DOI:10.19651/j.cnki.emt.2415290

基于改进型 TD3 算法的车载边缘计算任务卸载决策*

李 亚 王卫岗 张 原 刘瑞鹏
(河南理工大学物理与电子信息学院 焦作 454000)

摘 要：为满足复杂车辆任务在时延、能耗和计算性能方面的要求,同时减少网络资源的竞争和消耗,设计了一种基于车载边缘计算(VEC)的任务卸载策略,以最小化任务处理延迟和能源消耗之间平衡的长期成本为目标,将车联网中的任务卸载问题建模为马尔可夫决策过程(MDP),提出了在传统双延时深度确定性策略梯度(TD3)的基础上,利用长短期记忆网络(LSTM)来逼近策略函数和价值函数,将系统状态进行归一化处理以加速网络收敛并增强训练稳定性的改进算法(LN-TD3)。仿真结果表明,LN-TD3 性能与全部本地计算和全部卸载计算相比提高了两倍以上;收敛速度上与深度确定性策略梯度 DDPG、TD3 相比提高了约 20%。

关键词：车载边缘计算;TD3 算法;任务卸载;深度强化学习;马尔可夫决策过程
中图分类号：TN929 **文献标识码**：A **国家标准学科分类代码**：510.5015

Vehicle edge computing task offloading decision based on improved TD3 algorithm

Li Ya Wang Weigang Zhang Yuan Liu Ruipeng
(Institute of Physics and Electronic Information, Henan Polytechnic University, Jiaozuo 454000, China)

Abstract: A task offloading strategy based on Vehicle Edge Computing (VEC) is designed to meet the requirements of complex vehicular tasks in terms of latency, energy consumption, and computational performance, while reducing network resource competition and consumption. The goal is to minimize the long-term cost balancing between task processing latency and energy consumption. The task offloading problem in vehicular networks is modeled as a Markov Decision Process (MDP). An improved algorithm, named LN-TD3, is proposed building upon the traditional Twin Delayed Deep Deterministic Policy Gradient (TD3). This improvement incorporates Long Short-Term Memory (LSTM) networks to approximate the policy and value functions. The system state is normalized to accelerate network convergence and enhance training stability. Simulation results demonstrate that LN-TD3 outperforms both fully local computation and fully offloaded computation by more than two times. In terms of convergence speed, LN-TD3 exhibits approximately a 20% improvement compared to DDPG and TD3.

Keywords: VEC;TD3 algorithm;task offloading;deep reinforcement learning;markov decision process

0 引 言

随着物联网技术的快速发展,智能汽车日益普及,进一步推动了自动驾驶、增强现实、图像辅助导航、在线游戏、社交媒体等新型车载应用的激增[1-2]。这类应用具有计算密集型特征,并对延迟敏感,因而需要大规模的计算和存储资源进行高效处理。这将给车辆用户(vehicle user rquipment, VUE)的计算资源带来巨大的压力。处理新型车载应用程序所需的计算能力将很快耗尽 VUE 的车载资源[3],研究发现,VUE 大约需要 10^6 DMIPS 的计算能力来处理实时交通状况并在 100 ms 延迟约束下实现自动转向,然而 VUE 的车载资源无法满足新型车载应用的超低延迟和超高可靠性要求[4]。车载边缘计算(vehicle edge computing, VEC)是移动边缘计算(mobile edge computing, MEC)[5]在车联网场景中的应用,作为改善这种情况的有前景的解决方案,受到了广泛关注[6]。通过对资源要求较高的任务卸载到路边单元(road side unit, RSU)中连接的 MEC 服务器中计算,可以显著降低车载应用程序的执行延迟和能耗[7],同时可以节省核心网的网络带宽,降低网络拥塞的风险。

收稿日期:2024-01-05
*基金项目:中层大气和全球环境探测重点实验室开放课题(LAGEO-2022-02)项目资助

任务卸载是移动边缘计算的核心技术之一[8-9]。由于数据传输和远程任务计算会产生额外的能源和时间消耗,将计算任务卸载到边缘服务器并不总能带来好处[10]。一个关键的技术挑战是在做出卸载决策时平衡计算和通信的总体成本。Han 等[11]提出一个联合目标优化问题,通过基于启发式算法做出卸载决策来最小化延迟和能耗。Zhang 等[12]通过建立基于软件定义网络的软件架构,研究了车联网中的延迟最小化问题,并提出了一种近似计算卸载方案,以在最大允许延迟约束下最小化所有计算任务的总处理延迟。除了二进制卸载之外,部分卸载也得到了广泛的研究。Dai 等[13]提出了一种部分计算卸载方案,将车辆任务分为两部分并作为本地计算和边缘计算进行处理,以有效提高车辆性能。You 等[14]讨论了时分多址系统中的部分卸载问题,并定义了基于阈值的最优资源分配策略。上述工作从自身需求出发制定了任务卸载策略,然而,上述文献中考虑的优化算法都是基于将整个系统模型的信息视为已知条件。由于车辆位置不断变化,在真实的车联网环境很难获得完整、准确的环境参数。因此,上述优化方法在真实的车载边缘计算环境中并不完全有效或者需要巨大的成本。

