	111.00	0.3167
	Oracle	8918.01	80.54	0.3319	10565.67	101.35	0.3137

200 days can also save 53.4% empty mileage per trip for AM and the saving is 67.4% for PM. The underlying reason is that the system dynamics including trip demand and taxi distribution were relatively stable in neighboring weeks. Note we perform the experiment using the NYC taxi data in 2013. During that time the taxi market was relatively stable and the data from the previous week may well capture the market dynamics of the current week. However, this may no longer be the case in recent years with a growing number of for-hire vehicles. Under this circumstance, it is less likely that the solved policies are similar over days or weeks so the simple methods may not work well. On the other hand, with the simple DNN model which does not rely on the time of day and day of the week information, we still observe that DNN outperforms other baselines. This suggests the validity of the DNN model for stable scenarios and its potential for proactive relocation in a non-stationary market. And the DNN's performance can be further improved by incorporating network layouts that are tailored to spatio-temporal prediction, and by adding more features related to the time of day, spatial attributes, and exogenous factors that may affect demand and supply of OMS. These directions are worth future investigation using more recent data sources.

4.5. Sensitivity analysis

While we have shown the effectiveness of the learned relocation policy in reducing empty miles and improving the fairness of income, these results only represent the performance during normal weekdays and we are also interested in the performances of the

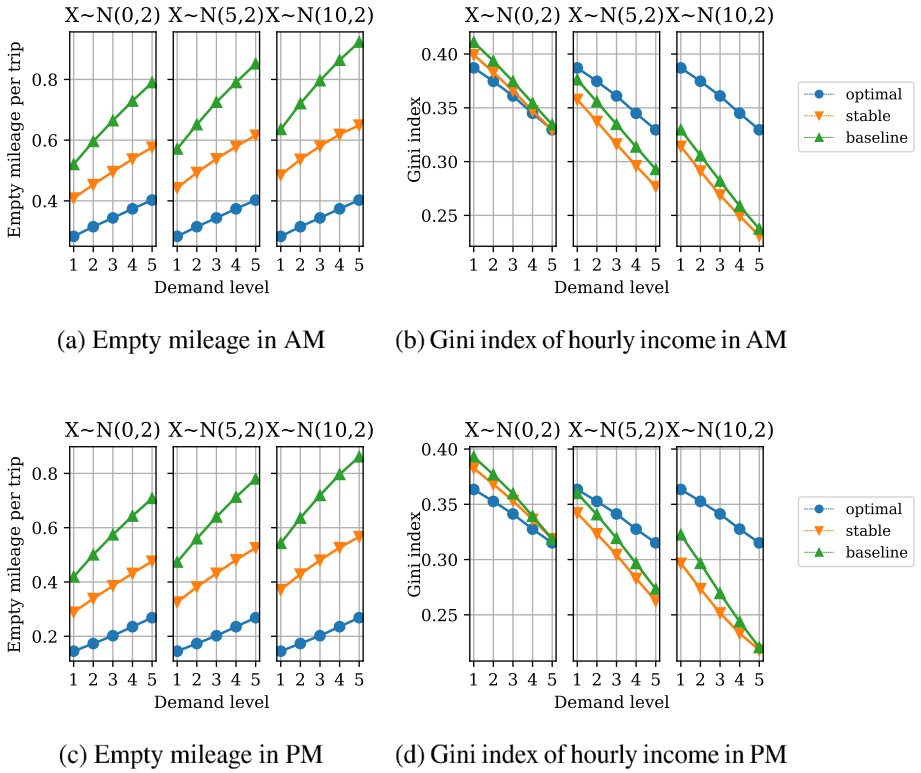


Fig. 16. Changes of empty mileage and Gini index under different demand levels and request time distributions.

learned policy when there are demand anomalies. To this end, we perform a sensitivity analysis to investigate the robustness of the proposed method.

We first perform the simulations using the same deep learning model and altering the demand levels (80%, 90%, 100%, 110% or 120% of the total passenger demand) and distributions of trip request time ($X \sim \mathcal{N}(0, 2)$, $X \sim \mathcal{N}(5, 2)$ and $X \sim \mathcal{N}(10, 2)$). For each combination of demand level (1–5, 1 for 80% and 5 for 120% of passengers) and X, the results are obtained by averaging over the results using the 24 days of test data. We find that, even when the demand is enlarged by 20%, the number of mismatches is still 0. This means that all passengers can be served with their expected departure time.

