计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
2026,35(2):40−52 [doi: 10.15888/j.cnki.csa.010075] [CSTR: 32024.14.csa.010075]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于深度强化学习的车联网可信任务卸载①

秦雪晴, 石　琼, 师智斌, 王梦丽

(中北大学 计算机科学与技术学院, 太原 030051)
通信作者: 石　琼, E-mail: shiqiong0641@nuc.edu.cn

摘　要: 针对车载边缘计算 (vehicular edge computing, VEC) 中路侧单元 (road side unit, RSU) 资源受限和高负载的难题, 以及现有的任务卸载优化方案局限于降低时延或能耗, 忽视了边缘节点所面临的安全问题, 提出一种基于信任感知和近端策略优化算法 (PPO) 的任务卸载方案. 首先, 构建 VEC 网络架构, 利用周围空闲车辆的计算资源, 将任务在本地执行或卸载至 RSU、空闲服务车辆进行计算处理, 以降低系统整体时延与能耗. 其次, 构建一种基于多源赋权和奖惩机制的动态反馈信任评估模型, 实现对边缘节点可信度的量化评估. 最后, 利用基于深度强化学习的 PPO 算法对任务卸载策略进行优化. 实验结果表明, 相较于 DQN、D3QN 和 TASACO 算法, 所提方案具有更好的收敛性和稳定性, 而且在任务执行时延和能耗等方面优于现有方案.

关键词: 车载边缘计算 (VEC); 任务卸载; 深度强化学习 (DRL); 信任评估; 网络安全; 车联网 (IoV)

引用格式: 秦雪晴,石琼,师智斌,王梦丽.基于深度强化学习的车联网可信任务卸载.计算机系统应用,2026,35(2):40–52. http://www.c-s-a.org.cn/1003-3254/10075.html

Trustworthy Task Offloading in Internet of Vehicles Based on Deep Reinforcement Learning

QIN Xue-Qing, SHI Qiong, SHI Zhi-Bin, WANG Meng-Li

(School of Computer Science and Technology, North University of China, Taiyuan 030051, China)

Abstract: In response to the challenges of resource constraints and high load in road side unit (RSU) within vehicular edge computing (VEC), as well as the limitations of existing task offloading optimization schemes that focus solely on reducing latency or energy consumption while neglecting the security issues faced by edge nodes, this study proposes a task offloading scheme based on trust awareness and the proximal policy optimization (PPO) algorithm. First, a VEC network architecture is constructed, which utilizes the computing resources of nearby idle vehicles to process tasks locally or offload them to RSU or idle service vehicles, in order to reduce the overall system latency and energy consumption. Second, a dynamic feedback trust evaluation model based on multi-source weighting and a reward-punishment mechanism is constructed to achieve a quantitative assessment of the trustworthiness of edge nodes. Finally, the task offloading strategy is optimized using the PPO algorithm based on deep reinforcement learning. Experimental results show that compared to the DQN, D3QN, and TASACO algorithms, the proposed scheme has better convergence and stability, and it outperforms existing schemes in terms of task execution latency and energy consumption.

Key words: vehicular edge computing (VEC); task offloading; deep reinforcement learning (DRL); trust evaluation; network security; Internet of Vehicles (IoV)

随着智能交通系统的不断发展, 大量延迟敏感型的应用 (如自动驾驶、智能导航等) 逐渐受到用户的青睐[1]. 这些车载应用普遍是计算密集型的, 对时延、能耗及可靠性的要求极高, 然而传统车辆受限于自身的

① 基金项目: 山西省基础研究计划青年科学研究项目 (202303021222098)
收稿时间: 2025-07-29; 修改时间: 2025-08-28; 采用时间: 2025-09-15; csa 在线出版时间: 2025-12-26
CNKI 网络首发时间: 2025-12-29

计算能力难以满足这些低延迟的新兴应用[2]. 为缓解这一局限, 移动边缘计算 (mobile edge computing, MEC) 和云计算成为可行的解决方案, 在一定程度上缓解了终端车辆计算资源不足的问题[3]. 在车载边缘计算 (vehicular edge computing, VEC) 环境中, 云计算存在传输时延高、带宽占用大等问题, 不适合处理延迟敏感型任务. 与云计算不同, MEC 通过在路侧单元 (road side unit, RSU) 部署边缘服务器, 构建分布式计算网络, 降低了系统的通信时延和能耗[4]. 然而, 当任务请求量增大时, 仍会导致较高的时延和负载. 由于车辆自身携带计算资源, 空闲车辆也可作为边缘服务器. 充分挖掘并利用车联网 (Internet of Vehicles, IoV) 中的空闲车辆对于降低任务时延和 RSU 的负载具有重要意义[5].

目前, 许多研究专注于优化任务卸载的时延和能耗, 未能对任务卸载的安全性和可靠性进行全面评估[6]. 由于边缘节点的开放性和复杂性, 它们可能会遭到恶意攻击, 一旦被攻击, 恶意节点可能会窃取数据, 导致用户隐私泄露, 严重威胁用户安全[7]. 因此, 在任务卸载过程中, 选择安全可靠的边缘节点是至关重要的.

针对 VEC 中任务卸载的研究在降低时延和能耗方面取得了显著成效. 例如, 文献[8]将时延和能耗作为优化目标函数, 提出一种基于改进遗传算法的任务卸载方法对目标函数进行求解. 文献[9]采用 Lyapunov 将长期优化过程分解, 在时延和能耗之间实现平衡. 文献[10]构建一种车-路-空架构, 并以负载均衡性作为优化目标, 提出了一种基于软件定义网络和深度强化学习 (DRL) 的任务卸载算法. 针对卸载资源分配问题. 文献[11]提出了一种基于博弈论的多边缘服务器选择算法, 通过博弈论制定竞争, 有效降低了时延和能耗. 文献[12]考虑到了任务之间的依赖关系, 将系统建模为有向无环图, 利用深度 Q 网络算法实现任务分配, 优化系统性能. 文献[13]提出了一种基于协同多智能体深度强化学习的任务卸载方法, 利用多智能体的集中式训练和分散式执行优化奖励, 提高了任务卸载效率. 对于 VEC 环境中的任务卸载问题, 现有研究致力于性能的优化, 忽视了边缘节点的安全性与可靠性. 在处理敏感数据时, 边缘节点的安全对系统的性能和稳定有着至关重要的影响.

