滴滴派单算法

金德才

2023

- A Taxi Order Dispatch Model based On Combinatorial Optimization
- Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach
- A Deep Value-network Based Approach for Multi-Driver Order Dispatching

派单算法背景

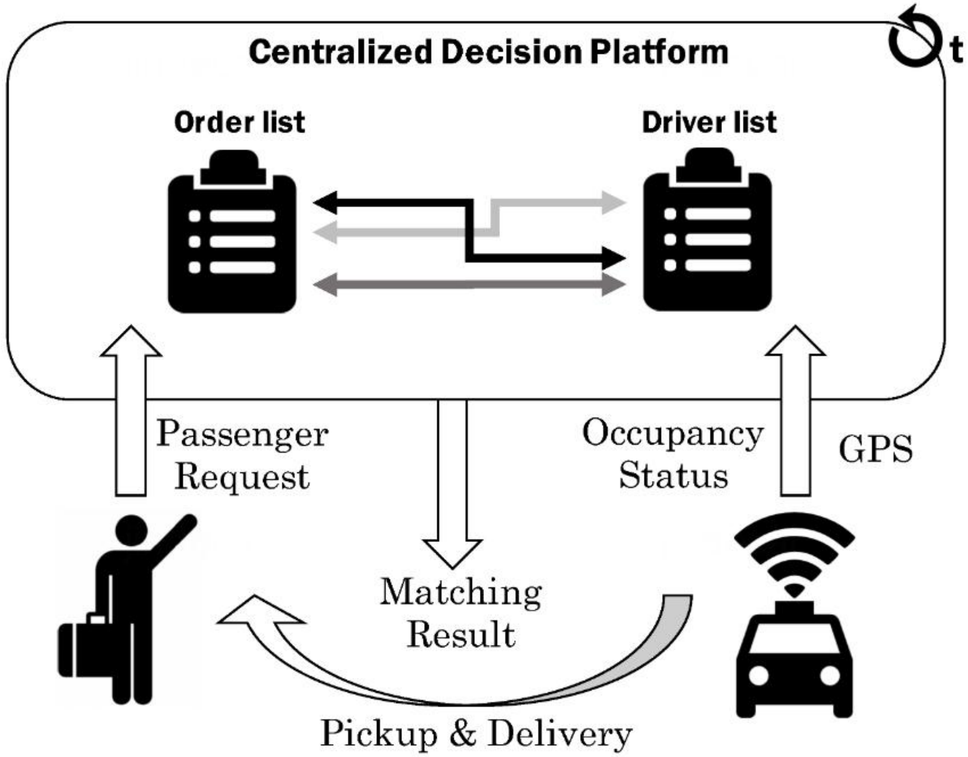


Figure 2: Architecture of order dispatch in on-demand ride-hailing services.

A Taxi Order Dispatch Model based On Combinatorial Optimization

- 订单匹配算法
- 乘客目的地预估算法

订单匹配算法

If there are N orders to be dispatched to M drivers, we represent the dispatch result as a matrix

$$\begin{pmatrix} a_{11} & \cdots & a_{1M} \\ \vdots & a_{ij} & \vdots \\ a_{N1} & \cdots & a_{NM} \end{pmatrix}, \text{where } 1 \le i \le N, 1 \le j \le M,$$

$$\text{and } a_{ij} = \begin{cases} 1 & \text{order } i \text{ is dispatched to driver } j, \\ 0 & \text{order } i \text{ is not dispatched to driver } j. \end{cases}$$

订单匹配算法

$$\begin{cases} \max\limits_{a_{ij}} E_{SR} = \dfrac{\sum_{i=1}^{N}[1-\prod_{j=1}^{M}(1-p_{ij})^{a_{ij}}]}{N}, \\ \text{s.t. } \forall j, \sum\limits_{i=1}^{N} a_{ij} \le 1, a_{ij} \in \{0,1\}. \end{cases}$$