深度强化学习(deep reinforcement learning, DRL)是复杂决策问题的一种有前途的解决方案[15]。通过 DRL 可以直接与环境进行交互来学习最佳卸载策略,并且 DRL 可以充分利用深度神经网络(deep neural network, DNN)强大的表示能力,即使在具有巨大状态和行动空间的复杂问题中,也可以充分近似最佳卸载策略。Li 等[16]将卸载问题建模为马尔可夫决策过程,并设计了基于深度 Q 网络(deep Q network, DQN)的卸载策略来动态调整卸载比率,以提高系统性能。为了实现车辆用户的任务计算延迟、处理速率和能耗之间的最佳平衡,针对车联网的边缘计算环境,Zhao 等[17]提出了一种基于 DQN 的计算任务分配和卸载算法。然而,当动作空间连续时,Q 学习(Q-learning)和 DQN 算法不再适用。为了应用 DRL 方法,许多现有工作对连续动作空间进行离散化,这可能会影响优化结果。Lu 等[18]提出了一种基于深度确定性策略梯度(deep deterministic policy gradient, DDPG)的改进算法,解决 MEC 环境下服务延迟、能耗和任务成功率的联合优化问题,有效提升了用户体验质量。

基于以上总结和分析,本文发现传统的 VEC 任务卸载决策是由车辆以分布式方式独立做出,因此 MEC 服务器以先到先服务的方式对任务进行处理,资源利用率低;其次车辆高速移动会带来时变的无线信道条件;最后二进制任务卸载决策仅适用于不可分割的任务,对于可分割的任务则需要考虑使用部分卸载和最佳卸载比例。因此,本文提出一种基于 DRL 的适用于 VEC 的中央卸载决策算法,即融合长短期记忆网络(long short term memory, LSTM)和归一化状态空间的双延时深度确定性策略梯度(twin delayed deterministic policy gradient offload decision with LSTM and Normalization, LN-TD3)卸载决策算法,用于训练神经网络模型并获得问题的最优决策。

LN-TD3 通过归一化状态空间作为输入改进了 TD3,并将 LSTM 结构引入到演员-评论家(Actor-Critic)网络中。LSTM 将历史状态信息添加到马尔可夫决策系统中,提升了训练效果。基于 Pytorch 平台进行实验仿真并验证算法的有效性。

1　系统模型

如图 1 所示,本文考虑一个由 RSU 和覆盖范围内的 N 辆沿高速公路快速行驶的 VUE 组成的 VEC 网络。由于分布式的车辆决策很难实现系统最优[19],因此选择使用 MEC 服务器对 RSU 覆盖范围内车辆产生的任务进行卸载决策,并向 RSU 发送卸载请求,由其对应的 MEC 服务器进行卸载计算,同时,车辆的移动性会产生时变的信道条件。用 $N=\{1,2,\cdots,n\}$ 表示车辆集合,每辆车随机生成计算任务,定义为 $T_n=\{D_n,C_n,T_n^{\max}\}$,其中 D_n 表示任务的数据规模(单位为 bit),C_n 表示完成任务需要的 CPU 周期数,$T_n^{\max}$ 表示任务最大容忍延迟。

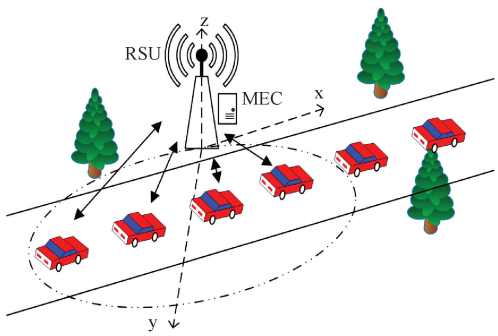

图 1　系统模型

1.1　任务卸载模型

本文提出一种由 MEC 服务器执行的集中式决策方案,将 MEC 服务器的决策时间 T 分割成等长的时隙,在任意时隙 t 内,MEC 服务器能够根据任务特征和计算资源完成对任务请求的卸载决策。MEC 服务器可以在决策时间 T 内从总共 S 个任务中选择完成 M 个任务请求,使 VEC 系统最优。

假设任务可以被分解为任意比例,那么任务卸载比例 η 可以用来描述卸载决策,在本地计算模式下,任务完全在车辆电子控制单元(electronic control unit, ECU)上计算。因此,本地计算延迟 T_{local} 和本地计算能量消耗 E_{local} 如式(1)、(2)所示。

$$T_{local}=\frac{(1-\eta)\cdot D_i(t)\cdot C_i}{f_{vehicle}} \tag{1}$$

$$E_{local}=\frac{(1-\eta)\cdot D_i(t)\cdot C_i}{f_{vehicle}}P_{vehicle} \tag{2}$$

其中,$D_i(t)$ 表示第 t 时隙的第 i 辆车产生的任务数据

大小，C_i 是车辆 i 计算 1bit 数据所需要的 CPU 周期数，$f_{vehicle}$ 是车辆 ECU 的频率，$P_{vehicle}$ 是车辆计算消耗的功率。由于是本地计算，所以此时 η 为 0。

1.2　距离模型

定义 $P_i(t)$ 为车辆 i 在时隙 t 的位置，$d_{ix}(t)$，d_y 为车辆 i 与 RSU 在时隙 t 沿 x 轴和 y 轴的距离，因此 $P_i(t)$ 可以表示为 $(d_{ix},d_y,0)$，其中 d_y 为固定值，$d_{ix}(t)$ 可以表示为：

$$d_{ix}(t)=d_{i0}+vt \tag{3}$$

其中，d_{i0} 为车辆在 x 轴的初始位置。

设 RSU 高度为 H_r，因此 RSU 的位置坐标可以表示为 $P_r=(0,0,H_r)$。所以车辆 i 与 RSU 在时隙 t 之间的距离可以表示为：

$$d_i(t)=\| P_i(t)-P_r \| \tag{4}$$

1.3　通信模型

设车辆 i 在时隙 t 的传输速率表示为 $tr_i(t)$。根据香农定理，任务数据传输速率 $tr_i(t)$ 可以表示为：

$$tr_i(t)=B\log_2\left(1+\frac{p_0\cdot g_i(t)\cdot(d_i(t))^{-\alpha}}{\sigma^2}\right) \tag{5}$$