The empty mileages per trip under different demand levels and request time are shown in Fig. 16b and c. It can be observed that the empty mileage per trip increases along as the demand level gets higher. This is because there are fewer available vehicles for each time step when the demand decrease but the supply does not change. We note that the increase of empty mileage per trip is linear for real taxis' policy but for the proposed method, the empty mileage becomes stable as the overall demand increases by 10%, this indicates the efficiency of our method under high demand level. As for the request time, we observe that the empty mileage per trip will increase as the mean of the request time increases. And the marginal increasing rates ranked from high to low are baseline, stable matching and optimal matching. This is because the larger the mean, the earlier we need to start considering matching the passenger with available drivers. This may result in vehicles being relocated for a longer distance. Nevertheless, our method is found to be consistently better than other benchmark policies with increasing mean values of request time, and this confirms that the request time does not influence the effectiveness of our method and is indicative of the robustness of our model.

In Fig. 16b and d, we present the result of the Gini index for different parameters. We observe that the Gini index decreases as the demand level becomes higher, the reason is similar to the case of Monday evening in which there are more trips per taxi and thus has a higher average income. With similar income variance, the Gini index should be lower when the average income is higher. We also observe that the request time plays a very important role in the fairness of drivers' income. The underlying reason is that the earlier we observe the request, the more likely we assign it to a vehicle that already waits in the same zone. That is, there is less chance for a vehicle being idled for a long time when the mean of X becomes larger. Note when $X \sim \mathcal{N}(0, 2)$, this mechanism does not work and the stable and baseline scenarios with batch matching would even generate more unfair matching than the optimal strategy. In the end, we report the consistency of the simulation results by altering the standard deviation for the request time distribution. In particular, we choose the mean as 5 min and vary the standard deviation from 1 to 4 min, with the increment of 1 min for each scenario. And we run each scenario for 50 times to calculate the average total empty mileage and the Gini index of drivers' income. The results are summarized in Table 5. In general, we observe that the changes in the standard deviation of the trip request time do not show a significant influence on the simulation results for all models. These results therefore confirm that the performance of the proposed method is stable under different variances of the request time.

Table 5
Simulation results under different standard deviations for the request time distribution.

Scenario		AM			PM		
Model	Request time distribution	Mean	Standard deviation ($\times 10^{-2}$)	Gini index	Mean	Standard deviation ($\times 10^{-2}$)	Gini index
Optimal	$\mathcal{N}(5, 1)$	0.3326	0.25	0.3682	0.1929	0.16	0.3488
	$\mathcal{N}(5, 2)$	0.3327	0.24	0.3684	0.1927	0.18	0.3484
	$\mathcal{N}(5, 3)$	0.3327	0.26	0.3686	0.1934	0.19	0.3487
	$\mathcal{N}(5, 4)$	0.3331	0.27	0.3684	0.1930	0.15	0.3484
Stable	$\mathcal{N}(5, 1)$	0.5491	0.48	0.3319	0.4411	0.37	0.3138
	$\mathcal{N}(5, 2)$	0.5494	0.44	0.3328	0.4407	0.37	0.3149
	$\mathcal{N}(5, 3)$	0.5501	0.44	0.3315	0.4408	0.40	0.3137
	$\mathcal{N}(5, 4)$	0.5513	0.48	0.3300	0.4422	0.36	0.3114
Baseline	$\mathcal{N}(5, 1)$	0.7239	0.60	0.3388	0.6391	0.51	0.3199
	$\mathcal{N}(5, 2)$	0.7244	0.58	0.3386	0.6383	0.51	0.3198
	$\mathcal{N}(5, 3)$	0.7257	0.59	0.3376	0.6392	0.54	0.3177
	$\mathcal{N}(5, 4)$	0.7290	0.61	0.3355	0.6410	0.50	0.3148

5. Conclusions

In this study, we propose the approach for deriving proactive relocation policies to reduce the empty mileage for large-scale OMS systems. We consider two types of passenger-vehicle matching: the optimal matching that minimizes the total mileage; and the stable matching that reduces both the empty mileage and the discrepancy of individuals' income. We model the relocation policy as a matrix in which each element represents the possibility of relocating vehicles to a specific zone. We first obtain the optimized relocation policies by solving the optimal/stable matching problem using historical data. These policies are then served as the targets to train a deep neural network and low rank approximation (LRA) is introduced in this step to improve the training efficiency and accuracy. Finally, we use the trained model to predict the relocation policy in real time.