针对边缘节点的安全问题的研究已取得一定进展. 例如, 文献[8]提出了一种基于声誉的信任模型, 将声誉作为评估的依据并利用声誉机制激励边缘服务器参与任务竞争. 文献[14]利用机器学习方法检测恶意行为, 增强了系统安全性. 文献[15]提出了轻量级的信任评估模型, 有效解决了终端设备资源受限和节点不信任问题. 文献[16]提出了一种基于邻近车辆推荐的信任评估模型, 并考虑了节点的移动性, 有效抵御恶意行为. 文献[17]提出一种基于时间衰减和交互频率的多反馈信任聚合模型, 量化了边缘节点的可靠性, 并显著提高了恶意节点检测的准确率. 文献[18]提出了一种多源反馈信任融合模型, 通过融合客观信息熵与历史交互频率对信任进行加权得到反馈信任, 更准确地衡量了边缘服务器的可信度. 文献[19]提出一种基于反馈的信任管理模型, 通过引入奖惩因子, 加快恶意节点可信度的下降速度, 从而提升网络安全性. 文献[8,14]需要大量计算资源和存储资源, 很难适用于资源受限的终端车辆. 文献[15,16]信任值的计算依赖于邻居推荐, 难以抵抗邻居的虚假反馈攻击. 文献[17,18]在信任值的计算上通过多源反馈提高了对边缘节点评估的准确性, 有效抵御了终端设备的虚假反馈攻击, 但无法应对长期伪装节点的突发攻击, 即摇摆攻击. 因此, 基于边缘计算环境的信任评估模型依旧有很大的研究和优化空间.

针对以上问题, 本文提出了一种基于信任感知和近端策略优化算法 (proximal policy optimization, PPO) 的任务卸载方法 (task offloading based on trust awareness and PPO algorithm, TOTAPPO). 主要贡献如下: (1) 构建了终端车辆与 RSU (V2R) 以及空闲服务车辆 (V2V) 之间的通信链路模型. (2) 在评估边缘节点可信度阶段, 构建了融合多源赋权和奖惩机制的动态反馈信任模型. 其中, 多源赋权侧重长期交互稳定性, 奖惩机制关注短期交互的动态变化, 两者的结合有效提升了信任评估的鲁棒性与准确性. (3) 在任务卸载决策阶段, 将时延、能耗和可信度的联合优化问题转化为系统收益最大化问题. 为实现优化目标, 将基站作为智能体, 利用基于 DRL 的 PPO 算法, 通过限制策略更新幅度提升训练稳定性, 从而实现快速且高效的最优卸载决策.

1 系统模型

1.1 VEC 网络模型

图 1 展示了 VEC 网络架构模型. 在该模型中, 本文针对特定区域内的网络环境进行研究, 该区域由终端车辆、空闲服务车辆以及 RSU 构成. RSU 部署在道路附近, 并内置 MEC 服务器; 同时空闲服务车辆也配备了一定的计算资源. 将空闲服务车辆和 RSU 共同定义为 VEC 环境的边缘节点. 在整个系统中, 终端车辆和边缘节点通过无线链路与基站通信. 基站不仅负责

评估边缘节点的可信度, 还为终端车辆生成的计算任务制定最优卸载决策.

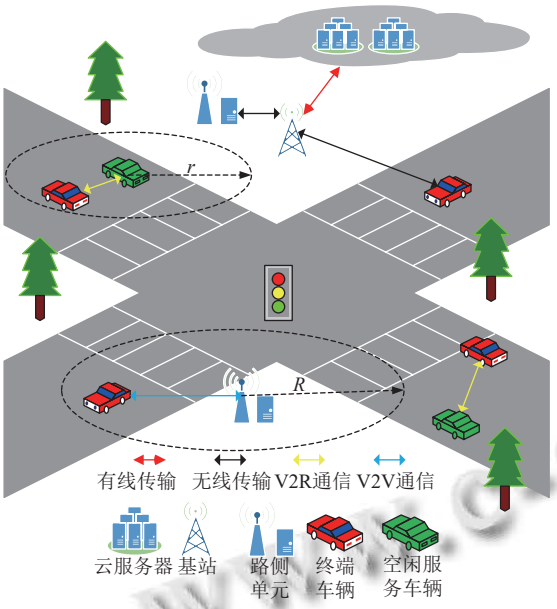


图 1 VEC 任务卸载网络模型

假设在给定区域 $(L\times L)$ 和时间段内的道路上有 1 台基站 BS 及 M 辆终端车辆, 第 m 辆车用 x_m $(1\leqslant m\leqslant M)$ 表示, 位置坐标用 $p_{x_m}=[X_m,Y_m]$ 表示; 有 N 个 RSU, 第 n 个 RSU 用 y_n $(1\leqslant n\leqslant N)$ 表示, 位置坐标用 $p_{y_n}=[X_n,Y_n]$ 表示; K 辆空闲服务车辆, 第 k 辆空闲服务车辆用 z_k $(1\leqslant k\leqslant K)$ 表示, 位置坐标用 $p_{z_k}=[X_k,Y_k]$ 表示. 假设 RSU 的覆盖区域半径为 R, 空闲服务车辆的覆盖区域半径为 r, 且空闲服务车辆和终端车辆沿同一方向行驶.

将时间 T 划分为 s 个大小相等的时隙, 即 $|T_1|=|T_2|=\cdots=|T_t|=\cdots=|T_s|$. 在时隙 T_t 内终端车辆 x_m 生成 I 个计算任务, 第 i 个计算任务定义为 D_i, 任务 $D_i=\left\{d_i,f_i,T_i^{\max}\right\}$, 其中 d_i 表示计算任务的数据大小, f_i 表示单位计算任务所需的计算资源, $T_i^{\max}$ 表示计算任务的最大容忍延迟. 在时隙 T_t 内终端车辆 x_m 生成的计算任务 D_i 可以通过 V2R 链路卸载到 RSU, 或通过 V2V 链路卸载到空闲服务车辆, 也可以选择在本地执行. 将计算任务 D_i 的卸载决策定义为 $a_i=\left\{a_i^{\text{local}},a_{i,n}^{\text{V2R}},a_{i,k}^{\text{V2V}}\right\}$ $(1\leqslant n\leqslant N,1\leqslant k\leqslant K)$, 其中 $a_i\in\{0,1\}$, $a_i^{\text{local}}=1$ 表示任务 D_i 在本地处理, $a_{i,n}^{\text{V2R}}=1$ 表示任务 D_i 卸载到路侧单元 y_n 上执行, 否则 $a_{i,n}^{\text{V2R}}=0$. $a_{i,k}^{\text{V2V}}=1$ 表示任务 D_i 卸载到空闲服务车辆 z_k 上计算, 否则 $a_{i,k}^{\text{V2V}}=0$.

1.2 通信模型

当终端车辆选择将任务 D_i 通过 V2V 链路连接到空闲服务车辆计算时, 任务 D_i 的上行链路传输速率 r_i^{V2V} 表示为[20]:

$$r_i^{\text{V2V}}=B^{\text{V2V}}\log_2\left(1+\frac{p_i^{\text{tran}}h^{\text{V2V}}}{\sigma^2}\right) \tag{1}$$

其中, B^{V2V} 表示 V2V 链路分配的信道带宽; p_i^{tran} 表示任务的发射功率; σ^2 表示高斯白噪声功率; h^{V2V} 表示终端车辆和空闲服务车辆之间的信道增益:

$$h^{\text{V2V}}=g\left(\frac{l_0}{l_{x_m,z_k}}\right)^{\xi} \tag{2}$$

其中, g 表示路径损耗常数; l_0 表示单位参考距离; ξ 表示路径损耗指数; l_{x_m,z_k} 表示终端车辆和空闲服务车辆之间的距离:

$$l_{x_m,z_k}=\sqrt{(X_k-X_m)^2+(Y_k-Y_m)^2} \tag{3}$$

当终端车辆选择将任务 D_i 通过 V2R 链路连接到 RSU 计算时, 任务 D_i 的上行链路传输速率 r_i^{V2R} 表示为:

$$r_i^{\text{V2R}}=B^{\text{V2R}}\log_2\left(1+\frac{p_i^{\text{tran}}h^{\text{V2R}}}{\sigma^2}\right) \tag{4}$$

其中, B^{V2R} 表示 V2R 链路分配的信道带宽; h^{V2R} 表示终端车辆和 RSU 之间的信道增益.

