DOI:10.7544/issn1000-1239.20210905
59(2):329-341,2022

基于深度强化学习的网约车动态路径规划

郑渤龙[1] 明岭峰[1] 胡 琦[1] 方一向[2] 郑 凯[3] 李国徽[1]
[1](华中科技大学计算机科学与技术学院 武汉 430074)
[2](香港中文大学(深圳)数据科学学院 广东深圳 518172)
[3](电子科技大学计算机科学与工程学院 成都 610054)
(bolongzheng@hust.edu.cn)

Dynamic Ride-Hailing Route Planning Based on Deep Reinforcement Learning

Zheng Bolong[1], Ming Lingfeng[1], Hu Qi[1], Fang Yixiang[2], Zheng Kai[3], and Li Guohui[1]
[1](*School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan* 430074)
[2](*School of Data Science, The Chinese University of Hong Kong (Shenzhen), Shenzhen, Guangdong* 518172)
[3](*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu* 610054)

Abstract With the rapid development of the mobile Internet, many online ride-hailing platforms that use mobile apps to request taxis have emerged. Such online ride-hailing platforms have reduced significantly the amounts of the time that taxis are idle and that passengers spend on waiting, and improved traffic efficiency. As a key component, the taxi route planning problem aims at dispatching idle taxis to serve potential requests and improving the operating efficiency, which has received extensive attention in recent years. Existing studies mainly adopt value-based deep reinforcement learning methods such as DQN to solve this problem. However, due to the limitations of value-based methods, existing methods cannot be applied to high-dimensional or continuous action spaces. Therefore, an actor-critic with action sampling policy, called AS-AC, is proposed to learn an optimal fleet management strategy, which can perceive the distribution of supply and demand in the road network, and determine the final dispatch location according to the degree of mismatch between supply and demand. Extensive experiments on New York and Haikou taxi datasets offer insight into the performance of our model and show that it outperforms the comparison approaches.

Key words mobile information processing systems; spatial-temporal data mining; deep reinforcement learning; ride-hailing route planning; fleet management

摘 要 随着移动互联网的快速发展,许多利用手机App打车的网约车平台也应运而生.这些网约车平台大大减少了网约车的空驶时间和乘客等待时间,从而提高了交通效率.作为平台核心模块,网约车路径规划问题致力于调度空闲的网约车以服务潜在的乘客,从而提升平台的运营效率,近年来受到广泛关注.现有研究主要采用基于值函数的深度强化学习算法(如deep Q-network, DQN)来解决这一问题.然而,由于基于值函数的方法存在局限,无法应用到高维和连续的动作空间.提出了一种具有动作采样策略的执行者-评论者(actor-critic with action sampling policy, AS-AC)算法来学习最优的空驶网约车

收稿日期:2021-09-02;**修回日期**:2021-11-19
基金项目:国家自然科学基金项目(61902134,62011530437);湖北省自然科学基金项目(2020CFB871);中央高校基本科研业务费专项资金(2019kfyXKJC021,2019kfyXJJS091)
This work was supported by the National Natural Science Foundation of China (61902134, 62011530437), Hubei Natural Science Foundation (2020CFB871), and the Fundamental Research Funds for the Central Universities (2019kfyXKJC021, 2019kfyXJJS091).

调度策略,该方法能够感知路网中的供需分布,并根据供需不匹配度来确定最终的调度位置.在纽约市和海口市的网约车订单数据集上的实验表明,该算法取得了比对比算法更低的请求拒绝率.

关键词 移动信息处理系统;时空数据挖掘;深度强化学习;网约车路径规划;车队调度

中图法分类号 TP399

网约车平台,如滴滴出行[1]和优步[2]等,允许用户在手机 app 或小程序上发起打车请求,由平台通过智能算法将请求与空闲的网约车进行匹配,在近年来得到了快速发展.一方面,与传统的司机在街道上寻找乘客的模式相比,网约车平台有效地减少了网约车的空驶时间,提升了司机的收益,并大大减少了乘客打不到车的情况,提高了城市交通系统的运行效率[3].另一方面,这些网约车平台积累了大量的包含乘客的出行模式等信息的数据,例如轨迹数据和订单数据,这些数据可以进一步用于交通领域的许多应用,如需求预测[4-6]、订单分派[7-9]和车队管理[10-12].

作为网约车平台的核心模块,网约车路径规划需要管理城市中的所有网约车,为空闲网约车指派巡航路线,将其调度到潜在的乘客位置,同时又要协调各辆网约车,防止其相互之间竞争乘客,以充分利用有限的网约车资源,平衡网约车的供应与乘客的打车需求之间的差距[13].然而,现实情况是,尽管在需求端每天有数万请求通过网约车平台进行处理,但仍然有许多的打车请求由于附近没有空闲网约车而未能及时处理,或是需要等待很长的时间才能打到车.而在供应端,依然有相当一部分的网约车无法接到乘客,需要空驶很长一段时间才能找到合适的乘客[14].这种供需不平衡的现象在大城市中每天都在发生,这既降低了平台的收益,也影响了用户的体验,并进一步导致了交通拥堵和资源浪费.因此,有效的网约车路径规划对网约车平台有着重要的意义.

制定有效的网约车路径规划策略主要存在 3 项挑战:1)需要对动态变化的供需分布进行建模,以准确预测每个区域对网约车资源的需求;2)需要防止空闲网约车聚集在热门区域,而导致这些区域出现运力过剩,而其他区域又运力不足的情况;3)需要考虑处于偏僻区域的网约车,让其能更快驶离当前区域,从而减少无效的巡航时间.

现有方法通过对历史网约车数据进行建模来解决上述挑战.RHC[15]通过构建供需模型调度网约车以应对静态交通环境.深度强化学习(deep reinforcement learning, DRL)算法[11,13,16]可以处理动态的交通环境,具有更好的性能.但是 DRL 算法在建模时面临着简化设定的问题,例如,大多数 DRL 算法都假设所有空闲网约车同时进行调度,并且假设网约车能够在下一时间片到达调度终点,而这在实际交通环境中是不可能的.而本文的调度策略会在网约车空闲时立即对其进行路径规划,不存在不合理的假设,更贴近真实的交通环境.

相比基于值函数的算法(如 deep Q-network, DQN),actor-critic 采用策略梯度的算法可以在连续动作空间或高维动作空间中选取合适的动作,而这是基于值函数的算法无法做到的;相比单纯策略梯度的算法,actor-critic 采用类似 DQN 的策略评估算法,可以进行单步更新而不是回合更新,比单纯策略梯度的算法更有效.因此,本文基于 actor-critic 的算法,提出了一种供需感知的深度强化学习算法,用于网约车调度.首先,通过定义智能体、状态、动作和奖励,将网约车路径规划问题转化为一个 Markov 决策过程(Markov decision process, MDP).然后,设计了一个具有动作采样策略的执行者-评论者(actor-critic with action sampling policy, AS-AC)网络结构,以学习最佳的调度策略.

本文的主要贡献包括 3 个方面:

1) 提出了一个基于实时供需状态的动态网约车路径规划框架,实现高效的大规模空闲网约车调度,通过包含实时的供需信息来适应动态变化的环境.

2) 设计了一种带有动作采样的 AS-AC 算法来选择可行的动作,增加了动作选择的随机性,从而有效地防止竞争.

3) 使用真实的网约车订单数据进行了大量实验,实验结果表明提出的方法相比对比方法有着更低的请求拒绝率.