订单匹配算法

Algorithm 1 Proposed HillClimbing Algorithm

```
1:  procedure HillClimbing(A, P)
2:      for i ← 1, M do
3:          D[i] ← j with the maximum probability P[i][j]
4:      end for
5:      for i ← 1, N do
6:          E[i] ← success rate of order i with D
7:          E0 ← average(E[i])
8:      end for
9:      for i ← 1, N do
10:         U ← drivers that are not assigned order i
11:         for j ← 1, len(U) do
12:             k ← U(j)
13:             if replace D[k] with order i, E0 increases then
14:                 replace D[k] with order i
15:                 E[i] ← success rate of order i with D
16:                 E0 ← average(E[i])
17:             end if
18:         end for
19:     end for
20: end procedure
```

司机接单概率预估

$$p_{ij} = p(y = 1|o_i, d_j) = \frac{1}{\exp(-\mathbf{w}^{\mathrm{T}}\mathbf{x}_{ij})}.$$

- Order-Driver related features: the pick-up distance, the broadcasting counts of the order to the driver, whether the order is in front of or behind the driver's current driving direction.
- Order related features: the distance and the estimated time arrival (ETA) between the origin and the destination, the destination category (airport, hospital, school, business district, etc.), traffic situation in the route, historical order frequency at the destination.
- Driver related features: Long-term behaviors (include historical acceptance rate of a driver, active locations of a driver, preference of different broadcast distances of a driver, etc.) and short-term interests of a driver such as orders recently accepted or not, etc.
- Supplemental features, such as day of the week, hour of the day, number of drivers and orders nearby.

效果评估

实验分流方法：
将乘客和实际分流

对照模型描述：
model 1 算一个综合的匹配分，综合应用A*，learning-to-rank
model 2 使用多智能体算法优化全局最小等待时间

Table 3: Measurements used in our comparison.

Measurement	Abbrev.
Percentage of served calls (Success Rate)	SR
Averaged pick up time(the time from the order accepted by a driver to the passenger getting on the car)	APT
Averaged dispatch time (the time from the passenger make the order to one driver accepted the order)	ADT
Percentage of cancelled calls (Cancellation Rate)	CR
Average total number of calls served by each cab (Fleet Utilization)	FU

Table 4: Number of drivers and orders.

	Driver Number	Order Number
Flow 1	27268	345631
Flow 2	27310	345683
Flow 3	27297	345711

Table 5: Results of three models.

	SR(%)	APT (min)	ADT (sec)	CR(%)	FU
Model 1	79.8	4.538	87	24.1	4.32
Model 2	80.1	4.163	106	22.8	4.47
Our Model	84.4	4.169	89	23.2	4.71

乘客目的地预估算法

Step 1. Estimate $\boldsymbol{\mu}_i, \Sigma_i$ for each destination of a user.

Step 2. Calculate $p(Y = y_i)$:

$$p(Y = y_i) = \frac{\text{freq}(y_i))}{\sum_{j=1}^{n} \text{freq}(y_j)},$$

and compute $p(T, Lat, Lng|Y = y_i)$ using Eq. (5).

Setp 3. Calculate $p(Y = y_i|T, Lat, Lng)$ using Bayesian rule:

$$p(Y = y_i|T, Lat, Lng) =$$

$$\frac{p(T, Lat, Lng|Y = y_i)p(Y = y_i)}{\sum_{j=1}^{n} p(T, Lat, Lng|Y = y_j)p(Y = y_j)}.$$

Step 4. Rank the destinations by $p(Y = y_i|T, Lat, Lng)$ and provide them as a list.

$$Lat, Lng, T|Y = y_i \sim N_3(\boldsymbol{\mu}_i, \Sigma_i). \quad (5)$$

循环数据的均值和方差计算

3:00, 15:00, 21:00的平均数是多少?

$$\begin{cases} \min\limits_{\mu} \sum\limits_{k=1}^{m} [\text{distance}(t_k, \mu)]^2, \\ \text{s.t. } \mu \in [0, 24), \end{cases} \tag{4}$$

$$\begin{cases} \min\limits_{\mu} \sum\limits_{k=1}^{m} [|(|t_k - \mu| - 12)| - 12)]^2, \\ \text{s.t. } \mu \in [0, 24). \end{cases}$$

$$\text{distance}(t_1, t_2) = \begin{cases} |t_1 - t_2| & \text{if } |t_1 - t_2| \le 12, \\ 24 - |t_1 - t_2| & \text{if } |t_1 - t_2| > 12. \end{cases}$$

$$\text{distance}(t_1, t_2) = -|(|t_1 - t_2| - 12)| + 12.$$

$$\sigma^2 = \frac{1}{m-1} \sum_{k=1}^{m} [|(|t_k - \mu| - 12)| - 12]^2.$$

目的地预估算法效果

对照模型： K-nearest neighbor method (K=100) based on two features: departure time and location.