1.3 计算模型

本文探索了 3 种计算模型: 本地计算模型、空闲服务车辆计算模型、RSU 计算模型. 具备计算能力的空闲服务车辆可为终端车辆提供计算支持; 同时, RSU 上部署了边缘服务器, 可为终端车辆提供计算服务.

1.3.1 本地计算模型

任务 D_i 在本地执行时的计算时延为[20]:

$$T_i^{\text{local}}=\frac{d_if_i}{c_i^{\text{local}}} \tag{5}$$

其中, c_i^{local} 表示任务 D_i 所在终端车辆的计算资源; d_i 表示 D_i 的任务大小; f_i 表示 D_i 所需的计算资源.

任务 D_i 在本地执行时的计算能耗为:

$$E_i^{\text{local}}=\kappa\left(c_i^{\text{local}}\right)^3T_i^{\text{local}} \tag{6}$$

其中, κ 表示功率系数.

1.3.2 空闲服务车辆计算模型

由于计算结果的数据量远小于输入任务的数据量, 因此, 本文不考虑计算结果的回传时延.

任务 D_i 选择卸载到空闲服务车辆执行时的传输时延和计算时延表示为:

$$T_i^{\mathrm{V2Vtran}} = \frac{d_i}{r_i^{\mathrm{V2V}}} \tag{7}$$

$$T_i^{\mathrm{V2Vcomp}} = \frac{d_i f_i}{c_i^{\mathrm{V}}} \tag{8}$$

其中, c_i^{V} 表示空闲服务车辆的计算资源.

任务 D_i 选择卸载到空闲服务车辆执行时的传输能耗和计算能耗表示为:

$$E_i^{\mathrm{V2Vtran}} = p_i^{\mathrm{tran}} \times T_i^{\mathrm{V2Vtran}} \tag{9}$$

$$E_i^{\mathrm{V2Vcomp}} = \delta_{\mathrm{V}} \times T_i^{\mathrm{V2Vcomp}} \tag{10}$$

其中, p_i^{tran} 表示传输功率, δ_{V} 表示空闲车辆的计算功率.

任务 D_i 选择卸载到空闲服务车辆时的总时延和总能耗表示为:

$$T_i^{\mathrm{V2V}} = T_i^{\mathrm{V2Vtran}} + T_i^{\mathrm{V2Vcomp}} \tag{11}$$

$$E_i^{\mathrm{V2V}} = E_i^{\mathrm{V2Vtran}} + E_i^{\mathrm{V2Vcomp}} \tag{12}$$

1.3.3 RSU 计算模型

任务 D_i 选择卸载到 RSU 执行时的传输时延和计算时延表示为:

$$T_i^{\mathrm{V2Rtran}} = \frac{d_i}{r_i^{\mathrm{V2R}}} \tag{13}$$

$$T_i^{\mathrm{V2Rcomp}} = \frac{d_i f_i}{c_i^{\mathrm{R}}} \tag{14}$$

其中, c_i^{R} 表示 RSU 的计算资源.

任务 D_i 选择卸载到 RSU 执行时的传输能耗和计算能耗表示为:

$$E_i^{\mathrm{V2Rtran}} = p_i^{\mathrm{tran}} \times T_i^{\mathrm{V2Rtran}} \tag{15}$$

$$E_i^{\mathrm{V2Rcomp}} = \delta_{\mathrm{R}} \times T_i^{\mathrm{V2Rcomp}} \tag{16}$$

其中, δ_{R} 表示 RSU 的计算功率.

任务 D_i 选择卸载到 RSU 时的总时延和总能耗表示为:

$$T_i^{\mathrm{V2R}} = T_i^{\mathrm{V2Rtran}} + T_i^{\mathrm{V2Rcomp}} \tag{17}$$

$$E_i^{\mathrm{V2R}} = E_i^{\mathrm{V2Rtran}} + E_i^{\mathrm{V2Rcomp}} \tag{18}$$

2 基于 DRL 的车联网可信任务卸载方法

在本节中, 首先构建了一个面向车联网的动态反馈信任模型, 然后定义了联合优化时延、能耗和可信度的任务卸载问题, 最后提出了基于信任感知和 PPO 算法的任务卸载方法.

2.1 基于多源赋权和奖惩机制的动态反馈信任模型

本文构建了一个基于多源赋权和奖惩机制的动态反馈信任模型, 智能体基站B通过在其通信范围内收集终端车辆的多维反馈信息, 实现对空闲服务车辆和 RSU 等边缘节点可信度的综合评估.

首先, 采用基于贝叶斯信任评估的期望计算服务满意度 $q_m^e(t)$, 用于衡量终端车辆 m 在时间窗口 t 内与边缘节点 e 的交互表现, 计算如下:

$$q_m^e(t) = \frac{s_m^e(t)+1}{s_m^e(t)+f_m^e(t)+2} \tag{19}$$

其中, $s_m^e(t)$ 表示在时间窗口 t 内, 终端车辆 m 向边缘节点 e 成功卸载任务的次数; $f_m^e(t)$ 表示卸载失败的次数.

考虑到车联网中基站资源有限, 如果采用时间衰减函数进行信任值更新, 基站需要存储每一个时间点的信任值[17]. 相比之下, 采用滑动窗口机制进行信任更新, 仅需存储上一个时间点的信任值, 降低了存储和计算的开销. 直接信任更新如下:

$$DT_m^e(t) = \eta q_m^e(t) + (1-\eta) DT_m^e(t-1) \tag{20}$$

智能体周期性地在其覆盖范围内广播请求, 收集所有终端车辆的直接信任值, 并将其存储到反馈信任矩阵中, 表示如下:

$$H_{m\to e}(t) = \begin{bmatrix} DT_{m_1}^{e_1}(t) & DT_{m_1}^{e_2}(t) & \cdots & DT_{m_1}^{e_n}(t) \\ DT_{m_2}^{e_1}(t) & DT_{m_2}^{e_2}(t) & \cdots & DT_{m_2}^{e_n}(t) \\ \vdots & \vdots & \ddots & \vdots \\ DT_{m}^{e_1}(t) & DT_{m}^{e_2}(t) & \cdots & DT_{m}^{e_n}(t) \end{bmatrix} \tag{21}$$

2.1.1 基于多源赋权的反馈信任值

基于多源赋权的反馈信任从相对信任和交互频率两个维度对终端车辆的直接信任信息进行加权融合, 有效提升了信任评估的准确性和长期稳定性.