其中，B 是传输带宽，p_0 是每辆车的发送功率，$g_i(t)$ 是车辆 i 在时隙 t 的信道增益，α 是路损因子，σ^2 是噪声功率。

使用自相关回归模型表示 $g_i(t)$ 和 $g_i(t+1)$ 之间的关系：

$$g_i(t)=\rho_i g_i(t-1)+e(t)\sqrt{1-\rho_i^2} \tag{6}$$

其中，ρ_i 表示连续时隙之间的归一化信道系数，$\boldsymbol{e(t)}$ 是服从复高斯分布的误差向量，与 $g_i(t)$ 有关。根据 Jake 的衰落谱，$\rho_i=J_0(2\pi f_d^i t)$，其中，J_0 是第一类零阶贝塞尔函数，$f_d^i t$ 为车辆 i 的多普勒频率，可以表示为：

$$f_d^i t=\frac{v}{\Lambda}\cos\theta \tag{7}$$

其中，v 是 VUE 移动速度，Λ 是波长，θ 为移动方向与上行通信方向 $P_r-P_i(t)$ 之间的夹角，因此 $\cos\theta$ 可以表示为：

$$\cos\theta=\frac{x_0\cdot(P_r-P_i(t))}{\| P_r-P_i(t) \|} \tag{8}$$

1.4　任务传输模型

车辆 i 上传部分本地任务的传输延迟 $T_{\mathrm{tr}}(t)$ 和能耗 $E_{\mathrm{tr}}(t)$ 为：

$$T_{\mathrm{tr}}(t)=\frac{\eta D_i(t)}{tr_i(t)} \tag{9}$$

$$E_{\mathrm{tr}}(t)=\frac{\eta D_i(t)}{tr_i(t)}\cdot P_{\mathrm{tr}} \tag{10}$$

其中，P_{tr} 为上行传输消耗的功率单位，当 $\eta=1$ 时，式(9)和(10)给出了全部卸载时上行传输的延迟和能耗，与上行传输相比，下行传输的延迟和能耗远小于与上行传输，因此往往忽略不计。

1.5　MEC 计算模型

任务上传到 MEC 后调度器分配相应资源用于计算任务，计算延时 T_{mec} 和功耗 E_{mec} 可以表示为：

$$T_{mec}=\frac{\eta\cdot D_i(t)\cdot C_i}{f_{mec}} \tag{11}$$

$$E_{mec}=\frac{\eta\cdot D_i(t)\cdot C_i}{f_{mec}} \tag{12}$$

部分卸载的延迟 $T_{partial}$ 和能耗 $E_{partial}$ 可以表示为：

$$T_{partial}=\max\{T_{local},T_{tr}+T_{mec}\} \tag{13}$$

$$E_{partial}=E_{local}+E_{tr}+E_{mec} \tag{14}$$

1.6　问题描述

本文采用任务处理延时与系统能耗的加权和来定义系统的系统开销。将问题表述为最优卸载决策问题，目标是通过 LN-TD3 算法最小化 VEC 产生的系统开销：

$$\begin{aligned}
&\min cost(t)=\sum_{t=1}^{T}C(t)\cdot(\lambda_1(t)\cdot T_{partial}+\lambda_2(t)\cdot E_{partial})\\
&S.T\\
&C1:\lambda_1(t)+\lambda_2(t)=1,\forall\lambda_1(t),\lambda_2(t)\subset[0,1],\\
&C2:\max\{T_{local},T_{tr}+T_{mec}\}\leqslant T^{\max},\\
&C3:C(t)=\sum_{j=1}^{J}C(t)=1,C_j(t)\in\{0,1\}\\
&C4:\sum_{t=1}^{T}C(t)\cdot D_j(t)\leqslant F_{MEC}\\
&C5:\eta(t)\in[0,1]
\end{aligned} \tag{15}$$

式中：cost 是系统开销，$\lambda_1(t)$ 是时间延时权重，$\lambda_2(t)$ 是能量消耗权重，约束 $C1$ 表示延迟与能量消耗权重的线性和为 1。约束 $C2$ 要求任务计算延迟不能大于任务的最大容忍延迟。$C(t)$ 是第 j 个任务是否被卸载的标志。约束 $C3$ 表明 MEC 服务器在每个时隙 t 内只能决定一个任务。F_{MEC} 是 MEC 服务器的最大计算能力。约束 $C4$ 表明任务所需的总计算能力不能超过 MEC 服务器的最大计算能力。本文的目标是优化变量 $\eta(t)$ 和 $C(t)$ 以使系统开销 $cost(t)$ 最小。

2　基于 DRL 的任务卸载决策算法

与传统的分布式卸载决策不同，LN-TD3 算法采用集中决策模式，同步考虑任务特征和网络状态。借助丰富的环境信息，可以实现对任务计算请求的有效动态决策。主要工作包括马尔可夫卸载决策模型和基于 LN-TD3 算法的最优卸载决策。