1 相关工作

1.1 车队管理

车队管理是网约车平台的一项主要决策任务[17],它包括确定车队的规模和配置、车队分配、车辆路线

规划等,这些都广泛应用于交通运输中[18].其中网约车路径规划是车队管理问题的核心任务,许多研究致力于有效地解决此问题或其变体,例如车辆路由问题(vehicle routing problem, VRP)[19]、最小车队问题(minimum fleet problem, MFP)[20]和乘车问题[21].大多数方法首先将这些问题转换成经典的图论问题,然后采用组合优化的方法去求解.例如,Vazifeh 等人[20]将最小车队问题建模为二部图上的最大匹配问题,然后采用 Hopcroft-Karp 算法确定服务所有订单所需的最小车队的规模.

随着网约车服务的普及,调度城市中的空闲网约车以平衡不同位置的供需成为一个新的研究热点.许多工作对城市中的网约车供需进行建模,调度空闲网约车来平衡供需,以最大程度地减少乘客的等待时间和网约车的空驶时间[12-13,15,22-24].例如,文献[18]将网约车调度任务建模为最小费用流(minimum cost flow, MCF)问题,并使用组合优化的方法对其进行求解.后退水平控制模型[15]根据供需模型和网约车的实时位置来调度网约车,以减少网约车空驶的距离.然而,这些基于模型的方法局限于预先确定的供需模型,因此难以适应车辆状态和出行请求都在不断变化的真实交通环境[16].

1.2 深度强化学习

深度强化学习将具有强大感知力的深度学习[25]方法和具有优秀决策力的强化学习[26]方法相结合,通过试错的方式来学习智能体在观测状态下应采取的最佳动作.DRL 利用神经网络来估计状态动作值,并通过更新网络参数来学习最佳行为策略,已经在许多具有挑战性的领域取得了突破性的进展,例如,围棋[27]、电子游戏[28-29]和自动驾驶[30].然而,主流的深度强化学习算法,如 DQN[31-32]和 Reinforce[33]主要适用于低维度且样本容易获取的任务,而在高维和非静态的动作空间中表现不佳[11].本文通过实时的供需估计和巧妙的采样策略,运用深度强化学习算法来解决网约车路径规划问题.

1.3 基于深度强化学习的网约车路径规划

由于网约车路径规划这一决策任务可以很自然地建模为 Markov 决策问题,许多最近的研究都致力利用深度强化学习设计模型无关的方法来调度网约车[11,13-14,17-18,24,34-35].这些方法可以分为基于值函数的方法(如 DQN)和基于策略梯度方法(如 A3C[36]).基于值函数的方法主要使用 DQN 来估计动作的价值,通过神经网络计算出动作价值,然后选择价值最大的动作.例如,MOVI[16]通过卷积神经网络(convolutional neural network, CNN)来提取供需分布特征并采用分布式的 DQN 策略学习单独的车辆的最优调度动作.COX[13]采用聚类算法将路网划分为许多用于网约车调度的区域,再通过环境感知的需求预测模型和实时的网约车状态来计算区域级别的供需,最后将这些供需信息加入到状态中,提高 DQN 的表现.与基于值函数的方法不同,策略梯度方法用神经网络来近似策略,采用梯度上升法来寻找最优策略.例如,CoRide[17]将订单调度和车队管理概述为部分可观测 Markov 决策问题(partially observable Markov decision process, POMDP),然后采用深度确定性策略梯度(deep deterministic policy gradient, DDPG)算法[37]来学习订单匹配和车队管理的最优策略,以最大化累积司机收入(accumulated driver income, ADI)和订单应答率(order response rate, ORR).文献[11]在求解动作空间时考虑了地理环境和协作环境,消除了无效的网格从而减小了动作空间,并提出 contextual actor-critic 多智能体强化学习算法用于网约车学习行为策略.

尽管这些方法相比基于模型的方法有着更强大的能力以适应复杂动态的交通环境,它们仍然受限于不现实的问题设定.例如,大多数方法要求所有的网约车在每个时间片同时进行调度决策,并要求网约车能在下一时间片到达调度终点,这在现实的动态交通场景中很难保证.此外,只把邻居网格作为动作空间会使得处于偏僻区域的网约车很难尽快驶离这些区域.与现有的方法不同,本文不对所有的网约车进行统一调度,而是将每一辆网约车看作单独的同质智能体,共享网络参数和调度策略,当一辆网约车变为空闲状态时立即对其进行调度.此外,提出的供需感知的强化学习方法采用变化的动作空间来代替静态的动作空间,在以邻近网格作为可行调度动作的基础上加上了全局热门区域,并采用动作过滤和采样来平衡全局供需分布.本文的框架在网约车变为空闲时就为其分配搜索路线,并激励网约车在其到达调度终点前尽快被分配请求,更接近真实的交通场景.

2 网约车路径规划问题

本节将介绍网约车路径规划问题的背景,包括问题设置、问题定义和框架概述,并将该问题描述为一个 Markov 决策过程.表 1 总结了本文常用的符号.

Table 1 Summary of Notation

表 1 常用符号

符号	描述
$g_i \in G$	$G=\{g_1,g_2,\cdots,g_{\|G\|}\}$的第 i 个网格
$t_j \in T$	$T=\{t_1,t_2,\cdots,t_{\|T\|}\}$的第 j 个时间片
$s_t \in S$	网约车在时刻 t 的状态
$a_t \in A$	在时刻 t 采取的调度动作
$r_t \in R$	在时刻 t 获得的奖励
γ	折扣因子
δ^t	空闲网约车与等待请求数量之差
C^t	时刻 t 的网约车供应量
D^t	时刻 t 的打车需求
$\rho(i)$	第 i 个网格的供需差
α	学习率
p_i	第 i 个动作的优先级
τ	控制优先级使用的参数
$rank(\cdot)$	动作价值排序函数
$x_t(l)$	时刻 t 在位置 l 的空闲网约车数
$y_t(l)$	时刻 t 在位置 l 的预测请求数
$\psi_t(l)$	时刻 t 在位置 l 的供需分布差异值

2.1 问题叙述

网约车路径规划问题由空间中的一组网约车车队、流形式的请求和一个管理所有网约车的调度中心所组成.

定义 1. 网约车队.假设空间中有一组网约车车队 $X=\{x_i \mid i \in \mathbb{N}\}$,每个网约车初始化在路网中的任意位置,然后巡航来寻找合适的请求.在整个调度过程中车队大小固定为 $|X|$.

定义 2. 请求.假设有一组请求流 $\Omega=\{\omega_j \mid j \in \mathbb{N}\}$,每个请求 $\omega_j=(o,d,t^o,t^*)$有一个起点 o,一个终点 d,请求开始时间 t^o 和最长等待时间 t^*.如果有空闲的网约车可以在 t^* 内到达起点 o,则该请求可以被该空闲网约车服务,否则,该请求将被调度中心拒绝.

定义 3. 调度中心.调度中心包括 3 个模块:车队管理模块、请求预测模块和调度策略模块.车队管理模块跟踪每辆网约车的实时位置和占用状态,并更新供应分布.请求预测模块根据历史订单信息预测未来的乘客请求分布.调度策略模块基于供需分布将空闲的网约车调度到未来请求可能出现的位置.