根据信息熵理论, 终端车辆的熵值为:

$$E_m = \ln M \sum_{e=1}^{E} p_m^e \ln p_m^e \tag{22}$$

其中, $p_m^e = DT_m^e(t) \Big/ \sum_{e=1}^{E} DT_m^e(t)$ 表示边缘节点可信度的比重; $E = n + k$ 表示边缘节点的总数量; M 表示终端车辆的数量.

终端车辆的信息熵权重为:

$$w_m^{\mathrm{entropy}} = (1-E_m) \Big/ \left(M - \sum_{m=1}^{M} E_m \right) \tag{23}$$

交互频率权重为:

$$w_{m,e}^{\text{inter}} = h_m^e \Big/ \sum_{m=1}^{M} h_m^e \tag{24}$$

基于信息熵和交互频率的综合权重为:

$$w_m^e = \frac{w_m^{\text{entropy}} \times w_{m,e}^{\text{inter}}}{\sum_{m=1}^{M} \left(w_m^{\text{entropy}} \times w_{m,e}^{\text{inter}} \right)} \tag{25}$$

基于多源赋权的反馈信任值计算如下:

$$FT_e^{\text{weight}}(t) = \sum_{m=1}^{M} DT_m^e(t) \times w_m^e \tag{26}$$

2.1.2 基于奖惩机制的反馈信任值

在反馈信任矩阵中, 若$DT_m^e(t) \geqslant 0.5$, 则被定义为正评价, 记为DT_e^+; 反之, 若$DT_m^e(t) < 0.5$, 则被定义为负评价, 记为DT_e^-. 初始反馈信任计算如下:

$$FT_e^{\text{start}} = \left(\sum_{m=1}^{M} DT_e^+ \right) M^{-1} \left(\sum_{m=1}^{M} DT_e^- \right)^{-1/\alpha} \tag{27}$$

其中, α为敏感因子.

智能体通过统计方法分析终端车辆对边缘节点的直接信任值在$[t_1, t_2]$时间段内的变化, 负评价次数变化计算如下:

$$\Delta[(e;t_1,t_2)^-] = \sum_{m=1}^{M} DT_{m,e}^-(t_2) - \sum_{m=1}^{M} DT_{m,e}^-(t_1) \tag{28}$$

引入动态奖惩机制旨在激励终端车辆保持诚实行为, 对于近期表现诚实的节点给予奖励, 以加快信任值的积累; 对于近期出现恶意行为的节点施加惩罚, 从而迅速降低其信任值. 基于奖惩机制的反馈信任计算如下:

$$FT_e^{rp}(t) = \begin{cases} \dfrac{FT_e^{\text{start}}}{1 + \sqrt{\Delta[(e;t_1,t_2)^-]/\tau_1}}, & \Delta[(e;t_1,t_2)^-] > 0 \\ FT_e^{\text{start}}, & \Delta[(e;t_1,t_2)^-] = 0 \\ \min\left(1, \mathrm{e}^{\Delta[(e;t_1,t_2)^+]/\tau_2} \times FT_e^{\text{start}}\right), & \Delta[(e;t_1,t_2)^-] < 0 \end{cases} \tag{29}$$

其中, τ_1表示下降率因子, 控制惩罚的强度; τ_2表示增长率因子, 控制奖励的强度. 当$\Delta[(e;t_1,t_2)^-] > 0$表示边缘节点$e$在时间段$[t_1, t_2]$内负评价次数增加, 相较于传统线性惩罚, $\sqrt{\cdot}$函数能避免因少量负面行为导致信任值过快下降, 同时确保大量负面行为出现时施加足够严厉的惩罚. 当$\Delta[(e;t_1,t_2)^-] < 0$表示边缘节点$e$在时间段$[t_1, t_2]$内负评价次数减少, 指数函数具有递增且增速加快的特性, 诚实行为越多则信任积累速度越快. 由于信任值定义在$[0,1]$区间, 使用$\min(1,\cdot)$避免数值溢出.

2.1.3 全局反馈信任值

融合基于多源赋权的反馈信任和基于奖惩机制的反馈信任得到全局反馈信任值, 表示如下:

$$GT_e(t) = \mu_1 \times FT_e^{\text{weight}}(t) + \mu_2 \times FT_e^{rp}(t) \tag{30}$$

其中, μ_1、μ_2表示权重, 均为可调参数.

在信任评估过程中, 多源赋权既考虑了长期反馈信息的质量, 又考虑了卸载频率, 能有效提升评估结果的稳定性. 奖惩机制关注边缘节点短期的行为表现, 能够快速识别并响应节点的异常行为. 虽然多源赋权在长期稳定性信任评估方面具有显著优势, 但其对节点行为突变的响应存在一定滞后性. 将两者结合不仅能兼顾信任评估的长期稳定性和短期敏感性, 还能增强系统对恶意行为的识别和防御, 提升系统整体安全性.

2.2 联合时延、能耗和可信度的优化问题

随着车载应用的快速发展, 车联网中的计算任务对时延、能耗和可靠性提出了更为严苛的要求. 在 VEC 网络中, 通过收集车辆和边缘节点的计算能力、边缘节点可信度和计算任务状态信息等, 智能体能够实现高效和智能的任务卸载与资源分配.

在终端车辆、空闲服务车辆与 RSU 协作的系统模型中, 本文为获得最优的任务卸载策略, 构建了一个联合优化问题, 目标是在最小化卸载延迟和能耗的同时, 最大化由边缘节点可信度提升所带来的系统收益. 因此, 系统的总体收益可表示为:

$$C = \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{i=1}^{N} -\alpha_1 T_i^w - \alpha_2 E_i^w + \alpha_3 GT^w(t) \tag{31}$$

其中, T_i^w表示任务D_i在设备w上执行的总时延; E_i^w表示任务D_i在设备w上执行的总能耗; $GT^w(t)$表示执行任务设备w的可信度, α_1、α_2和α_3分别为时延、能耗和可信度的权重系数.

优化问题可以表述为最大化系统收益, 如下:

$$P_1 : \max C \tag{32}$$

约束条件如式 (33)–式 (37):

$$\text{C1} : w \in \{x_m, y_n, z_k\} \tag{33}$$

$$\text{C2} : GT_{\text{local}} = 1,\ GT_{y_n} \in (0, 1],\ GT_{z_k} \in (0, 1] \tag{34}$$

$$\text{C3} : T_i^{\text{local}}, T_i^{\text{V2V}}, T_i^{\text{V2R}} \leqslant T_i^{\max} \tag{35}$$

$$\mathrm{C4}: 0 < c_{D_i} \leqslant c^{\max} \tag{36}$$

$$\mathrm{C5}: l_{x_m,z_k} < r,\ l_{x_m,y_n} < R \tag{37}$$

其中, C1表示车辆计算任务的执行位置, $w = x_m$, $w = y_n$, $w = z_k$ 分别表示计算任务在本地、RSU y_n、空闲服务车辆z_k上执行. C2 表示执行任务设备的可信度, $GT_{\text{local}} = 1$表示终端车辆的可信度, GT_{y_n} 表示执行任务的 RSU 的可信度, GT_{z_k} 表示执行任务的空闲服务车辆的可信度. C3表示车辆计算任务在本地或空闲服务车辆或 RSU 的执行时间不超过任务的最大容忍延迟. C4表示分配给任务的计算资源不超过卸载设备所具备的最大计算资源. C5表示终端车辆与空闲服务车辆的距离和终端车辆与 RSU 的距离在覆盖范围内.