2.1　马尔可夫卸载决策模型

由于车联网中车辆密度的随机性和网络环境的动态性，用传统的数学方法解决该问题的计算复杂度将会极高[20]。为了应对这一挑战，本文通过 MDP 模型对问题进行建模，并应用 DRL 技术来解决该问题。MDP 模型一般用 5 元组（$\boldsymbol{S},\boldsymbol{A},\boldsymbol{P},\boldsymbol{R},\gamma$）来表示。其中，$\boldsymbol{S}$ 代表系统状态，

$\boldsymbol{A}$ 代表系统动作。$\boldsymbol{P}=p(\boldsymbol{s}_{t+1}\mid\boldsymbol{s}_t,\boldsymbol{a}_t)$ 是状态转移概率,但在本文的问题中 $\boldsymbol{P}$ 是未知的。$\boldsymbol{R}(\boldsymbol{s}_t,\boldsymbol{a}_t)$ 决定了在状态 $\boldsymbol{s}_t$ 下执行动作 $\boldsymbol{a}_t$ 所获得的瞬时奖励。马尔可夫决策过程的目标是为每个状态 $\boldsymbol{s}_t$ 确定最优策略 $\boldsymbol{\pi}_{s_t}$,以最大化长期奖励。这个长期奖励是由未来折扣奖励的总和决定的,如式(16)所示,其中 γ 为折扣因子。

$$\boldsymbol{R}_t=\sum_{i=t}^{T}\gamma^{i-t}\boldsymbol{r}(\boldsymbol{s}_i,\boldsymbol{a}_i) \tag{16}$$

根据上述定义,针对该问题的 MDP 模型可以表述如下:

状态:在时隙 t,系统的 Agent 观察车联网的动态环境,获取 MEC 服务器和车辆的各种状态;这些状态会影响卸载决策。

$$\boldsymbol{s}(t)=[F_{remain}(t),q_i(t),D_i(t),flag_i(t)] \tag{17}$$

其中,$q_i(t),D_i(t),flag_i(t)$ 分别表示车辆 i 在时隙 t 的位置坐标、生成任务的数据大小和遮挡标志。$F_{remain}(t)$ 表示 MEC 服务器在时隙 t 的剩余可用资源。在神经网络的训练过程中,任务数据的取值范围过大,会导致训练速度变慢。此外,不同任务类型取值范围的差异和不确定性会导致整个系统的不稳定和收敛性差。因此,有必要对 $\boldsymbol{s}(t)$ 的取值范围进行归一化。那么归一化状态可以表示为:

$$\hat{\boldsymbol{s}}(t)=\left[\frac{F_{remain}(t)}{F_{MEC}},\frac{q_i(t)}{L},\frac{D_i(t)}{D_{sum}},flag_i(t)\right] \tag{18}$$

其中,F_{MEC} 是 MEC 服务器最大可用资源,L 是场地范围内坐标最大值,D_{sum} 是当前时刻参与卸载的任务的总和。

动作:由于 DRL 的目的是根据当前状态选择进行任务卸载的车辆和最佳卸载比例,因此将时隙 t 的系统动作定义为:

$$\boldsymbol{a}(t)=[k(t),\eta(t)] \tag{19}$$

其中,$k(t)$ 表示在时隙 t 选择哪一辆车进行任务卸载,$\eta(t)$ 表示在时隙 t 任务卸载的比例。

奖励:在时隙 t,Agent 执行动作 $\boldsymbol{a}(t)$ 后获得奖励 $\boldsymbol{r}(t)$,DRL 模型的训练过程是否收敛取决于奖励函数,LN-TD3 算法的奖励函数为标函数式(15),因此,Agent 需要最小系统开销。奖励函数可表示为:

$$\boldsymbol{r}(t)=\boldsymbol{r}(\boldsymbol{s}(t),\boldsymbol{a}(t))=-cost(t) \tag{20}$$

2.2　LN-TD3 算法实现

LN-TD3 算法采用了 Actor-Critic 网络,Actor 的作用是定义参数化策略并根据观察到的环境状态生成动作,Critic 负责通过处理从环境中获得的奖励来评价当前策略的好坏。将 LSTM 结构引入到 Actor-Critic 网络中,利用记忆推理来提取车辆任务和环境信息[21],并通过观察状态空间和综合分析数据特征来进行有效的网络学习。

LN-TD3 的具体流程如图 2 所示,在时隙 t,Agent 观察车联网的实时状态 $\boldsymbol{s}(t)$。通过 LN-TD3 算法得到最优动作 $\boldsymbol{a}(t)$,LN-TD3 算法的 DRL 模型有两个神经网络,包括主网络和目标网络,并且都包括一个 Actor 网络和两个 Critic 网络用于逼近策略函数和价值函数。主网络中的 Actor 网络和 Critic 网络的参数表示为 $\boldsymbol{\phi},\boldsymbol{\theta}_1$ 和 $\boldsymbol{\theta}_2$。目标网络中的 Actor 网络和 Critic 网络的参数表示为 $\boldsymbol{\phi}',\boldsymbol{\theta}'_1,\boldsymbol{\theta}'_2$。Actor 网络 $\boldsymbol{\pi}_{\phi}$ 用于生成动作 $\boldsymbol{a}(t)$,即:

$$\boldsymbol{a}(t)=\boldsymbol{\pi}_{\phi}(\boldsymbol{s})+\epsilon,\epsilon\sim\mathcal{N}(0,\mu) \tag{21}$$