效果： achieves a high accuracy about 93%, which is 4% higher than the baseline model on top-3 accuracy

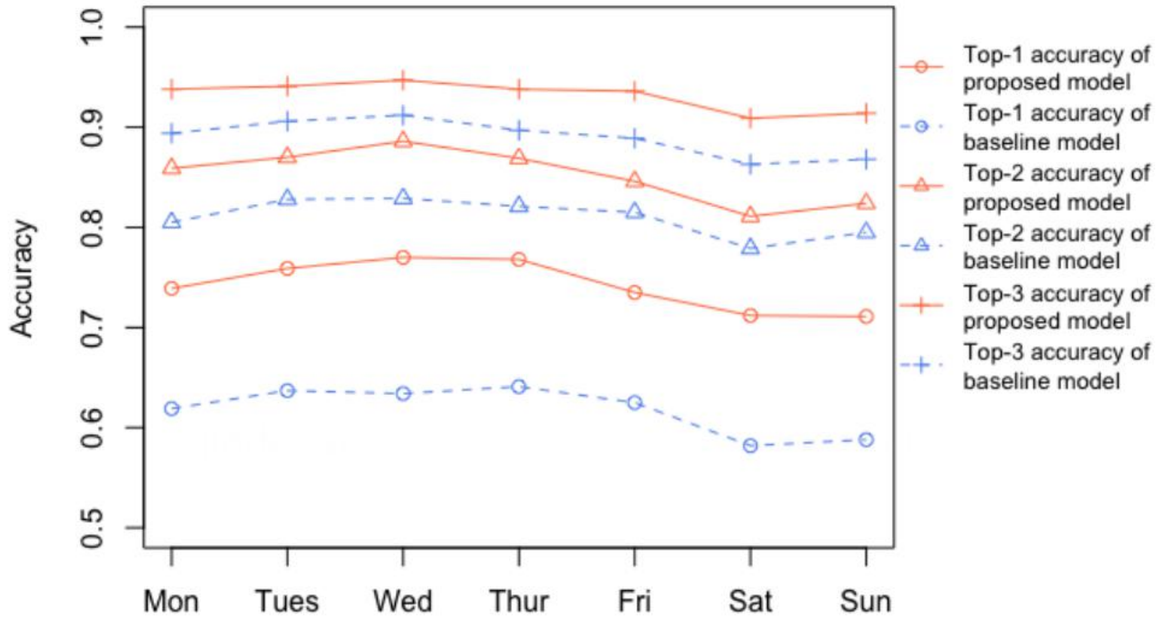


Figure 6: Top-1, top-2 and top-3 accuracies of our proposed model and the baseline model.

Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach

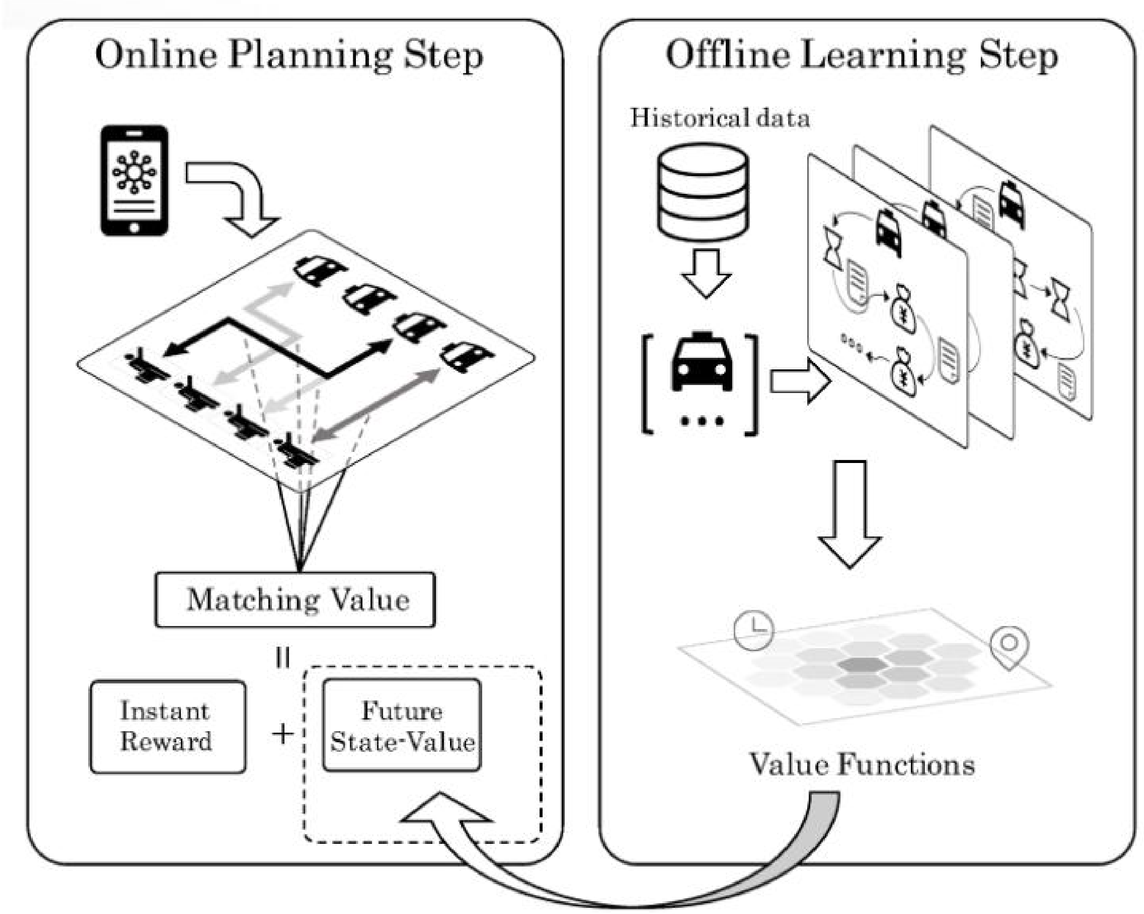
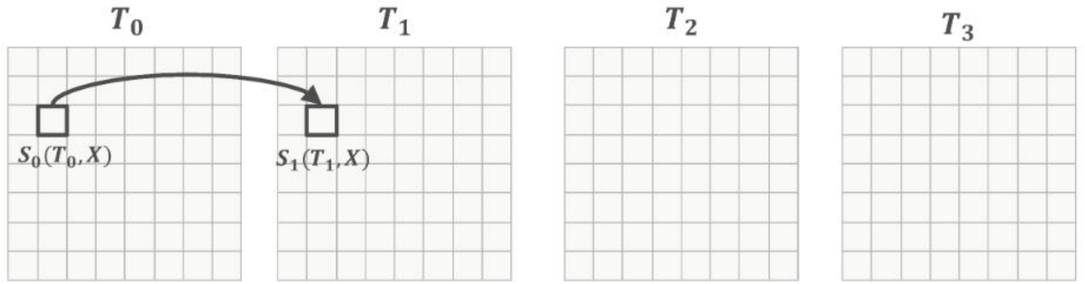


Figure 1: Illustration of the proposed algorithm.