定义 4. 拒绝率.拒绝率(reject rate, RR)是被拒绝的请求数占请求总数的比例.使用 $\Omega_{rej}=\{\omega_k \mid k \in \mathbb{N}\}$来表示拒绝请求的集合,其中 ω_k 是第 k 个被拒绝请求,则拒绝率计算如下:

$$RR=\frac{|\Omega_{rej}|}{|\Omega|}. \tag{1}$$

给定一组网约车和一个乘客请求流,调度中心需要为空闲网约车规划巡航路线以使得其能够尽快服务潜在的请求,从而最小化请求拒绝率.

2.2 框架总览

ST-GCSL[24]模型将请求预测问题建模为图节点预测问题,采用图神经网络技术,同时提取短期与长期时空依赖关系,具有很高的预测精度.因此,在框架中,使用 ST-GCSL 作为请求预测模型.具体而言,使用 Uber H3 库① 将城市路网划分为一组六边形网格 $G=\{g_1,g_2,\cdots,g_{|G|}\}$;将一天划分为一个时间片序列,表示为 $T=\{t_1,t_2,\cdots,t_{|T|}\}$.

如图 1 所示,车队管理模块跟踪网约车的实时位置,以获取下一个时间片网约车供应量,而请求预测模块则根据历史的请求时空分布,预测未来的网约车需求分布.供需感知的深度强化学习调度策略将供需结合起来,以确定空闲网约车的调度动作,然后网约车将会朝着调度终点巡航.对于收到的新请求,调度中心分配最近的空闲网约车为其服务.当网约车被分配给一个请求后,它会首先沿着最短路径前往请求的起点去接乘客,然后驶向请求的终点以完成服务.如果一个请求在其最长等待时间内都没有被分配给网约车,则该请求将被拒绝.

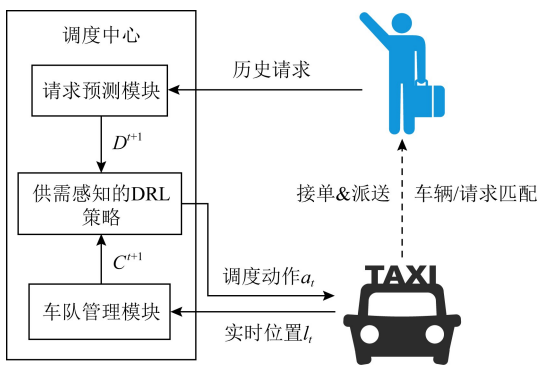


Fig. 1 Interaction between taxis and passengers with the dispatching center

图 1 网约车、乘客在调度中心下的交互

2.3 Markov 决策过程的构建

本文将网约车路径规划问题建模为 Markov 决策过程(MDP).具体地,将网约车视为与外部环境

① https://github.com/uber/h3-java

交互的智能体,并将每次路线规划看作是一次决策.采用六边形网格划分空间对动作空间进行离散化.如图 2 所示,将空闲网约车调度到目标网格视为是一个动作.

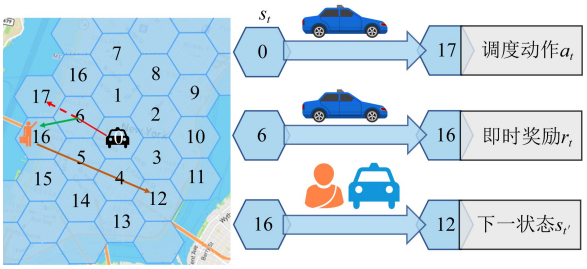


Fig. 2 An example of MDP formulation

图 2 一个 MDP 叙述的例子

MDP 包含状态、动作、奖励、回合、策略和状态-动作价值函数 6 个关键元素.

1) 状态 $s_t \in S$.在模型中的状态输入为一个五元组,$s_t=(g_i,t,\delta^t,C_i^t,D_i^t)$,元组中的元素分别表示网格下标、当前时间片的索引、全局的空闲车辆与等待中的请求数量之差,以及该网格的网约车供应量和乘客需求.其中网约车的供应定义遵从文献[13]的方法,可以通过式(2)估计:

$$C_i^t=N_{\text{idle}}^t+N_{\text{drop}}^t+N_{\text{disp}}^t, \tag{2}$$

其中,N_{idle}^t,N_{drop}^t 和 N_{disp}^t 分别表示时间 t 在网格 g_i 内的空闲车辆数、将会在 t 所在时间片内在该网格内下车的网约车数量和在时间 t 之前被调度到该网格的网约车数量.

网约车需求计算为:乘客请求为在网格 g_i 中当前等待中的请求 N_{wait}^t 和通过预测模型预测的未来可能出现的请求数量 N_{pred}^t 之和:

$$D_i^t=N_{\text{wait}}^t+N_{\text{pred}}^t, \tag{3}$$

其中,N_{wait}^t 与 N_{pred}^t 分别表示在网格 g_i 中当前等待中的请求和通过预测模型预测的未来可能出现的请求数量.

2) 动作 $a_t \in A$.动作 $a_t=\langle g_j \rangle$ 是指将空闲网约车派往某个特定的目的网格 g_j.所有可行的动作组成了动作空间,用 A 表示.A 包括当前区域的 k-阶邻居区域中需求与供应之差大于当前区域的所有区域和全局的热门区域.因此对于不同的供需状态 s,动作空间 A 也是不同的.值得注意的是,由于网约车在调度的执行过程中也可以接单,因此其并不一定能到达调度终点(中途被分配给请求).

3) 奖励 r_t.使用文献[8]中有效行驶比作为即时奖励,其定义为:

$$r_t=\begin{cases}\dfrac{d_o}{d_s}, & \text{网约车被分配给请求},\\ 0, & \text{否则},\end{cases} \tag{4}$$

其中,d_s 为调度过程的持续时间,该过程从网约车空闲开始到乘客下车结束;设 d_o 为调度过程中乘客乘坐网约车的持续时间.式(4)计算的奖励 r_t 考虑服务请求的收入和空闲巡航的成本.注意到较大的 r_t 表示网约车在短时间闲置后迅速分配给乘客,因此获得的奖励很高.如果网约车到达调度目的地时未分配任何请求,则奖励为 0.

4) 回合.在该问题设定中,一个回合是从 8:00 到 22:00 的繁忙时段.因此,时间 t 在 22:00 之后的状态为终止状态.

5) 策略 $\pi(a|s)$.策略函数表示从状态到可行动作 A 的概率分布的映射.对于确定性的策略,输出是一个特定的动作.通过深入的与环境进行交互,代理旨在学习到一个最优的策略 π^*,以最大化预期奖励.

6) 状态-动作价值函数 $Q_\pi(s,a)$.状态-动作价值函数是司机从状态 s 开始,执行调度动作 a,并且之后按照策略 π 执行调度动作,直到回合结束能得到的奖励和的期望,由于使用有效性行驶比作为奖励,因此 Q 表示的是预期累积有效行驶比,定义为

$$Q_\pi(s,a)=E_\pi\Big[\sum_{k=0}^{\infty}\gamma_{t+k+1}^{kr}\mid s_t=s,a_t=a\Big], \tag{5}$$

其中,t 表示当前时间片,考虑到短期的奖励的影响更大,估值也更准,这里加入参数 γ 作为未来奖励的折扣因子,以对长期的奖励进行衰减.