其中,ϵ 是动作噪声,可以增加 DRL 的探索程度,ϵ 服从均值为 0,方差为 μ 的正态分布。

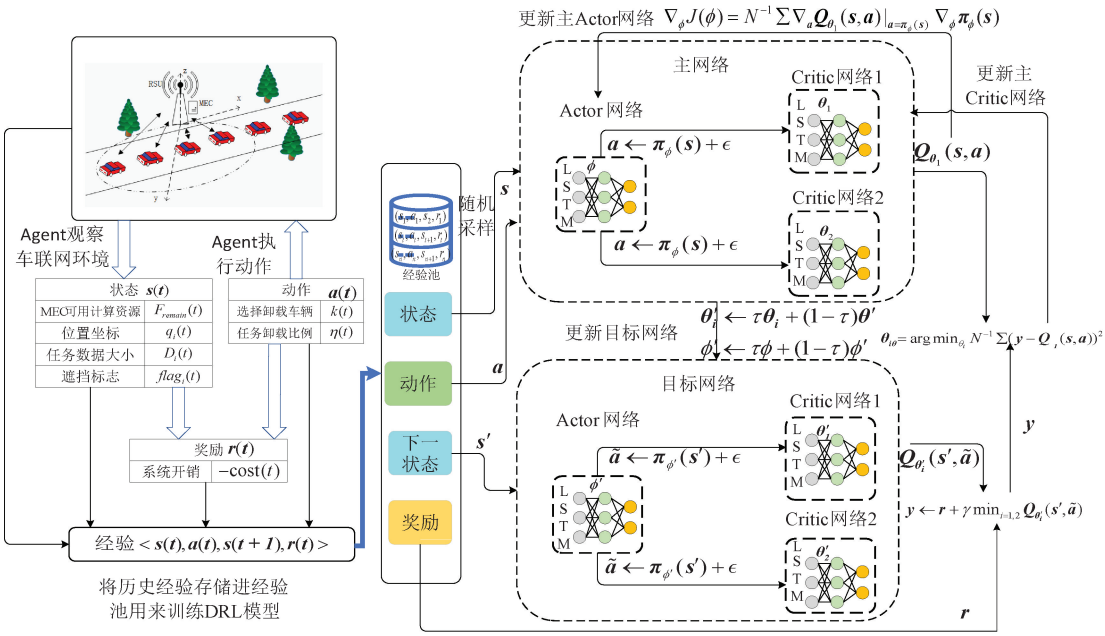

图 2　LN-TD3 算法流程

Agent 执行动作 $\boldsymbol{a}(t)$ 后，系统状态从 $\boldsymbol{s}(t)$ 更新为 $\boldsymbol{s}(t+1)$，并获得在当前状态执行当前动作的奖励值 $\boldsymbol{r}(t)$，然后将 $\boldsymbol{s}(t),\boldsymbol{a}(t),\boldsymbol{s}(t+1),\boldsymbol{r}(t)$ 存储进经验池，用于更新 LN-TD3 算法的网络参数。LN-TD3 使用确定性策略梯度来更新主 Actor 网络的参数。确定性策略梯度为：

$$\nabla_{\phi}J(\boldsymbol{\phi})=N^{-1}\sum\nabla_{a}\boldsymbol{Q}_{\theta_1}(\boldsymbol{s},\boldsymbol{a})\mid_{a=\pi_{\phi}(s)}\nabla_{\phi}\boldsymbol{\pi}_{\phi}(\boldsymbol{s}) \tag{22}$$

其中，$\boldsymbol{Q}_{\theta_1}$ 是主网络中的第一个 Critic 网络，用于近似价值函数。$\boldsymbol{\pi}_{\phi}$ 是主网络中 Actor 网络，用于近似卸载策略。

主 Critic 网络参数可用下面公式进行更新：

$$\boldsymbol{\theta}_i=\operatorname{argmin}_{\theta_i}N^{-1}\sum(\boldsymbol{y}-\boldsymbol{Q}_{\theta_1}(\boldsymbol{s},\boldsymbol{a}))^2 \tag{23}$$

其中，$\boldsymbol{Q}_{\theta_1}$ 是当前 Q 值，$\boldsymbol{y}$ 是目标网络中的目标 Q 值，受深度双 Q 学习 $\boldsymbol{s}'$ 的启发，LN-TD3 使用截断双 Q 学习来防止对 Q 值的过高估计，即使用两个 Critic 网络对 Q 值函数进行估计，并选择两者中较小的 Q 值作为估计值，$\boldsymbol{y}$ 的计算公式为：

$$\boldsymbol{y}=\boldsymbol{r}+\gamma\min_{i=1,2}\boldsymbol{Q}_{\theta_i'}(\boldsymbol{s}',\tilde{\boldsymbol{a}}) \tag{24}$$

其中，$\boldsymbol{r}$ 是奖励，γ 是在范围[0,1]之间的折扣因子，$\tilde{\boldsymbol{a}}$ 是目标 Actor 网络 $\boldsymbol{\pi}_{\phi'}$ 根据下一个状态生成的动作。

LN-TD3 目标网络更新采用软更新，即每次主网络参数更新后，目标网络参数都会在一定程度上靠近主网络，其中，τ 是一个远小于 1 的超参数：

$$\boldsymbol{\theta}_i'\leftarrow\tau\boldsymbol{\theta}_i+(1-\tau)\boldsymbol{\theta}' \tag{25}$$

$$\boldsymbol{\phi}'\leftarrow\tau\boldsymbol{\phi}+(1-\tau)\boldsymbol{\phi}' \tag{26}$$