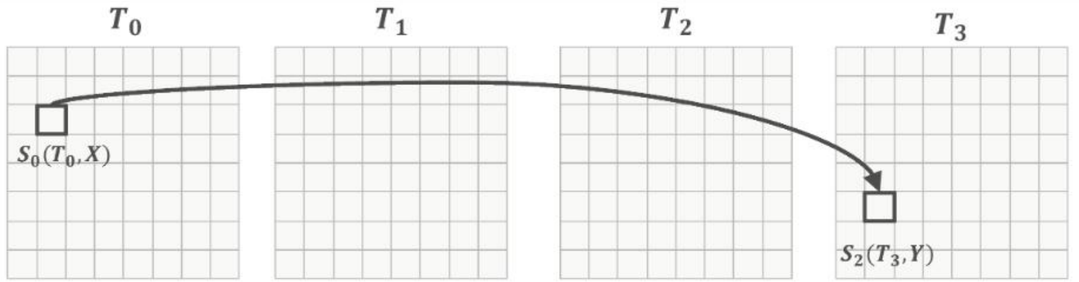
Learning

智能体：每个司机
环境及状态：时空索引
动作：两种类型，接单和不接单
回报：不接单0，接单实时算

$$R_\gamma = \sum_{t=0}^{T-1} \gamma^t \frac{R}{T}.$$



Idle action: $V(S_0) \leftarrow V(S_0) + \alpha(0 + \gamma V(S_1) - V(S_0))$



Serving action: $V(S_0) \leftarrow V(S_0) + \alpha(R_\gamma + \gamma^3 V(S_2) - V(S_0))$

Learning

Algorithm 1 Policy evaluation (dynamic programming) for the local-view MDP

Input: Collect historical state transitions $D = \{(s_i, a_i, r_i, s_i')\}$; each state is composed of a time and space index: $s_i = (t_i, g_i)$.

Initialize $V(s)$, $N(s)$ as zeros for all possible states.
for $t = T - 1$ to 0 **do**
  Find a subset $D^{(t)}$ where $t_i = t$ in s_i.
  for each sample (s_i, a_i, r_i, s_i') in $D^{(t)}$ **do**
    $N(s_i) \leftarrow N(s_i) + 1$,
    $V(s_i) \leftarrow V(s_i) + \frac{1}{N(s_i)}[\gamma^{\Delta t(a_i)} V(s_i') + R_\gamma(a_i) - V(s_i)]$.
  end for
end for

Return: Value function $V(s)$ for all states

- 连续式还是分幕式：这里建模为连续式的强化学习问题，应用TD方法进行求解
- 探索的说明：这里状态动作价值的探索是如何进行的呢？用户在不同时间出现在不同地点的可能性极大，所以每个时空的状态价值没有特意探索，也不太好进行探索，因为订单的起点不受平台控制，对于偏僻地区或者订单少的地区可能需要考虑在时间或者空间上进行平滑一下

Planning

$$\underset{a_{ij}}{\operatorname{argmax}} \quad \sum_{i=0}^{m}\sum_{j=0}^{n} Q_\pi(i,j)a_{ij} \tag{4}$$

$$s.t. \quad \sum_{i=0}^{m} a_{ij} = 1, \quad j = 1,2,3\ldots,n$$
$$\sum_{j=0}^{n} a_{ij} = 1, \quad i = 1,2,3\ldots,m \ . \tag{5}$$

where

$$a_{ij} = \begin{cases} 1 & \text{if order j is assigned to driver i} \\ 0 & \text{if order j is not assigned to driver i} \end{cases}$$

这里状态动作价值由上一步离线计算，规划由KM算法进行求解

Planning - max advantage

$$\underset{a_{ij}}{\text{argmax}} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} A_\pi(i,j)a_{ij} \tag{6}$$

$$s.t. \quad \begin{matrix} \sum_{i=1}^{m} a_{ij} \leq 1, & j = 1,2,3\ldots,n \\ \sum_{j=1}^{n} a_{ij} \leq 1, & i = 1,2,3\ldots,m \end{matrix} . \tag{7}$$

where

$$A_\pi(i,j) = \gamma^{\Delta t_j} V(s'_{ij}) - V(s_i) + R_\gamma(j) \tag{8}$$

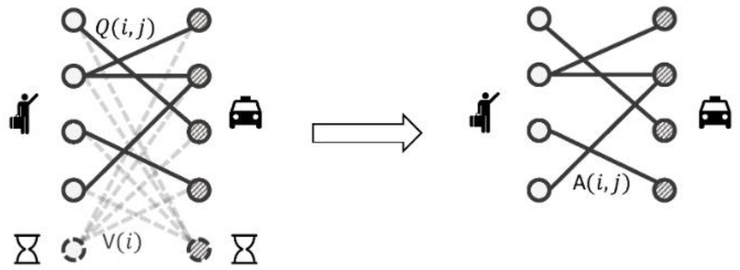


Figure 5: Advantage trick. We remove all connections with the default node.

Algorithm 2 Real-time order dispatch algorithm

Input: Compute value functions $V(s)$ for all states using algorithm 1, and load them as a Look-up-table (LUT).