更具体地,在图 2 中给出了上述 MDP 概述的一个例子.在时刻 t,一辆处于状态 s_t 的空闲车辆被指派了一条从网格 0 到网格 17 的搜索路线,执行前往网格 17 的动作.当它行驶到网格 6 时,在网格 16 出现一个请求并且这辆车被分配给了这个请求.因此它前往该请求的位置去接乘客,然后驶向位于网格 12 的订单终点,获得对应的奖励 r_t,并到达下一状态 $s_{t'}$.

3 供需感知的深度强化学习

本节介绍一种供需感知的深度强化学习算法,称为 AS-AC 算法.

3.1 动作空间的确定

首先,通过考虑以下 2 种类型的网格来初始化可行动作空间 A:

1) 地理邻居网格.为了确保合理的调度距离,选择当前网格的邻居网格.具体来说,在实验中选择了纽约数据集的三阶邻域和海口数据集的二阶邻域内的网格.

2) 全局热门网格.在下一个时间片中预测请求数量最多的少数网格(纽约为 5 个网格,海口为 3 个网格),因为大多数请求都出现在这些网格中.

为了减少计算开销,需要从初始动作空间 A 中移除无效的动作.这是因为,如果当前网格的供需差大于调度目标网格的供需差,停留在当前网格更可能获得更高的奖励,并且可以减少调度开销. 因此,可以从动作空间中删除这些调度动作.

对于 A 中的每个网格 g_i,首先根据式(2)(3)分别计算供应和需求,然后通过如式(6)计算供需差 ρ_i:

$$\rho_i=\begin{cases}D_i-C_i, & 若\ D_i>C_i,\\ 0, & 其他.\end{cases} \tag{6}$$

最后,从动作空间 A 中移除供需差小于网约车当前网格的动作.

3.2 AC 模型

actor-critic(AC)算法结合了基于值函数的方法与基于策略梯度方法的优势,避免了低维度动作空间的限定,可扩展性更强.因此,使用 AC 模型学习得到空闲网约车的最佳调度策略,它通过调整策略网络的权重来适应动态变化的环境.不同于 DQN 采用价值函数来制订策略,AC 采用一个叫做 actor 的策略网络来直接近似策略,用于选择动作,用一个称为 critic 的价值网络来评估所选择的动作.相比基于值函数的方法,它不仅能够实现更稳定的学习过程,还适用于高维度或连续的动作空间.

本文采用了 2 种技术来提升网络的性能:1)对于 critic 网络,类似 DQN,采用原始和目标 2 个相同的价值网络来实现周期稳定更新;2)通过嵌入地理邻居和供需差来过滤无效动作使得能够调整策略以适应动态变化的动作空间.

采用小批量反向传播算法训练网络,对于 critic 价值网络,如图 3 所示,根据贝尔曼方程得到时序差分误差,以此平方作为损失函数:

$$td_{\text{error}}=r_t+\gamma V(s_{t+1};\theta'_v)-V(s_t;\theta_v), \tag{7}$$

$$L(\theta_v)=td^2_{\text{error}}, \tag{8}$$

其中,$\gamma\in[0,1]$为折扣因子,θ_v 表示价值网络的参数,θ'_v 表示目标价值网络的参数.计算损失函数之后,原始价值网络的参数 θ_v 根据损失函数的梯度更新为

$$\theta_v\leftarrow\theta_v+\alpha_c\nabla_{\theta_v}L(\theta_v), \tag{9}$$

其中,α_c 是 critic 网络的学习率.价值网络的输入为智能体的当前状态 s_t,输出为常量 $V(s_t;\theta_v)$ 表示当前状态价值.

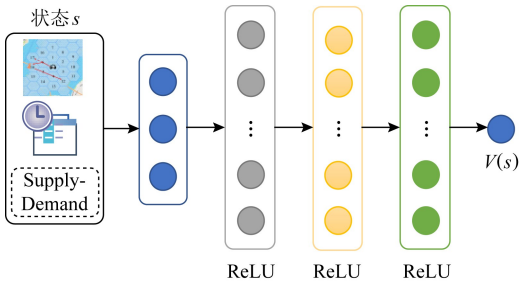


Fig. 3 The structure of critic network

图 3 Critic 价值网络结构

对于 actor 策略网络,如图 4 所示,采用第 3.3 节提到的动作采样策略替换传统 actor 网络最后的 softmax 层.为了减小动作选择的方差,引入了一个优势函数来代替 critic 网络中的原始回报,以衡量选取的动作值与所有动作平均值的好坏,用时序差分来近似优势函数,即:$A(s_t,a_t)=td_{\text{error}}$,那么策略网络的梯度为

$$\nabla_{\theta_p}J(\theta_p)=\nabla_{\theta_p}\log\pi(a_t|s_t)A(s_t,a_t), \tag{10}$$

其中,θ_p 是策略网络参数.根据计算得到的策略梯度,策略网络参数 θ_p 更新为

$$\theta_p\leftarrow\theta_p+\alpha_a\nabla_{\theta_p}J(\theta_p), \tag{11}$$

其中,α_a 是 actor 网络的学习率.策略网络的输入为智能体的当前状态 s_t,输出为数组 $Q(s_t;a_t)$ 表示当前状态下执行所有动作对应的状态价值函数.

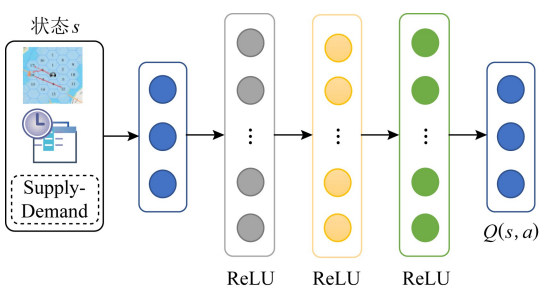


Fig. 4 The structure of Actor network

图 4 Actor 策略网络结构

AC 模型训练算法如算法 1 所示.首先,初始化一个容量为 N 的记忆库 M,并分别随机初始化 actor 和 critic 网络的权重 θ_p,θ_v 和 θ'_v.然后,通过模拟器训练模型 20 000 回合,在每回合中,分 2 步运算:1)基于历史订单数据模拟乘客的行程请求,根据网约车初始状态 s_0,在每个时间片内,采用随机策略进行调度,记录网约车的状态变化,采取的动作和

奖励,将其作为一条经验存储于记忆库 M 中,以供后续线下学习;2)随机从记忆库中采样 b 个状态转移样本,计算时序误差、损失函数和梯度,然后批量更新 actor-critic 网络参数.其中目标价值网络 θ'_v 是原始价值网络 θ_v 的一个副本,并每隔 B 个回合与 θ_v 同步一次,以保证训练过程的稳定性.AC 算法采用离线经验回放的方式来更新参数,这样能够充分利用样本,防止过拟合.

算法 1. AC 算法.