We conduct numerical experiments using the yellow taxi data in New York City for AM (7:00–9:00) and PM (18:00–20:00) for 224 workdays in 2013. The matching results indicate the huge potential to reduce the empty mileage. It is found that optimal matching can reduce the percentage of empty mileage in total VMT from 40.2% to 6.7% for AM period, and from 35.1% to 3.8% for PM period. Stable matching is not as efficient as optimal matching, which can reduce the percentage of empty mileage to 22.0% for AM period and to 17.6% for PM period. Nevertheless, the matching results show that stable matching yields a more concentrated income distribution with a higher median and mode. We further explore the difference of relocation policies between different strategies. It is observed that the real drivers exhibit an experience-driven relocation pattern which choosing relocation destination based on the demand distribution, while the optimal/stable matching results suggest a different way to make relocation decision, which relocates less vehicles between different zones and chooses the relocation destination based on the previous drop-offs.

We use the relocation policies of the first 200 days to train the deep neural network and perform the simulation over the rest 24 days' data with different scenarios. The simulation results are highly consistent with the matching results, which confirms the effectiveness of our method. The simulated result of optimal strategy can save 54.1% empty mileage for AM and 70.3% for PM. The simulated stable strategy can save 25.6% for AM and 32.8% for PM. For optimal strategy, we also compare our method with the oracle and two other benchmark methods based on historical observation. The results show that our DNN model in general performs better than historical data based benchmark approaches and the performances are close to the actual oracle policy. Sensitivity analysis is also performed to evaluate the robustness of our method. And we confirm that the proposed methods remain efficient and robust under different levels of passenger demand and distributions of the request time.

This work suggests the significant potential to reduce the total empty mileage by adopting proactive relocation policies in OMS systems, and by simulation we show the proposed method can predict these policies and achieve a reasonable performance. One future direction is to compare our method with results that are obtained from predictions of future demand and then optimizing in real-time. Moreover, the current method can be improved with better prediction of relocation policies. Note the daily traffic contains rich information and the relocation policy may not solely depend on the distribution of riders, and one can also consider the fusion of data from different sources and perform feature engineering to improve the results from the deep learning model.

CRediT authorship contribution statement

Zengxiang Lei: Conceptualization, Methodology, Formal analysis, Visualization. **Xinwu Qian:** Conceptualization, Methodology, Formal analysis. **Satish V. Ukkusuri:** Conceptualization, Supervision.

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.trc.2020.102678.

References

Bai, Ruibin, Li, Jiawei, Atkin, Jason A.D., Kendall, Graham, 2014. A novel approach to independent taxi scheduling problem based on stable matching. J. Oper. Res. Soc. 65 (10), 1501–1510.

Castiglione, Joe, Chang, Tilly, Cooper, Drew, Hobson, Jeff, Logan, Warren, Young, Eric, Charlton, Billy, Wilson, Christo, Mislove, Alan, Chen, Le, et al., 2016. Tncs today: a profile of San Francisco transportation network company activity. San Francisco County Transportation Authority (June 2016), 2016.

Chuah, Seong Ping, Xiang, Shili, Wu, Huayu, 2018. Optimal rebalancing with waiting time constraints for a fleet of connected autonomous taxi. In: 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), pages 629–634, Feb 2018.

Deerwester, Scott, Dumais, Susan T., Furnas, George W., Landauer, Thomas K., Harshman, Richard, 1990. Indexing by latent semantic analysis. J. Am. Soc. Inform. Sci. 41 (6), 391–407.

Donovan, Brian, Work, Dan, 2016. New York City Taxi Trip Data (2010–2013). University of Illinois at Urbana-Champaign.

Dubins, Lester E, Freedman, David A, 1981. Machiavelli and the gale-shapley algorithm. Am. Math. Month. 88 (7), 485–494.

Erhardt, Gregory D., Roy, Sneha, Cooper, Drew, Sana, Bhargava, Chen, Mei, Castiglione, Joe, 2019. Do transportation network companies decrease or increase congestion? Sci. Adv. 5 (5), eaau2670.

Gale, David, Sotomayor, Marilda, 1985. Some remarks on the stable matching problem. Discr. Appl. Math. 11 (3), 223–232.

Hochreiter, Sepp, Schmidhuber, Jürgen, 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.

Hopcroft, John E, Karp, Richard M, 1973. An n5/2 algorithm for maximum matchings in bipartite graphs. SIAM J. Comput. 2 (4), 225–231.

Huang, Kai, de Almeida Correia, Goncalo Homem, An, Kun, 2018. Solving the station-based one-way carsharing network planning problem with relocations and non-linear demand. Transport. Res. Part C: Emerg. Technol. 90, 1–17.

Iglesias, Ramon, Rossi, Federico, Wang, Kevin, Hallac, David, Leskovec, Jure, Pavone, Marco, 2018. Data-driven model predictive control of autonomous mobility-on-demand systems. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 1–7.