该联合优化问题是 NP-难问题, 用传统数学方法难以获得精确解. 考虑到 DRL 在处理高维、动态和不确定性决策问题中的显著优势, 因此, 本文采用 DRL 方法以获得该问题的近似解或全局最优解.

2.3 基于 PPO 的可信任务卸载方案

为了解决上述问题并实现优化目标, 本文将终端车辆的任务卸载问题建模为马尔可夫决策过程 (Markov decision process, MDP). 在该模型中, 基站B被视为智能体, 智能体根据当前环境状态选择动作, 并从环境中获取相应奖励, 环境随之更新至下一个状态. MDP 可表示为$M = (S, A, P, R)$, 其中S 为状态空间, A为动作空间, $P(s'|s,a)$表示在状态s下采取动作a后转移到下一状态s'的概率, R则表示所获得的即时奖励.

2.3.1 状态空间

状态空间包含从 VEC 环境中观测到的信息, 在任意时刻t的状态空间S_t表示为:

$$S_t = \left\{c_{x_m}, c_{y_n}, c_{z_k}, GT_{y_n}, GT_{z_k}, d_i, f_i, T_i^{\max}\right\} \tag{38}$$

其中, c_{x_m}、c_{y_n}和c_{z_k}分别表示终端车辆、RSU 和空闲服务车辆的计算资源. GT_{y_n}和GT_{z_k}分别表示 RSU 与空闲服务车辆的可信度.

2.3.2 动作空间

对于任务D_i卸载决策, 智能体可选择将任务D_i本地执行或将其卸载至空闲服务车辆或 RSU. 基于当前环境状态, 智能体在任意时刻t的动作空间A_t可表示为:

$$A_t = \{a_i\} \tag{39}$$

2.3.3 奖励函数

奖励函数的设计应与系统的优化目标相一致, 综合考虑时延、能耗以及边缘节点可信度, 奖励函数可表示为:

$$R(S_t, A_t) = \beta_1 \cdot \exp(T_i(t)) + \beta_2 \cdot \exp(E_i(t)) + \beta_3 \cdot \exp(GT^e(t)) \tag{40}$$

其中, β_1、β_2和β_3分别表示时延奖励、能耗奖励和可信度奖励的权重.

2.3.4 基于 DRL 的 PPO 算法

强化学习是通过智能体与环境持续交互, 旨在最大化累计奖励, 是求解 MDP 问题的有效方法. 深度强化学习结合了深度学习的强大表达能力, 有效提升了策略学习的效率与鲁棒性. 其中, 近端策略优化 (PPO) 算法[21]作为一种深度强化学习方法, 能适应离散和连续的状态-动作空间, 并具有降低模型复杂度、加速收敛速度以及避免陷入局部最优解等优势.

PPO 算法是一种基于策略梯度的方法, 通过优化策略网络以实现任务卸载策略的优化. 基于深度强化学习的 PPO 算法网络架构如图 2 所示. 该架构主要由 3 个神经网络组成: 一个参数为θ的新策略网络$\pi_\theta(a|s)$、一个参数为θ_{old}的旧策略网络$\pi_{\theta_{\text{old}}}(a|s)$和一个参数为$\phi$的状态价值网络$V_\phi(s)$.

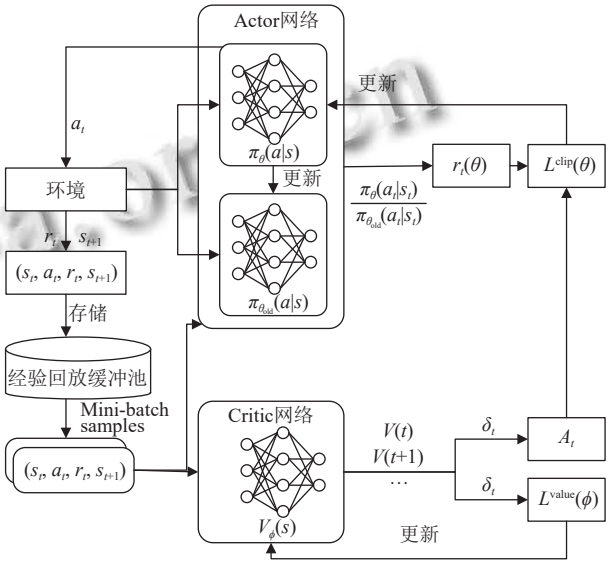


图 2 PPO 算法网络架构

PPO 算法利用新策略网络$\pi_\theta(a|s)$与环境进行交互以采集经验样本. 具体而言, 通过策略网络输出的动作分布进行采样, 执行所选动作后, 观察环境反馈的即时奖励以及转移到的下一状态, 生成形式为(s_t, a_t, r_t, s_{t+1})的元组, 并将其存储到经验回放缓冲区. 当缓冲区容量

达到设置的上限时, 交互阶段结束, 进入模型更新阶段.

智能体B在采集经验后, 需要对所执行的动作进行评估, 以指导后续策略更新. 在强化学习中, 常用的评估标准是累积奖励, 即从当前时刻开始, 智能体在未来能够获得的总奖励. 累积奖励R_t表示为:

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{41}$$

其中, $\gamma \in [0,1]$表示折扣因子, 用于平衡未来奖励的重要性; r_{t+k}表示在第$t+k$时刻获得的即时奖励.

状态值函数$V(s_t)$用于评估状态的好坏, 表示在状态s_t下, 智能体在遵循当前策略π的情况下, 所期望获得的累积回报.

$$V^{\pi}(s_t) = \mathbb{E}_{\pi}[R_t|s_t] \tag{42}$$

动作值函数$Q(s_t,a_t)$表示在状态s_t下采取动作a_t, 并在此后按照策略π执行, 所期望获得的累积回报.

$$Q^{\pi}(s_t,a_t) = \mathbb{E}_{\pi}[R_t|s_t,a_t] \tag{43}$$

为解决原始策略梯度会导致策略更新幅度过大、训练不稳定的问题, PPO 算法引入了重要性采样比率和剪切目标函数来限制策略更新的步长, 从而提高训练的稳定性和效率.

PPO 算法通过使用当前策略对先前采集的交互数据进行重新评估, 重要性采样比率是用来衡量新旧策略之间的差异程度:

$$r_t(\theta) = \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \tag{44}$$

其中, $\pi_{\theta_{\text{old}}}$表示旧策略; π_{θ}表示当前策略.