① 初始化记忆库 M 的容量为 N;
② 初始化 critic 的源网络参数 θ_v 和目标网络参数 θ'_v 为随机值;
③ 初始化 actor 网络参数 θ_p 为随机值;
④ for $m=1$ to $max\text{-}episodes$ do
⑤ 重置所有网约车为初始状态 s_0;
⑥ for $t=1$ to $|T|$ do
⑦ 对每辆空闲网约车生成并存储状态转移元组(s_t,a_t,r_t,s_{t+1})到 M;
⑧ end for
⑨ 从记忆库 M 中随机抽取小批量样本$(s_t^i,a_t^i,r_t^i,s_{t+1}^i)$;
⑩ for each $(s_t^i,a_t^i,r_t^i,s_{t+1}^i)$ do
⑪ $y_i \leftarrow r_t^i+\gamma V(s_{t+1}^i;\theta'_v)$
⑫ 计算样本的时序误差 $td_{\text{error}}^i=y_i-V(s_t^i;\theta_v)$;
⑬ 计算损失函数 $L_i=(td_{\text{error}}^i)^2$;
⑭ end for
⑮ 根据式(9)批量更新 critic 的源网络参数 θ_v;
⑯ 根据式(11)批量更新 actor 的策略网络参数 θ_p;
⑰ if m mod $B==0$ then
⑱ 更新 critic 的目标网络参数 $\theta'_v \leftarrow \theta_v$;
⑲ end if
⑳ end for

3.3 动作采样策略

在当前状态 s 下,为了得到具体的动作,基于状态-动作价值函数 $Q(s,a)$ 从 A 中选择一个动作. 然而,传统的"ε-贪婪"策略会以 $1-\varepsilon$ 的概率选择价值最大的动作,导致大部分空闲出租车都前往热门区域,造成"羊群效应",无法很好地对网约车进行协调.为了实现网约车之间的协调,采取了动作优先级采样策略,与文献[38]中基于时序差分误差的经验优先级采样策略不同,依据动作价值函数 $Q(s,a)$ 对可行性动作进行采样.具体而言,对动作价值函数 $Q(s,a)$ 做类似 softmax 处理,以确保动作被采样的概率与动作价值是单调递增的关系,同时确保最小价值的动作被采样的概率非零.具体地,第 i 个动作被采样的概率计算为

$$Pr(a_i)=p_i^\tau \Big/ \sum_{j=1}^{|A|} p_j^\tau, \tag{12}$$

其中,$p_i>0$ 是 A 中第 i 个动作的优先级,而 τ 是确定使用多少优先级的参数,当 $\tau=0$ 时,对应于统一随机采样.对于优先级 p_i,有如下 2 种变体:

1) 比例优先级.其中 $p_i=Q(s,a_i)$,它表示从 A 中根据状态价值成比例的采样动作.

2) 基于排序的优先级.其中 $p_i=\dfrac{1}{rank(a_i)}$,其中 $rank(a_i)$是 a_i 的序号,由状态-动作价值排序得到.

值得注意的是,由于基于排序的优先级对异常值不敏感,具有更好的鲁棒性,因此实验中默认选择其来制订采样策略,与对比方法进行比较,并且在实验中比较这 2 种变体.最终,根据动作采样策略从 A 中采样得到调度动作.

3.4 调度地点确定

为了实现更加细粒度的供需平衡,使用文献[16]中供需分布不匹配度来确定每个空闲网约车的具体调度位置.对于调度位置 $l\in a_t.g$,令 $x_t(l)$ 和 $y_t(l)$分别为 l 处空闲网约车的数量和预测请求数量,则 l 处的供需分配不匹配度计算为

$$\psi_t(l)=\frac{x_t(l)}{\sum\limits_{l\in a_t.g} x_t(l)}-\frac{y_t(l)}{\sum\limits_{l\in a_t.g} y_t(l)}. \tag{13}$$

对于每次调度,从目的网格 $a_t.g$ 中贪婪地选择需求供应不匹配最小的位置作为调度目的地,然后网约车沿着最短路径前往目的地.当目的网格中没有空闲网约车时,选择该网格中心作为调度目的地,因为它能保证与网格中的其他位置距离不会过远.算法 2 中详细地介绍了 AS-AC 的执行过程.

算法 2. AS-AC 算法.

输入:当前状态 s_t;

输出:一个调度动作 a_t.

① 计算源动作价值 $Q(s,a_i),\forall i=1,2,\cdots,|G|$;

② 初始化动作空间 A 为地理邻居和全局热门网格;

③ 从 A 移除无效的动作;

④ 初始化大小为$|G|$的数组 F,并设置 $F_i=1$, $\forall a_i\in A$;

⑤ 通过状态-动作价值 $Q(s,a_i)\times F$ 对动作 $a_i\in A$ 进行排序,并计算对应优先级;

⑥ 根据式(12)采样一个动作 a_t;

⑦ return a_t.

4 实验与结果

解决交通问题的强化学习方法通常需要一个动态模拟环境用于训练模型和评估算法,采用并拓展了车队管理模拟器 COMSET[①],以训练并评估调度策略.实验用 Java 实现调度策略和所有的对比方法,使用 Python3.6 和 Tensorflow1.15.0 来构建并训练模型,其中网络模型在一台装配有 Nvidia Tesla P100 GPU 和 16 G 内存的机器上训练.

4.1 环境模拟器与数据集

基于真实数据集,采用并拓展了能够模拟外部环境的 COMSET 模拟器训练 DRL 模型.该模拟器是对网约车平台如何管理网约车和处理乘客请求的整个过程进行建模.具体而言,在模拟开始时,根据输入的 OSM JSON 文件创建一个地图,并通过输入边界多边形对其进行裁剪.乘客请求从历史订单数据中读取,每一个请求对应一个历史订单记录,只保留上下车位置均在边界多边形范围内的请求,并按照上车时间的先后顺序以流的形式进入系统. 固定数量的网约车被部署在地图上的随机位置,并按照调度策略给出的搜索路线进行巡航.当请求进入系统后,控制中心为其分配最近的空闲网约车;网约车接受请求,前往请求起点接乘客,再将其送到请求终点,期间车辆都沿着最短路径行驶.更重要的,模拟器会通过历史订单数据校正每条路段的旅行速度,因此 COMSET 产生的平均旅行时间与真实数据基本一致.

本文采用纽约和海口 2 个真实网约车数据集评估模型,有关数据集的信息统计在表 2 中,值得注意的是每条订单记录包含起点与终点信息、订单类型、旅行时间、乘客数量等信息.

1) 纽约数据集[②].纽约黄色网约车订单数据包括了 2016 年 1 月到 6 月中上车和下车点均在纽约市的网约车订单,而路网数据来源于 OpenStreetMap[③],其中包含了 4 360 个节点和 9 542 条边.选取 2016 年 6 月 1 日至 21 日数据用于训练模型,6 月 22 日至 28 日数据用评估模型.

2) 海口数据集[④].海口数据来源于海口市 2017 年 5 月到 10 月的网约车订单记录,路网包含 3 298 个节点和 8 034 条边.选取 2017 年 10 月 1 日至 21 日数据用于训练,10 月 22 日至 28 日数据用于评估模型.

Table 2 Dataset Statistics

表 2 数据集统计信息

数据集	订单数量	边数	节点数	大小/GB
纽约	69 406 526	9 542	4 360	10.1
海口	12 374 094	8 034	3 298	2.28

4.2 对比算法与度量标准

通过实验对比以下 6 种算法的效果:

1) RD[29].从路网中随机选取一个节点作为目的地,选取从当前位置到目的地的最短路径作为搜索路线.

2) MCF-FM[12]. MCF-FM 根据权重采样的方法选择调度终点,并在每个时间片中对空闲司机进行统一调度.

3) FMU[10]. FMU 根据动态的节点权重进行采样,并为等待中的请求增加额外的权重.

4) Q-learing[39]. 标准表格式的 Q-learning 学习一个含有"ε-贪婪"策略的 q 表.实验中,状态简化为只包含智能体的位置和时间,并设置 $\varepsilon=0.1$.

5) DQN[31]. 状态、动作和奖励定义与本文一样,采取"ε-贪婪"策略选取 $Q(s,a)$ 值最大的动作作为调度目的地,实验中 $\varepsilon=0.1$.