1: **for** every timestamp of online order dispatch **do**
2: Collect available drivers $i \in [1, \ldots, m]$ and active orders $j \in [1, \ldots, n]$.
3: Perform required filters on driver-order pairs and remove invalid connections.
4: Compute advantage function for each valid driver-order pair using (8).
5: Solve (6) using the KM algorithm.
6: Forward the matching information to relevant drivers and passengers. Remaining orders and drivers will go to the next round.
7: **end for**

算法的改进点：
1.价值函数拆分为3部分，更好理解和通过调整各部分权重配合运营需求
2.裁掉了接单概率小的司机，移除了默认连接，提高了效率

效果评测 - Toy Example

- 9* 9网格
- 司机起点和订单终点从均匀分布中采样
- 接单距离不超过曼哈顿距离2
- 订单取消概率分布符合Gauss(2.5,2)[0,5]
- 订单起点和下单时间分布如下

$$\pi^{(1)} = 1/3, \quad \pi^{(2)} = 2/3;$$

$$\mu^{(1)} = [3, 3, 5], \quad \mu^{(2)} = [6, 6, 15];$$

$$\sigma^{(1)} = [2, 2, 3], \quad \sigma^{(2)} = [2, 2, 3].$$

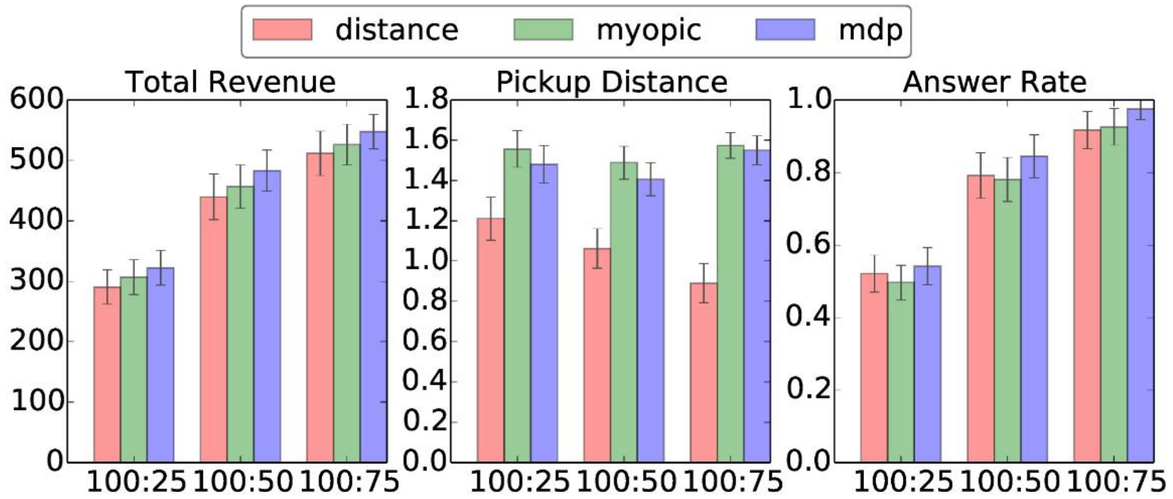


Figure 6: Comparison of distance-based method, myopic method and the proposed MDP method in three metrics on the toy example environment. X-axis stands for the order-driver ratios. Better viewed in color.

效果评测 - Simulate

Table 1: Comparison of policy iteration results in the simulator for four median-sized cities. Rate stands for the completion rate for orders.

city	MDP V_0		MDP V_{converge}		
Metric	GMV	Rate	GMV	Rate	Days to Converge
City C	+0.6%	+0.5pp	+0.8%	+0.9pp	4
City D	+0.9%	+1.0pp	+1.4%	+1.5pp	8
City E	-0.1%	+0.1pp	+0.5%	+0.5pp	13
City F	+0.8%	+0.7pp	+1.2%	+1.1pp	7

司机起点和订单采用真实数据的回放

效果评测 - Online Test

- 实验方法：将一天分成大的时间片，3小时或者6小时，轮流应用线上和实验算法，过程持续两周
- γ = 0.9
- gains in global GMV and completion rate ranging from 0.5% to 5%

(a) 18:00-18:10, evening peak (b) 09:00-09:10, after morning peak

Figure 8: Sampled value function for the same city at different times. Red indicates higher values, blue for lower ones. Better viewed in color.