为了提高策略更新的有效性和稳定性, PPO 算法在策略更新的过程中引入裁剪机制. 该机制通过限制策略的更新幅度, 防止新策略过度偏离旧策略, 从而在学习效率和训练稳定性之间实现平衡, 也会使得 PPO 算法在复杂环境中展现出更强的鲁棒性和泛化能力. PPO 算法使用以下目标函数进行策略梯度计算:

$$L^{\text{clip}}(\theta) = \mathbb{E}_t\left[\min\{r_t(\theta)A_t, clip(r_t(\theta),1-\varepsilon,1+\varepsilon)A_t\}\right] \tag{45}$$

其中, θ表示当前策略网络的参数; A_t表示优势函数; ε是裁剪阈值, 限制策略更新的幅度, 防止策略发生过大变化.

在 PPO 算法中, 优势函数$\widehat{A_t}$是构造策略梯度目标函数的核心组件, 用来衡量某一动作相对于平均水平的好坏. 为了提高训练稳定性并降低估计方差[22], PPO 算法采用广义优势估计 (generalized advantage estimation, GAE) 方法. GAE 通过对多个时序差分 (temporal-difference, TD) 误差进行加权平均, 有效结合了低偏差和低方差的特点. 其计算公式如下:

$$\widehat{A_t} = \delta_t + \gamma\lambda\widehat{A_{t+1}} = \sum_{k=0}^{T-t}(\gamma\lambda)^k\delta_{t+k} \tag{46}$$

其中, δ_t表示 TD 误差; λ表示衰减因子, 用于控制偏差与方差的平衡.

价值网络通过结合即时奖励r_t与下一状态s_{t+1}的价值估计$V_{\phi}(s_{t+1})$来计算 TD 误差:

$$\delta_t = r_t + \gamma V_{\phi}(s_{t+1}) - V_{\phi}(s_t) \tag{47}$$

最后, 使用策略梯度上升优化策略网络的参数θ:

$$\theta \leftarrow \theta + \alpha\nabla_{\theta}L^{\text{clip}}(\theta) \tag{48}$$

其中, α是策略网络学习率.

PPO 算法中值网络的更新是通过最小化其当前估计值与目标值之间的均方误差 (MSE) 来实现的, 其损失函数定义如下:

$$L^{\text{value}}(\phi) = E_t\left[V_{\phi}(s_t) - \left(r_t + \gamma V_{\phi}(s_{t+1})\right)\right]^2 \tag{49}$$

其中, $V_{\phi}(s_t)$表示价值网络对状态s_t的估计结果; $r_t+\gamma V_{\phi}(s_{t+1})$是目标值, 通过即时奖励和下一状态价值估计计算的.

使用梯度下降优化价值网络的参数ϕ, 计算如下:

$$\phi \leftarrow \phi - \beta\nabla_{\phi}L^{\text{value}}(\phi) \tag{50}$$

其中, β是价值网络学习率.

PPO 算法通过迭代优化的方式实现策略改进. 其训练机制可分为 3 个阶段: 首先, 智能体与环境持续交互, 将收集的经验元组存储到经验缓冲区中. 随后, 算法按照预设的时间步长从缓冲区中随机采样批量数据, 使用梯度上升、梯度下降分别更新策略网络和价值网络参数. 通过不断优化策略网络和价值网络的参数, 可逐步提升任务卸载策略的性能. 基于 PPO 的计算卸载优化算法如算法 1 所示.

算法 1. 基于 PPO 的可信任务卸载算法

输入: 任务卸载环境.
输出: 任务卸载策略.

1. 初始化策略网络参数θ, $\theta_{\text{old}}\leftarrow\theta$和价值网络参数$\phi$
2. 初始化用于存储经验的回放缓冲区
3. 初始化信任值及其他参数
4. for n=1 to N_{episodes} do:

5. for t=1 to T do:
6. 智能体B观察环境获取状态s_t
7. 根据策略π_θ选择动作a_t
8. 获取奖励r_t和下一状态s_{t+1}
9. 将(s_t,a_t,r_t,s_{t+1})存储到回放缓冲池
10. end for //结束循环
11. for t=1 to T do:
12. 计算$V_\phi(s_{t+1})$和$V_\phi(s_t)$的状态值
13. 根据式 (47) 计算时序差分误差
14. 根据式 (49) 计算价值网络损失函数
15. 根据式 (50) 使用梯度下降法更新ϕ
16. 根据式 (46) 计算广义优势估计$\widehat{A}_t$
17. 根据式 (45) 计算裁剪目标函数
18. 根据式 (48) 使用梯度上升法更新θ
19. 更新$\theta_{\text{old}}=\theta$
20. end for //结束循环
21. 清空回放缓冲区
22. end for //结束循环

在算法 1 中, 初始化参数的复杂度是O(1). 在交互阶段, 采样的复杂度为O(T), 这一操作从$n=1$到N_{episodes}重复, 所以复杂度为O(NT). 在策略优化阶段, 每一步涉及前向和反向传播, 复杂度为O(F), 每步循环T次, 每回合循环N次, 所以复杂度为O(NTF). 总体的时间复杂度为两阶段复杂度相加, 即O(NTF).

3 仿真实验和结果分析

3.1 实验设置

3.1.1 实验参数

本文实验基于 Python 3.8 和 PyTorch 2.2.0 框架实现, 运行环境为 Ubuntu 22.04 系统, 在 RTX 4090 GPU 服务器上执行. 模型训练采用 Adam 优化器进行参数更新. 主要仿真实验参数如表 1 所示.

表 1 主要参数

参数	参数取值	说明
B_1	40	V2R信道总带宽 (MHz)
B_2	20	V2V信道总带宽 (MHz)
R	100	RSU覆盖范围 (m)
d_i	[1, 1.5]	计算任务大小 (Mb)
f_i	[500, 800]	任务所需计算资源 (cycles/bit)
$T_i^{\max}$	0.3	任务最大容忍延迟 (s)
g	−40	路径损耗常数 (dBm/Hz)
p_i^{tran}	1	终端车辆的传输功率 (W)
GT	0.5	初始信任值
γ	0.95	折扣因子
λ	0.97	GAE参数
buffer_size	10 000	回放缓冲区大小
batch_size	128	样本量

3.1.2 对比实验

为了评估信任模型的性能, 与文献[17–19]方案进行对比.

为了评估任务卸载的性能, 与以下方案进行比较.

(1) DQN: 是一种将深度神经网络与 Q-learning 相结合的强化学习方法, 通过神经网络逼近 Q 值函数, 从而实现策略的学习与优化[23].

(2) D3QN: 是一种结合了 Double DQN 和 Dueling network 结构的深度强化学习方法, 通过分离状态价值函数和动作优势函数, 从而提升学习效率并减轻 Q 值高估问题[24].

(3) TASACO: 是一种基于最大熵和信任感知的离策略深度强化学习方法, 通过优化策略的期望回报和熵, 实现高效稳定的策略学习[25].