6) AS-AC. 本文提出的具有动作采样策略的供需感知的执行者-评论者方法.

为了评估上述算法,统一采用 3 种度量标准:

1) 拒绝率(reject rate, RR),指被拒绝的请求数占请求总数的比例,它在所有度量标准中具有最高的优先级;

2) 巡航时间(cruising time, CT),指网约车从空闲状态到乘客上车的平均时间,由总的空闲行驶时间除以接收的请求数量得到;

① https://github.com/steve-tjiang/GISCUP-2020

② https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

③ https://www.openstreetmap.org

④ https://outreach.didichuxing.com/app-vue/HaiKou?id=999

3) 等待时间(waiting time, WT),指乘客从发出请求到司机到达上车点的平均等待时间,由总的等待时间除以请求数量.

4.3 实验参数与结果对比

由表 2 可知,纽约数据集的规模比海口数据集大很多,如果对海口数据集采用和纽约数据集相同的划分粒度,则每个分区的数据将十分稀疏.为了避免这一问题,根据数据集的规模调整时空划分的粒度.具体而言,在纽约和海口数据集上设定时间片长度分别为 5 min 和 10 min;采用 Uber H3 库将纽约和海口路网分别划分为 99 和 30 个网格.每个请求的最大生命周期 $t^*=5$ min,即乘客最多等待 5 min,如果 5 min 内没有司机接单,则该请求被视为拒绝,并从系统中移除.对于提出的方法,其设定的参数如下:actor 与 critic 网络(如图 3、图 4)均采用 3 层全连接层,每层包含 128 个隐藏单元,并采用 ReLU 激活函数.记忆库容量 $N=100\,000$,采样的批量大小 $b=64$,critic 中目标网络更新的周期 $B=1\,000$,折扣因子 $\gamma=0.9$,学习率 $\alpha_c=0.001$,$\alpha_a=0.000\,5$.

为了验证方法的鲁棒性,在不同网约车数量下进行对比实验,实验结果如表 3 所示,其中每种度量标准的最佳结果用粗体表示.

Table 3 The Results on Two Real-World Datasets

表 3 在 2 个真实数据集上的实验结果

算法	纽约				海口			
	车辆数量	RR/%	CT/s	WT/s	车辆数量	RR/%	CT/s	WT/s
RD	3 000	20.455	294.273	172.887	1 000	13.778	476.998	175.959
MCF-FM		11.589	201.736	154.601		**11.053**	**434.703**	168.295
FMU		11.448	200.119	147.247		11.751	443.969	172.081
Q-learning		12.810	212.073	149.571		11.883	446.816	171.890
DQN		11.799	203.957	148.322		11.530	443.125	169.169
AS-AC(V2)		**11.350**	**199.864**	**145.538**		11.105	435.964	**166.771**
RD	3 200	17.382	315.166	161.891	1 100	11.012	558.108	163.322
MCF-FM		8.550	226.591	146.605		7.891	508.623	152.216
FMU		8.316	224.391	138.249		8.406	515.441	156.067
Q-learning		9.750	236.636	139.827		8.679	520.044	156.454
DQN		8.748	228.908	138.579		8.305	516.032	152.355
AS-AC(V2)		**8.207**	**224.008**	**136.152**		**7.866**	**508.563**	**150.474**
RD	3 400	14.063	333.936	149.930	1 200	9.168	648.785	152.751
MCF-FM		5.925	253.864	133.695		5.744	592.969	138.305
FMU		5.699	251.736	125.263		6.221	599.675	142.935
Q-learning		7.162	164.298	128.779		6.498	604.154	143.117
DQN		6.279	257.628	125.541		6.216	599.640	143.056
AS-AC(V2)		**5.583**	**251.188**	**122.294**		**5.649**	**591.388**	**136.273**
RD	3 600	11.018	354.330	137.960	1 300	7.875	744.609	144.888
MCF-FM		3.826	284.578	114.584		4.416	685.874	127.436
FMU		3.694	283.008	109.107		4.580	687.824	130.828
Q-learning		5.171	295.916	116.500		5.043	695.467	132.281
DQN		4.499	291.128	109.023		4.995	694.182	137.858
AS-AC(V2)		**3.517**	**281.893**	**105.345**		**4.120**	**682.324**	**125.564**
RD	3 800	8.387	378.426	126.342	1 400	6.961	844.107	138.454
MCF-FM		2.317	319.335	95.761		3.532	783.201	118.868
FMU		2.238	318.244	93.262		3.627	784.489	121.673
Q-learning		3.610	330.401	102.919		4.083	791.974	123.593
DQN		3.254	328.414	94.573		3.999	790.446	126.698
AS-AC(V2)		**2.036**	**316.698**	**89.713**		**3.332**	**779.584**	**115.251**

续表 3

算法	纽约				海口			
	车辆数量	*RR*/%	*CT*/s	*WT*/s	车辆数量	*RR*/%	*CT*/s	*WT*/s
RD	4 000	6.445	408.392	117.027	1 500	6.189	944.239	133.211
MCF-FM		1.389	358.508	81.134		2.969	884.114	112.251
FMU		1.294	357.187	80.280		2.851	881.921	112.680
Q-learning		2.495	368.197	90.350		3.384	891.032	116.546
DQN		2.485	369.515	83.724		3.100	887.122	110.958
AS-AC(V2)		**1.102**	**355.576**	**78.564**		**2.600**	**877.342**	**108.230**

注:黑体数字为每种度量的最佳结果.

一般而言,网约车数量越多,可以服务越多请求,这样拒绝率(*RR*)、平均巡航时间(*CT*)以及平均等待时间(*WT*)都会降低.更具体些,可以从表 3 中得出以下 4 点结论:

1) MCF-FM 和 FMU 相比于随机调度策略 RD 在所有的度量标准上都有很大的提升.这是因为 MCF-FM 和 FMU 均根据历史订单数据为节点分配合理的权重,直观上,热门区域具有较高的权重,再根据权重采样[40]可以很好地平衡供应与需求.

2) 基于历史数据学习动作价值函数的 Q-learning 相比于 RD 也有所提升,但其提升的程度没有 MCF-FM 与 FMU 大,这是因为 Q-learning 的性能受限于确定性的状态空间,而由于交通环境的复杂性,实际产生的状态是无穷多的,所以表格式方法无法对其完整建模.

3) 虽然标准 DQN 算法也可以实现不错的效果,但是其采用的"ε-策略"会贪婪地调度车辆前往价值高的区域,这会导致大部分车辆前往热区,造成"羊群效应",而在其他区域因车辆少而乘客请求得到不到服务,直至拒绝.所以其提升效果也受限.

4) 除了在网约车数量为 1 000 的海口数据集上,提出的 AS-AC 算法在所有度量标准上均实现了最佳的效果,提升程度最大.相比于 MCF-FM, FMU 和 DQN 等已有最佳方法,在不同数据集、不同网约车数量的情况下,AS-AC 算法可以在拒绝率上最高提升 0.3 个百分点,巡航时间上减少 0.1～4.6 s,乘客等待时间减少 1.8～4.1 s.由于每天都有百万级别的订单产生,所以每一点提升都会带来巨大效益,可以大大减少能源消耗,减少交通拥堵现象,提升用户体验,提高平台收益.