LN-TD3 的具体卸载执行过程如图 3 所示，首先，使用随机参数对各网络进行初始化，然后系统将进行 N 回合的迭代训练，每回合被划分为 T 个时隙，在每个时隙，Actor 网络会给出卸载动作，执行动作后到达下一个状态，并获得奖励。

算法 1 LN-TD3 算法

使用随机参数 ϕ 初始化 Actor 网络 π_{ϕ}
使用随机参数 θ_1,θ_2 初始化 Critic 网络 $\boldsymbol{Q}_{\theta_1},\boldsymbol{Q}_{\theta_2}$
初始化目标网络 $\phi'\leftarrow\phi,\theta_1'\leftarrow\theta_1,\theta_2'\leftarrow\theta_2$
初始化经验池 $\mathcal{B}$;
for *episode* ← 1 to N **do**
　初始化环境参数，得到初始状态 s_0，并归一化;
　for *step* ← 1 to T **do**
　　根据公式 (21) 获得动作 $\boldsymbol{a}(t)$;
　　执行动作 $\boldsymbol{a}(t)$ 获得下一状态 $\boldsymbol{s}(t+1)$ 和奖励 $\boldsymbol{r}(t)$;
　　将 $<\boldsymbol{s}(t),\boldsymbol{a}(t),\boldsymbol{s}(t+1),\boldsymbol{r}(t)>$ 放入 $\mathcal{B}$ 中.
　　if 已收集的经验数 >$\mathcal{B}$ 容量 **then**
　　从 $\mathcal{B}$ 中随机抽取一批经验 $<\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}',\boldsymbol{r}>$ 计算:
　　1) 根据式 (21) 计算扰动后的动作 $\tilde{\boldsymbol{a}}$
　　2) 根据式 (24) 计算目标 Q 值 $\boldsymbol{y}$
　　根据式 (23) 更新 θ_1,θ_2
　　if t mod d **then**
　　根据式 (22) 更新 ϕ
　　根据式 (25)(26) 更新目标网络参数 $\phi',\theta_1',\theta_2'$
　　end if
　　end if
　end for
end for

图 3　LN-TD3 算法卸载过程

Critic 网络对动作进行评价并将历史经验存储进经验池，当经验池满后随机抽取一批经验对 Actor 网络、Critic 网络和目标网络参数进行更新，重复以上步骤直到收敛或达到最大训练次数。虽然更新目标网络采用软更新的方法有利于算法的稳定性，但是有时 Critic 提供的值函数估计值不准确，会使 Actor 将策略往错误的方向改进。因此 LN-TD3 采用策略延迟更新，即每隔 Critic 更新 d 次后，再对 Actor 进行更新，尽可能等待 Critic 收敛后在进行更新操作，进一步提高了算法的稳定性。

3　性能仿真与结果分析

3.1　实验环境与参数设置

实验环境为 Python 3.9，Pytorch 2.0.1，CUDA 12.0。显卡为 NVIDIA GTX 4060ti，16 G 内存，处理器为拥有 4 核心 12 线程的 INTEL i5-12400f。

LN-TD3 算法模型中包含 6 个神经网络，每个神经网络均具有 3 个隐藏层，第 1 个隐藏层为 400 个神经元的 LSTM 层，第 2、3 个隐藏层分别为 400，300 个神经元的全连接层(fully connected，FC)，第 1、2 层隐藏层使用 Relu 激活函数，第 3 层使用 Tanh 激活函数。DRL 优化器为 Adam。Actor 网络和 Critic 网络学习率分别设置为 6×10^{-7} 和 6×10^{-7}，经验池大小为 40 000，每次训练抽取样本数 Bach size 为 64，折扣因子 γ 为 0.99，动作噪声 ϵ 为 0.1，软更新因子 μ =0.01 具体仿真参数如表 1 所示。

表 1　仿真实验参数

参数	值
VUE 速度 v	10 m/s
VUE 传输功率 p_0	200 mW
VUE 数量 N	10
ECU 计算频率 $f_{vehicle}$	1.2 GHz
MEC 服务器计算频率 f_{mec}	4 GHz
时延权重 λ_1	0.5
能耗权重 λ_2	0.5
RSU 覆盖半径	500 m
RSU 高度 H_r	10 m
任务规模 D_n	60～140 M
RSU 带宽 B	20 MHz
高斯白噪声 σ	−100 dBm

3.2　收敛性能

首先验证 LN-TD3 算法的收敛性能，本实验采用了两种不同的 DNN 架构对任务卸载的性能进行比较：LSTM 网络嵌入 DNN 架构和 3 层 FC 架构。FC 是最经典，使用最广泛的 DNN 架构之一，已广泛应用于许多研究之中。每个 FC 层有分别具有 400，400，300 个神经元，并使用 Relu 激活函数。训练 700 回合，每回合最多进行 180 步。

在相同的环境中进行测试,并记录两种架构下的系统开销。图 4 显示了两种 DNN 架构在不同 VUE 数量下的收敛曲线,从图中可以看出 LSTM 嵌入 DNN 架构收敛速度明显比 3 层 FC 网络快。