3.2 实验结果分析

3.2.1 信任模型性能评估

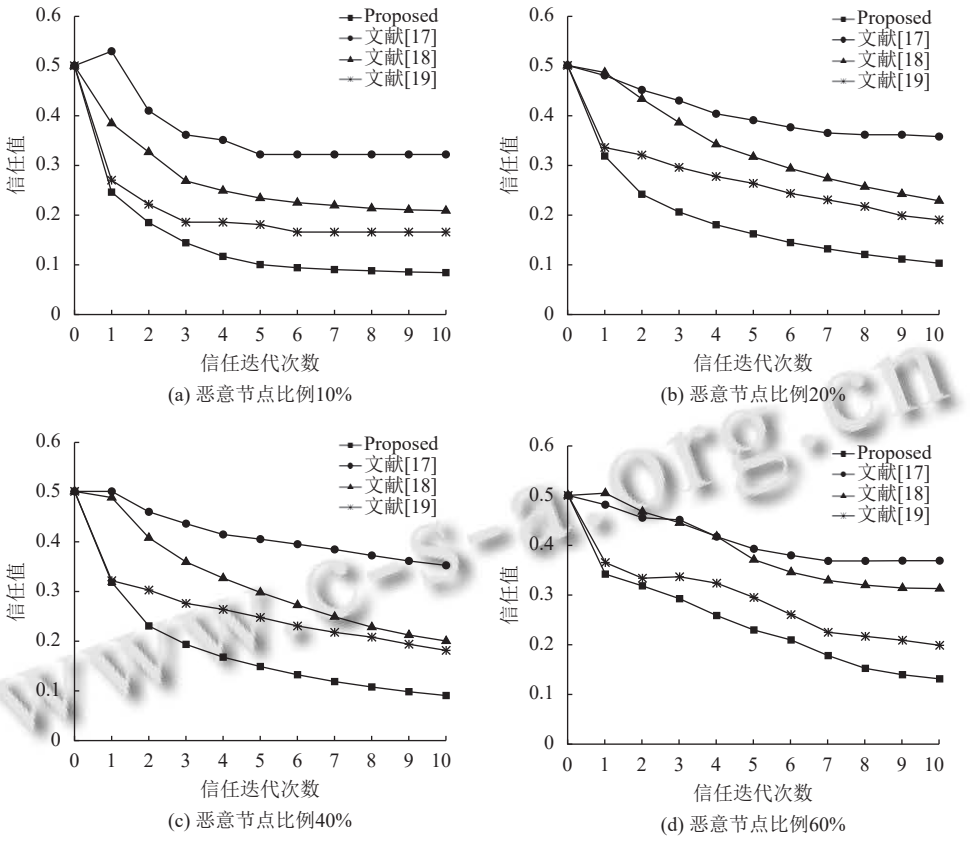
为了验证本文所提出的动态信任模型的有效性, 从局部性能的交互信任值和全局性能的任务失败率两个维度对信任模型进行了评估. 在仿真环境中, 设置了 100 台终端车辆、10 个 RSU 和 10 个空闲服务车辆共计 20 个边缘节点, 并模拟了恶意攻击行为, 恶意边缘节点的比例分别设定为 10%、20%、40% 和 60%. 为了证明本文的动态信任模型的有效性, 将其与文献[17–19]中提出的信任模型进行对比. 这些信任模型均基于边缘计算环境, 具有较高的代表性和可比性.

(1) 局部性能分析

在系统中边缘节点遭到虚假反馈攻击的场景下, 随着信任迭代次数的增加, 各类模型的信任值变化趋势如图 3 所示. 从图 3(a)–(d) 中可以看出, 在边缘节点遭到虚假反馈攻击时, 本文所提方案能够迅速识别恶意节点, 并使其信任值迅速下降, 最终稳定在较低水平. 相比其他方法, 本文通过融合多维赋权和动态奖惩机制, 在恶意节点频繁扰动的情况下, 该模型能精准识别异常行为并加速降低其信任值, 具有高敏感性和抑制性.

(2) 全局性能分析

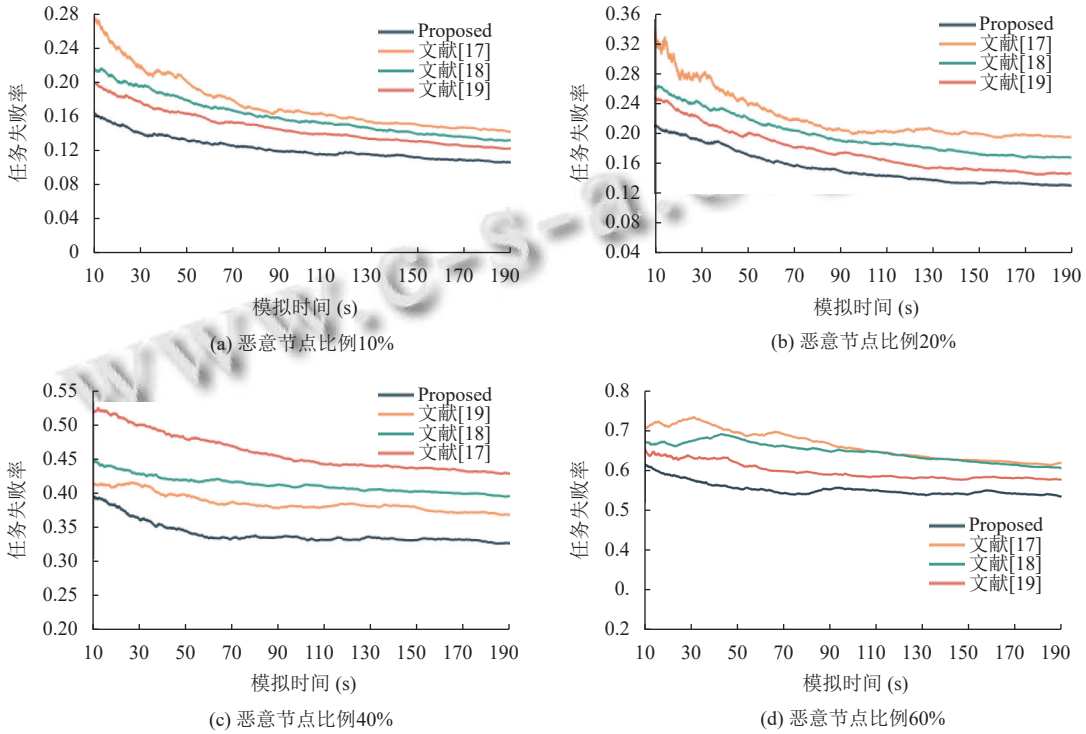
任务失败率是指终端车辆所生成的计算任务被卸载至恶意边缘节点处理的任务数占总任务数的比例. 该指标能够有效反映信任模型在识别和规避恶意节点方面的可靠性. 任务失败率越低, 说明信任评估模型在保障系统安全性和任务成功率方面表现越优越. 在系统中边缘节点遭到摇摆攻击的场景下, 各类模型的任务失败率的对比曲线如图 4 所示.