4.4 动作采样策略的讨论

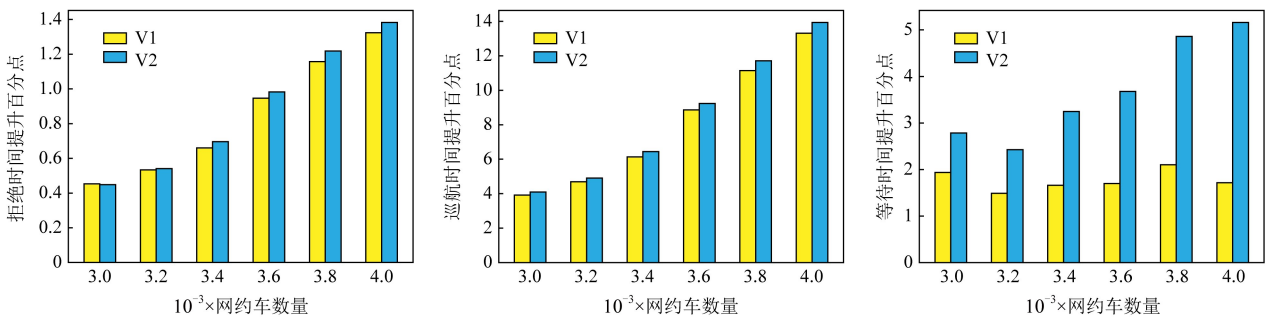
表 3 的实验结果表明了 3.3 节提出动作采样策略的有效性.同时,针对 3.3 节中提到的 2 种采样优先级变体,本节进行对比实验,以验证排序优先级的优越性.其中:

1) 采用比例优先级的 AS-AC 记为 V1;

2) 采用排序优先级的 AS-AC 记为 V2.

为了方便比较,将 2 个变体相对 DQN 的提升程度作为实验结果,如图 5 和图 6 所示.可以观察到:

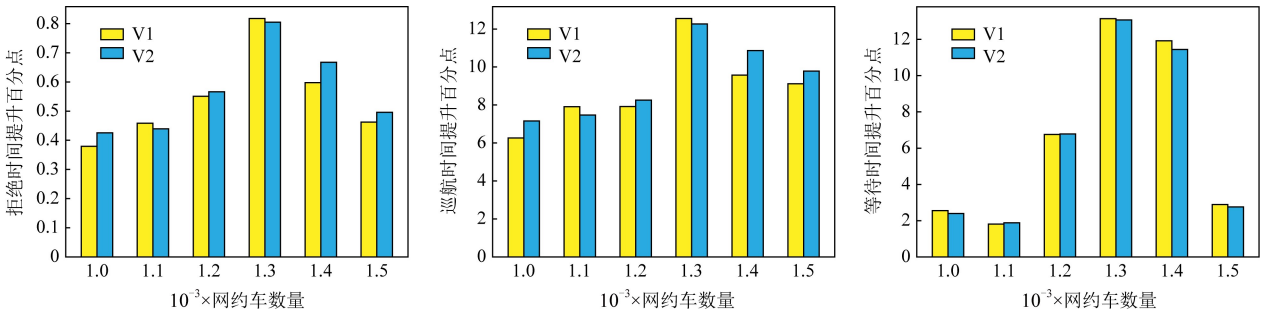
1) 总体来说,对于每种变体,相比于 DQN 的提升程度随网约车数量的增加而增大,比如,在纽约数据集上,当车辆数量为 3 000 时,DQN 的拒绝率为 11.799%,而 V1 与 V2 相比于 DQN 分别提升 0.453% 和 0.448%;当车辆数量增加到 4 000 时,DQN 的拒绝率为 2.485%,而 V1 与 V2 相比于 DQN 分别提升 1.323%和 1.383%.由此可以看出随着车辆数的



(a) 拒绝时间提升百分点 (b) 巡航时间提升百分点 (c) 等待时间提升百分点

Fig. 5 Comparison of variants on New York dataset

图 5 2 种变体在纽约数据集上的比较



(a) 拒绝时间提升百分点 (b) 巡航时间提升百分点 (c) 等待时间提升百分点

Fig. 6 Comparison of variants on Haikou dataset

图 6 2 种变体在海口数据集上的比较

增加,2 种变体能更好实现车辆之间的协调;

2) 相比于变体 V1,变体 V2 达到更好的效果.这是因为基于排序的优先级对异常值不敏感,具有更好的鲁棒性,因此其更加有效,并选取其作为默认的采样策略优先级.

5 结 论

本文提出了一种基于供需感知的深度强化学习模型用于网约车调度.该模型通过定义合适的智能体、状态、动作和奖励,将网约车路径规划问题转化为 Markov 决策过程,然后提出了一个具有动作采样策略的执行者-评论者网络结构(AS-AC),以学习最佳的空闲网约车调度策略.实验结果表明,本文提出的供需感知的深度强化学习方法在不同数据集上均取得了比对比方法更好的表现,验证了执行者-评论者网络在协调网约车上的有效性,此外,通过对不同的动作采样策略进行对比试验,证明基于排序优先级的采样策略比采用比例优先级的采样策略效果更优.

在下一步工作中,可以采用多智能体强化学习的方法,以及考虑更复杂的实时路况信息,让模型学到更优的调度策略,以更好地平衡城市中的供需分布.

作者贡献声明:郑渤龙提出了算法思路和实验方案;明岭峰和胡琦负责完成实验并撰写论文;方一向、郑凯和李国徽提出指导意见并修改论文.

参 考 文 献

[1] Beijing Xiaoju Technology Co, Ltd. DiDi [OL]. [2021-11-19]. https://www.didiglobal.com/(in Chinese)
北京小桔科技有限公司. 滴滴[OL]. [2021-11-19]. https://www.didiglobal.com/

[2] Uber. Uber in China [OL]. [2021-11-19]. https://www.uber.com (in Chinese)
(Uber. 优步中国[OL]. [2021-11-19]. https://www.uber.com)

[3] Zhou Ming, Jin Jiarui, Zhang Weinan, et al. Multi-agent reinforcement learning for order-dispatching via order-vehicle distribution matching[C] //Proc of the 28th ACM Int Conf on Information and Knowledge Management. New York: ACM, 2019: 2645-2653

[4] Bai Lei, Yao Lina, Kanhere Salil S, et al. STG2Seq: Spatial-temporal graph to sequence model for multi-step passenger demand forecasting[C] //Proc of the 28th Int Joint Conf on Artificial Intelligence. San Francisco, CA: Morgan Kaufmann, 2019: 1981-1987

[5] Bai Lei, Yao Lina, Li Can, et al. Adaptive graph convolutional recurrent network for traffic forecasting[C] //Proc of the 34th Conf on Neural Information Processing Systems. Cambridge: MIT Press, 2020

[6] Yu Bing, Yin Haoteng, Zhu Zhangxing. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting[C] //Proc of the 27th Int Joint Conf on Artificial Intelligence. San Francisco, CA: Morgan Kaufmann, 2018: 3634-3640

[7] Li Minne, Qin Zhiwei, Jiao Yan, et al. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning[C] //Proc of the 28th The World Wide Web Conf. New York: ACM, 2019: 983-994

[8] Tang Xiaocheng, Qin Zhiwei, Zhang Fan, et al. A deep value-network based approach for multi-driver order dispatching[C] //Proc of the 25th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining. New York: ACM, 2019: 1780-1790