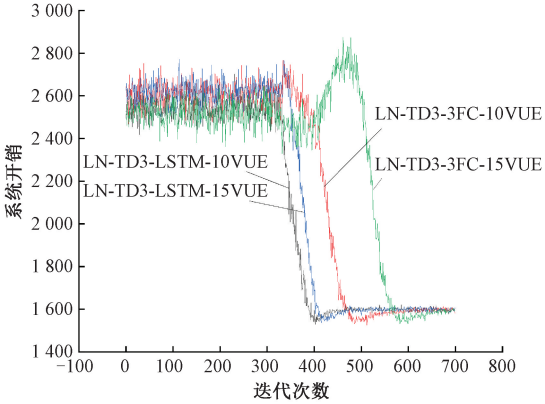

图 4　不同网络架构下算法性能比较

3.3　性能对比分析

为进一步验证 LN-TD3 算法的性能,分别采用以下 4 种不同解决方案和基线算法进行对比实验:

本地计算(Local):所有的任务均由 VUE 在本地处理。

边缘计算(Edge):将所有任务均上传至边缘服务器进行处理。

基于 DQN:DQN 的动作空间是离散的。因此需将连续的动作空间划分为等间隔的区间,然后选择每个区间的中心点作为离散化后的动作值。

基于 DDPG:DDPG 继承了 DQN 的优点,并利用策略梯度技术解决了高维动作空间带来的问题。DDPG 的动作空间是连续的。

为保证实验的公平性,对 DQN 算法和 DDPG 算法均进行状态标准化和嵌入 LSTM 网络进行处理。

图 5 显示了各种算法和解决方案所获得的系统开销的对比图,其中本地计算和边缘计算两种方案的系统开销较大,原因是对于部分卸载来说两者都没有充分利用彼此的计算资源,而依赖于 5 G 的高可靠低时延的特点,并且边缘服务器具有较高的计算资源,所以边缘计算的系统开销要低于本地计算。DQN 算法可以收敛,但是 DQN 算法对连续的动作进行了离散化,无法收敛至最优卸载策略。LN-TD3 算法使用裁剪的双 Q 学习、策略延迟更新等操作解决了 DDPG 算法中存在的高估 Q 值问题,无论是算法的收敛速度还是稳定性都优于 DDPG。DDPG 和 LN-TD3 算法都可以充分利用 VUE 和边缘服务器的计算资源,并且都可以使用连续动作空间,因此都可以收敛到最优策略。

图 6 展示了不同任务规模下各种法性能比较,所有算法的系统开销随任务规模的增加而增加,呈正相关,因为

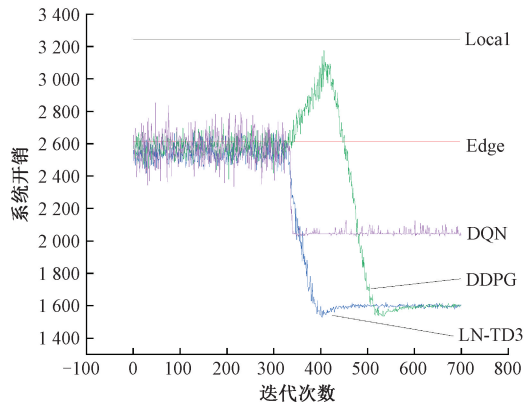

图 5　不同算法下系统开销

更大的数据量会导致卸载任务消耗更多的时间和能源。LN-TD3 算法可以达到更好的效果,因为它的增长趋势比其他算法要慢。随着数据量的增加,本地计算系统开销相对于其他算法增长较快,这表明卸载任务的数据量越大,LN-TD3 从卸载计算中获得的延迟和能耗收益就越大。

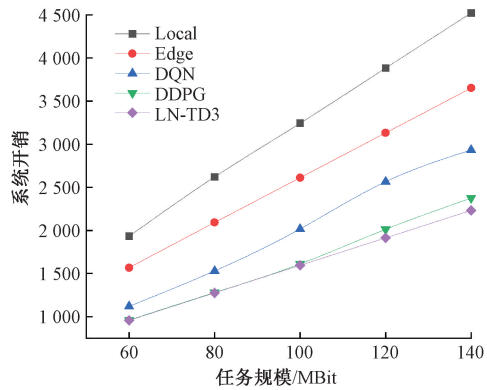

图 6　不同算法在不同任务规模下的系统开销

图 7 显示了不同 VUE 数量下使算法收敛的迭代次数。由于 VUE 数量的增加导致 DRL 模型的状态和动作空间变大,因此 DRL 算法需要更多的迭代训练才能获得最优策略。所以随着 VUE 数量的增加,不同的 DRL 算法达到收敛所需的迭代次数也会增加。而 DQN 算法因无法

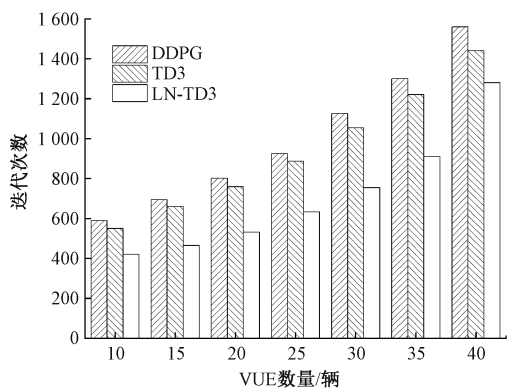

图 7　不同 VUE 数量下算法收敛的迭代次数

使用连续动作空间，而无法收敛到最优解因此不进行比较，此外，与 DDPG 和 TD3 相比，LN-TD3 算法始终需要最少的迭代次数。

4　结　　论

本文研究了车载边缘计算场景下的多用户任务卸载系统，将车辆的任务卸载问题构建为 MDP，提出一种基于深度强化学习的任务卸载决策算法，利用 LSTM 网络来近似策略和价值函数，通过归一化状态空间以提高算法收敛性。该算法能够根据任务特征和网络状态，动态调整卸载决策，以最小化系统开销。实验结果表明，本文提出的 LN-TD3 算法的判决结果能够有效降低系统开销，为车载边缘计算的实际应用提供了可行的解决方案。