Kek, Alvina G.H., Cheu, Ruey Long, Meng, Qiang, Fung, Chau Ha, 2009. A decision support system for vehicle relocation operations in carsharing systems. Transport. Res. Part E: Logist. Transport. Rev. 45 (1), 149–158.

Kingma, Diederik P., Ba, Jimmy, 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kuhn, Harold W., 1955. The hungarian method for the assignment problem. Naval Res. Logist. Quart. 2 (1–2), 83–97.

Kümmel, Michal, Busch, Fritz, Wang, David Z.W., 2016. Taxi dispatching and stable marriage. Proc. Comput. Sci. 83, 163–170.

Los Angles World Airports. About laxit, accessed March, 2020. Available online at https://www.flylax.com/lax-it.

Lyft. How drivers and passengers are paired, accessed September, 2019. Available online at https://help.lyft.com/hc/en-us/articles/115012926847-How-drivers-and-passengers-are-paired.

Markovsky, Ivan, 2008. Structured low-rank approximation and its applications. Automatica 44 (4), 891–909.

Miao, Fei, Han, Shuo, Lin, Shan, Stankovic, John A., Zhang, Desheng, Munir, Sirajum, Huang, Hua, He, Tian, Pappas, George J., 2016. Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach. IEEE Trans. Autom. Sci. Eng., vol. 13, 2, pp. 463–478.

Miao, Fei, Han, Shuo, Lin, Shan, Wang, Qian, Stankovic, John A., Hendawi, Abdeltawab, Zhang, Desheng, He, Tian, Pappas, George J., 2017. Data-driven robust taxi dispatch under demand uncertainties. IEEE Trans. Control Syst. Technol. (99), 1–17.

Mnich, Matthias, Schlotter, Ildikó, 2017. Stable marriage with covering constraints–a complete computational trichotomy. In: International Symposium on Algorithmic Game Theory. Springer, pp. 320–332.

NYC Taxi, Limousine Commission, and Department of Transportation. Improving efficiency and managing growth New York's for-hire vehicle sector, 2019.

Pavone, Marco, Smith, Stephen L., Frazzoli, Emilio, Rus, Daniela, 2012. Robotic load balancing for mobility-on-demand systems. Int. J. Robot. Res. 31 (7), 839–854.

Qian, Xinwu, Zhan, Xianyuan, Ukkusuri, Satish V., 2015. Characterizing urban dynamics using large scale taxicab data. In: Engineering and Applied Sciences Optimization. Springer, pp. 17–32.

Sayarshad, Hamid R., Oliver Gao, H., 2018. A scalable non-myopic dynamic dial-a-ride and pricing problem for competitive on-demand mobility systems. Transport. Res. Part C: Emerg. Technol. 91, 192–208.

Smith, Stephen L., Pavone, Marco, Schwager, Mac, Frazzoli, Emilio, Rus, Daniela, 2013. Rebalancing the rebalancers: optimally routing vehicles and drivers in mobility-on-demand systems. In: 2013 American Control Conference, pp. 2362–2367.

Uber marketplace. How does Uber match riders with drivers?. Available online at < https://marketplace.uber.com/matching/ > (accessed September, 2019).

Weikl, Simone, Bogenberger, Klaus, 2013. Relocation strategies and algorithms for free-floating car sharing systems. IEEE Intelligent Transport. Syst. Magaz. 5 (4), 100–111.

Wen, Jian, Zhao, Jinhua, Jaillet, Patrick, 2017. Rebalancing shared mobility-on-demand systems: A reinforcement learning approach. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 220–225.

Winter, Konstanze, Cats, Oded, van Arem, Bart, Martens, Karel, 2017. Impact of relocation strategies for a fleet of shared automated vehicles on service efficiency, effectiveness and externalities. June. In: 2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), pp. 844–849.

Zhan, Xianyuan, Qian, Xinwu, Ukkusuri, Satish V., 2016. A graph-based approach to measuring the efficiency of an urban taxi service system. IEEE Trans. Intell. Transport. Syst. 17 (9), 2479–2489.

Zhang, Rick, Pavone, Marco, 2016. Control of robotic mobility-on-demand systems: a queueing-theoretical perspective. Int. J. Robot. Res. 35 (1–3), 186–203.

Zhang, Rick, Rossi, Federico, Pavone, Marco, 2016. Model predictive control of autonomous mobility-on-demand systems. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 1382–1389.

Zhu, Chenguang, Prabhakar, Balaji, 2017. Reducing inefficiencies in taxi systems. In: 2017 IEEE 56th Annual Conference on Decision and Control (CDC), pp. 6301–6306.