对于b，早高峰把大部分司机带到中心区，导致这个地方车辆过多，接单困难，价值下降,接下来系统会适当减少来到中心区的订单

A Deep Value-network Based Approach for Multi-Driver Order Dispatching

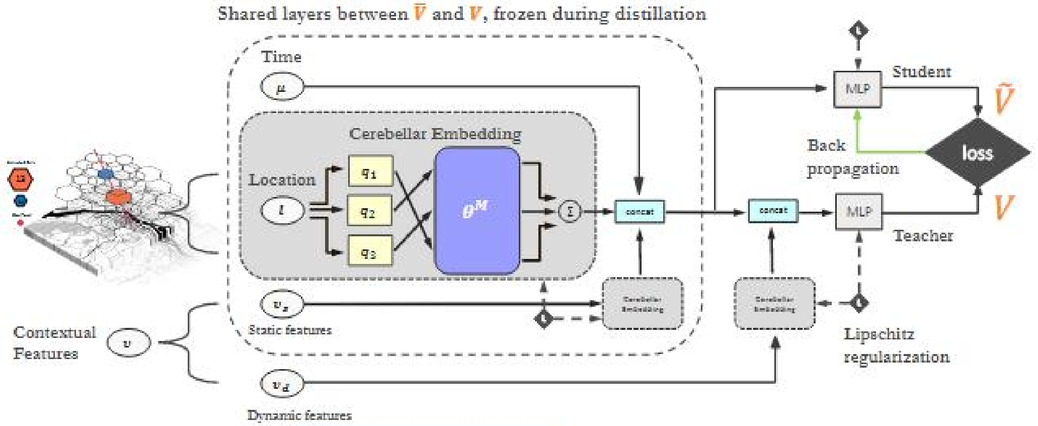


Figure 1: Feature marginalization and network structure of the Lipschitz-regularized dispatching value function V and its distilled network $\tilde{V}$.

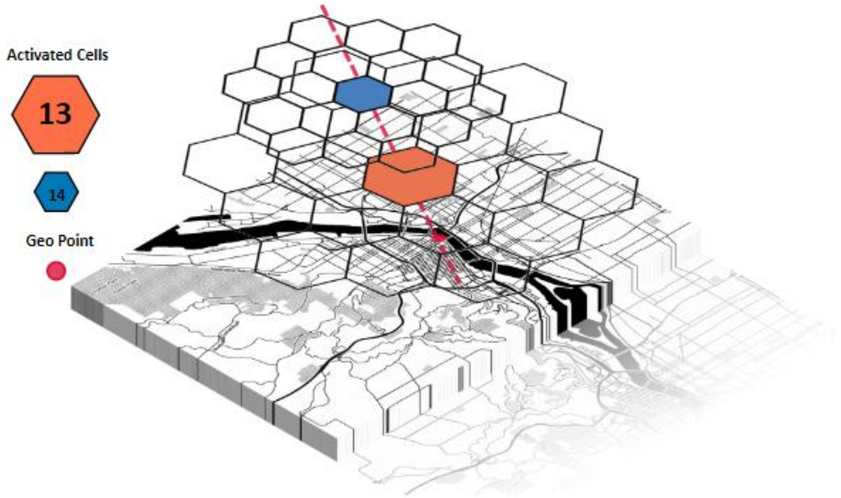


Figure 2: Coarse Coding with Hierarchical Hexagon Grid. The geo point (red) activates two grid cells (orange and blue). The final representation is the average of the two grid cells' embedding vectors.

A Deep Value-network Based Approach for Multi-Driver Order Dispatching

Table 2: Results from online AB test.

	Relative Improvement		
	Answer rate (%)	Finish rate (%)	TDI (%)
City B	**0.60** ± 0.057	**0.49** ± 0.069	**0.73** ± 0.210
City C	**1.16** ± 0.062	**1.11** ± 0.083	**0.93** ± 0.198
City D	**1.39** ± 0.077	**1.20** ± 0.113	**1.65** ± 0.482