参考文献

[1] FENG J, LIU Z, WU C, et al. Mobile edge computing for the internet of vehicles: Offloading framework and job scheduling[J]. IEEE Vehicular Technology Magazine, 2018,14(1): 28-36.

[2] ZHANG K, MAO Y, LENG S, et al. Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading[J]. IEEE Vehicular Technology Magazine, 2017,12(2): 36-44.

[3] LU S, SHI W. The emergence of vehicle computing[J]. IEEE Internet Computing, 2021,25(3): 18-22.

[4] AHMED M, RAZA S, MIRZA M A, et al. A survey on vehicular task offloading: Classification, issues, and challenges[J]. Journal of King Saud University-Computer and Information Sciences, 2022, 34(7): 4135-4162.

[5] ABBAS N, ZHANG Y, TAHERKORDI A, et al. Mobile edge computing: A survey[J]. IEEE Internet of Things Journal, 2017,5(1): 450-465.

[6] ZHANG J, LETAIEF K B. Mobile edge intelligence and computing for the internet of vehicles[J]. Proceedings of the IEEE, 2019,108(2): 246-261.

[7] FATEMIDOKHT H, RAFSANJANI M K, GUPTA B B, et al. Efficient and secure routing protocol based on artificial intelligence algorithms with UAV-assisted for vehicular ad hoc networks in intelligent transportation systems[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, 22(7): 4757-4769.

[8] FENG C, HAN P, ZHANG X, et al. Computation offloading in mobile edge computing networks: A survey[J]. Journal of Network and Computer Applications, 2022,202: 103366.

[9] 丁飞，沙宇晨，洪莹，等. 智能网联汽车计算卸载与边缘缓存联合优化策略[J]. 系统仿真学报，2023，35(6): 1203.

[10] ZHAN W, LUO C, WANG J, et al. Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing[J]. IEEE Internet of Things Journal, 2020,7(6): 5449-5465.

[11] HAN Y, ZHAO Z, MO J, et al. Efficient task offloading with dependency guarantees in ultra-dense edge networks[C]. 2019 IEEE Global Communications Conference(GLOBECOM): IEEE, 2019: 1-6.

[12] ZHANG J, GUO H, LIU J, et al. Task offloading in vehicular edge computing networks: A load-balancing solution[J]. IEEE Transactions on Vehicular Technology, 2019,69(2): 2092-2104.

[13] DAI Y, XU D, MAHARJAN S, et al. Joint load balancing and offloading in vehicular edge computing and networks[J]. IEEE Internet of Things Journal, 2018,6(3): 4377-4387.

[14] YOU C, HUANG K. Multiuser resource allocation for mobile-edge computation offloading[C]. 2016 IEEE Global Communications Conference(GLOBECOM): IEEE, 2016: 1-6.

[15] ZHANG L, ZHOU W, XIA J, et al. DQN-based mobile edge computing for smart Internet of vehicle[J]. EURASIP Journal on Advances in Signal Processing, 2022,2022(1): 1-16.

[16] LI C, XIA J, LIU F, et al. Dynamic offloading for multiuser muti-CAP MEC networks: A deep reinforcement learning approach[J]. IEEE Transactions on Vehicular Technology, 2021,70(3): 2922-2927.

[17] ZHAO H T, ZHANG T, CHEN Y, et al. Task distribution offloading algorithm of vehicle edge network based on DQN[J]. Journal on Communications, 2020,41(10): 172-178.

[18] LU H, HE X, DU M, et al. Edge QoE: Computation offloading with deep reinforcement learning for Internet of Things[J]. IEEE Internet of Things Journal, 2020,7(10): 9255-9265.

[19] HU X, HUANG Y. Deep reinforcement learning based offloading decision algorithm for vehicular edge computing[J]. PeerJ Computer Science, 2022, 8: e1126.

[20] HUANG J, WAN J, LV B, et al. Joint computation offloading and resource allocation for edge-cloud collaboration in internet of vehicles via deep reinforcement learning[J]. IEEE Systems Journal, 2023.

[21] 张仪，冯伟，王卫军，等. 融合 LSTM 和 PPO 算法的移动机器人视觉导航[J]. 电子测量与仪器学报，2022,36(8): 132-140.

作者简介

李亚，副教授，硕士生导师，主要研究方向为新一代移动通信技术，人工智能。

E-mail: liya@hpu.edu.cn

王卫岗，硕士研究生，主要研究方向为新一代移动通信技术，人工智能。

E-mail: doctorwwg@home.hpu.edu.cn

张原，硕士研究生，主要研究方向为新一代移动通信技术，人工智能。

E-mail: 212311020017@home.hpu.edu.cn

刘瑞鹏，硕士研究生，主要研究方向为新一代移动通信技术，人工智能。

E-mail: liurayp@163.com