[9] Xu Zhe, Li Zhixin, Guan Qingwen, et al. Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach[C] //Proc of the 24th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining. New York: ACM, 2018: 905-913

[10] Li Wenli. A fleet manager that brings agents closer to resources: GIS Cup[C] //Proc of the 28th Int Conf on Advances in Geographic Information Systems. New York: ACM, 2020: 655-658

[11] Lin Kaixiang, Zhao Renyu, Xu Zhe, et al. Efficient large-scale fleet management via multi-agent deep reinforcement learning[C] //Proc of the 24th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining. New York: ACM, 2018: 1774-1783

[12] Ming Lingfeng, Hu Qi, Dong Ming, et al. An effective fleet management strategy for collaborative spatio-temporal searching: GIS Cup[C] //Proc of the 28th Int Conf on Advances in Geographic Information Systems. New York: ACM. 2020: 651-654

[13] Liu Zhidan, Li Jiangzhou, Wu Kaishun. Context-aware taxi dispatching at city-scale using deep reinforcement learning [J]. IEEE Transactions on Intelligent Transportation Systems, 2020, DOI: 10.1109/TITS.2020.3030252

[14] Gao Yong, Jiang Dan, Xu Yan. Optimize taxi driving strategies based on reinforcement learning[J]. International Journal of Geographical Information Science, 2018, 32(8): 1677-1696

[15] Miao Fei, Han Shuo, Lin Shan, et al. Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach [J]. IEEE Transactions on Automation Science and Engineering, 2016, 13(2): 463-478

[16] Oda T, Joe-Wong C. MOVI: A model-free approach to dynamic fleet management[C] //Proc of IEEE INFOCOM Conf on Computer Communications. Piscataway, NJ: IEEE, 2018: 2708-2716

[17] Jin Jiarui, Zhou Ming, Zhang Weinan, et al. Coride: Joint order dispatching and fleet management for multi-scale ride-hailing platforms[C] //Proc of the 28th ACM Int Conf on Information and Knowledge Management. New York: ACM, 2019: 1983-1992

[18] Xu Zhe, Men Chang, Li Peng, et al. When recommender systems meet fleet management: Practical study in online driver repositioning system[C] //Proc of 29th The World Wide Web Conf. New York: ACM, 2020: 2220-2229

[19] Duan Lu, Zhan Yang, Hu Haoyuan, et al. Efficiently solving the practical vehicle routing problem: A novel joint learning approach[C] //Proc of the 26th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining. New York: ACM, 2020: 3054-3063

[20] Vazifeh M M, Santi P, Resta G, et al. Addressing the minimum fleet problemin on-demand urban mobility[J]. Nature, 2018, 557(7706): 534-538

[21] Bertsimas D, Jaillet P, Martin S. Online vehicle routing: The edge of optimization in large-scale applications[J]. Operations Research, 2019, 67(1): 143-162

[22] Braverman A, Dai J G, Liu X, et al. Empty-car routing in ridesharing systems[J]. Operations Research, 2019, 67(5): 1437-1452

[23] Qu Meng, Zhu Hengshu, Liu Junming, et al. A cost-effective recommender system for taxi drivers[C]//Proc of the 20th ACM SIGKDD Int Conf on Knowledge discovery and data mining. New York: ACM, 2014: 45-54

[24] Zheng Bolong, Hu Qi, Ming Lingfeng, et al. SOUP: Spatial-temporal demand forecasting and competitive supply[J]. IEEE Transactions on Knowledge and Data Engineering, 2021, DOI: 10.1109/TKDE.2021.3110778

[25] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553): 436-444

[26] Kaelbling L P, Littman M L, Moore A W. Reinforcement learning: A survey[J]. Journal of Artificial Intelligence Research, 1996, 4: 237-285

[27] Silver D, Huang A, Maddison C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. Nature, 2016, 529(7587): 484-489

[28] Kaiser L, Babaeizadeh M, Milos P, et al. Model-based reinforcement learning for atari[C] //Proc of the 8th Int Conf on Learning Representations. Piscataway, NJ: IEEE, 2020: 1-28

[29] Ye Deheng, Liu Zhao, Sun Mingfei, et al. Mastering complex control in MOBA games with deep reinforcement learning [C] //Proc of the AAAI Conf on Artificial Intelligence. Menlo Park: AAAI, 2020: 6672-6679

[30] Wachi A. Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving[C] //Proc of the 28th Int Joint Conf on Artificial Intelligence. San Francisco: Morgan Kaufmann, 2019: 6006-6012

[31] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529-533

[32] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double Q-learning[C] //Proc of the AAAI Conf on Artificial Intelligence. Menlo Park: AAAI, 2016: 2094-2100

[33] Sutton R S, McAllester D A, Singh S P, et al. Policy gradient methods for reinforcement learning with function approximation[C] //Proc of Advances in Neural Information Processing Systems. Cambridge: MIT Press, 1999: 1057-1063

[34] He Suining, Shin K G. Spatio-temporal capsule-based reinforcement learning for mobility-on-demand network coordination[C] //Proc of Int World Wide Web Conf. New York: ACM, 2019: 2806-2813

[35] Wang Zhaodong, Qin Zhiwei, Tang Xiaocheng, et al. Deep reinforcement learning with knowledge transfer for online rides order dispatching[C] //Proc of Int Conf on Data Mining. Piscataway, NJ: IEEE, 2018: 617-626

[36] Mnih V, Badia A P, Mirza M, et al. AsynchronousMethods for Deep Reinforcement Learning[C] //Proc of the Int Conf on Machine Learning. New York: ACM, 2016: 1928-1937

[37] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[C] //Proc of the 4th Int Conf on Learning Representations. Piscataway, NJ: IEEE, 2016: 1-14

[38] Schaul T, Quan J, Antonoglou I, et al. Prioritized experience replay [C] //Proc of the 4th Int Conf on Learning Representations. Piscataway, NJ: IEEE, 2016: 1-21

[39] Sutton R S, Barto A G. Reinforcement learning: An introduction[G]. Cambridge: MIT Press, 2018

[40] Hu Qi, Ming Lingfeng, Tong Chengdong, et al. An effective partitioning approach for competitive spatial-temporal searching (GIS Cup)[C] //Proc of the 27th ACM SIGSPATIAL Int Conf on Advances in Geographic Information Systems. New York: ACM, 2019: 620-623

Zheng Bolong, born in 1989. PhD. His main research interests include spatial-temporal data, time series data and high dimensional data management.

郑渤龙,1989 年生.博士.主要研究方向包括时空数据、时序数据和高维数据管理.

Ming Lingfeng, born in 1996. Master candidate. His main research interests include spatial-temporal data management and mining.

明岭峰,1996 年生.硕士研究生.主要研究方向包括时空数据管理与挖掘.

Hu Qi, born in 1997. Master candidate. His main research interests include spatial-temporal data management and mining.

胡　琦,1997 年生.硕士研究生.主要研究方向包括时空数据管理与挖掘.

Fang Yixiang, born in 1988. PhD. His main research interests include big data management and mining.

方一向,1988 年生.博士.主要研究方向包括大数据管理和挖掘.

Zheng Kai, born in 1983. PhD. His main research interests include spatial-temporal data and time series data management.

郑　凯,1983 年生.博士.主要研究方向包括时空数据、时序数据管理.

Li Guohui, born in 1973. PhD. His main research interests include data engineering and real time computing.

李国徽,1973 年生.博士.主要研究方向包括数据工程与实时计算.