(a) 恶意节点比例10% (b) 恶意节点比例20%

(c) 恶意节点比例40% (d) 恶意节点比例60%

图 3 恶意边缘节点信任值



(a) 恶意节点比例10% (b) 恶意节点比例20%

(c) 恶意节点比例40% (d) 恶意节点比例60%

图 4 各类模型任务失败率

在图 4(a) 中, 恶意边缘节点的比例设置为 10% 时, 所提方案的任务失败率相较于文献[17]最大降低了 10.62%, 最小降低了 3.54%. 在图 4(b) 中, 恶意边缘节点的比例设置为 20% 时, 所提方案的任务失败率相较于文献[17]最大降低了 13.07%, 最小降低了 6.46%. 在图 4(c) 中, 恶意边缘节点的比例设置为 40% 时, 所提方案的任务失败率相较于文献[17]最大降低了 12.07%, 最小降低了 10.18%. 在图 4(d) 中, 恶意边缘节点的比例设置为 60% 时, 所提方案的任务失败率相较于文献[17]最大降低了 9.33%, 最小降低了 8.36%. 综上, 本文提出的方案通过融合多源赋权与动态奖惩机制, 有效保障了信任评估的准确性, 激励了可靠行为并抑制恶意行为. 这得益于多源赋权能过滤虚假反馈, 提升信任评估的可靠性, 奖惩机制能够对节点的行为变化做出快速响应, 对正常节点加速信任值的积累, 对恶意节点施加惩罚并迅速降低其信任值. 在摇摆攻击环境下, 二者的结合既保证了信任评估的全面性, 又提升了对恶意行为的敏感性, 从而实现对恶意节点与正常节点的有效区分, 使任务卸载目标集中于高可信度节点, 减少因恶意服务导致的任务执行失败, 有效增强了车载边缘计算环境的稳定性与安全性.

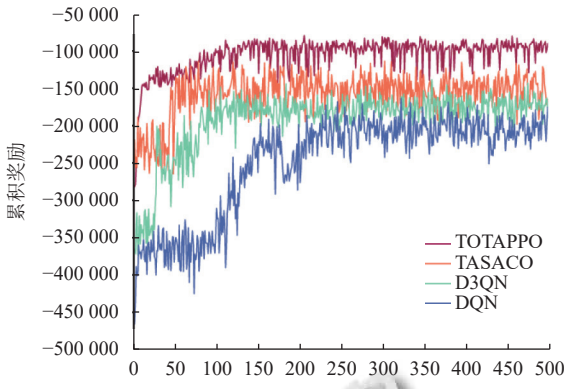
3.2.2 算法收敛性能评估

为了评估本文提出的基于信任感知和 PPO 算法的任务卸载模型 (TOTAPPO) 的收敛性能, 本文把累积奖励作为收敛性能的指标进行对比.

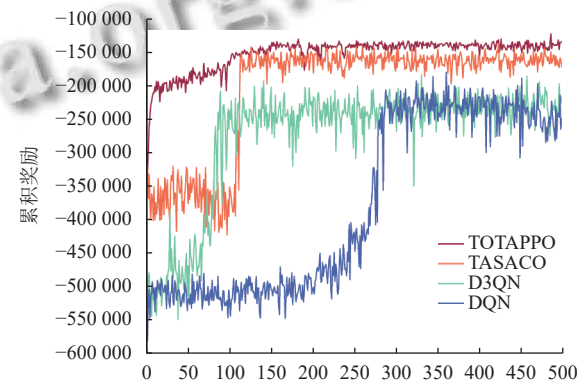
(1) 测试场景 1. 5 个 RSU 和 5 辆空闲服务车辆共计 10 个边缘节点, 300 辆终端车辆, 累积奖励的收敛曲线如图 5(a) 所示.

(2) 测试场景 2. 20 个 RSU 和 10 辆空闲服务车辆共计 30 个边缘节点, 500 辆终端车辆, 累积奖励的收敛曲线如图 5(b) 所示.

从图 5 中可以看出, 不同算法在训练过程中的累积奖励的变化趋势存在明显差异. 在图 5(a) 中, DQN 算法在初始阶段收敛速度较慢, 约经过 230 次迭代后才收敛. DQN 和 D3QN 算法在后期易陷入局部最优, 累积奖励波动较大, 整体稳定性较差. 相比之下, TASACO 和 TOTAPPO 算法在训练初期就展现出更快的收敛速度和更强的策略探索能力. 其中, TASACO 算法的累积奖励稳定在−150 000 左右, 而 TOTAPPO 算法的累积奖励稳定在−100 000 左右, 在收敛质量和稳定性方面均显著优于其他算法. 在图 5(b) 中可以看出, 测试场景扩大后, DQN 算法的收敛速度依旧最慢, 约经过 280 次迭代后收敛. D3QN 算法的收敛速度最快, 但陷入局部最优, 其累积奖励稳定在−250 000 左右. 相比之下, TASACO 算法和 TOTAPPO 算法稳定性较好, 收敛速度较快, 其中, TOTAPPO 算法在累积奖励的收敛效果优于 TASACO 算法. 综上, TOTAPPO 算法在 VEC 环境中表现出更优越的性能. 这是因为 PPO 算法引入剪切概率比机制, 通过限制新旧策略之间的变化幅度, 避免策略在更新过程中发生剧烈波动, 从而有效提升了训练的稳定性. 此外, 本文在 PPO 框架中融合了对边缘节点可信度的评估机制, 实现了对恶意节点的有效识别与抑制. 这使得 TOTAPPO 在应对车联网环境中的高动态和恶意攻击行为方面展现出良好的适应性与鲁棒性.



(a) 测试场景1算法的收敛性能



(b) 测试场景2算法的收敛性能

图 5 TOTAPPO 模型的收敛性能

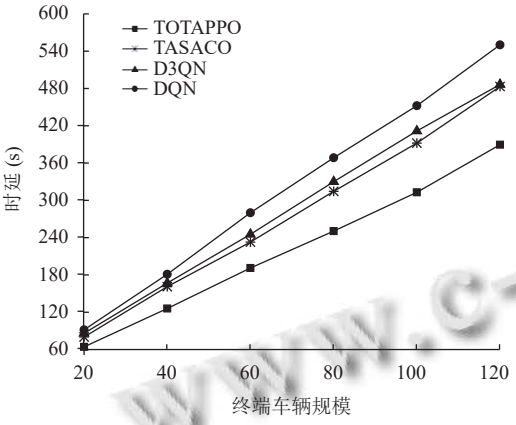
3.2.3 任务卸载性能评估

(1) 不同终端车辆规模下的时延和能耗对比

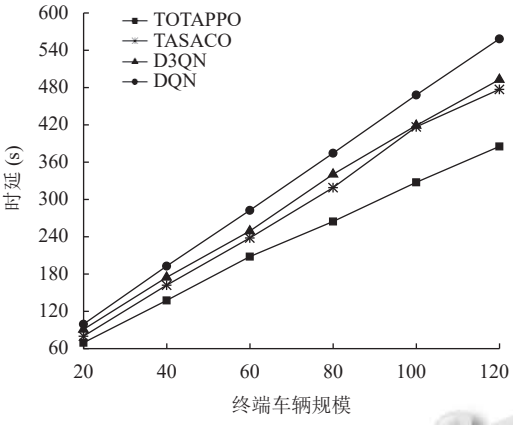
仿真环境设置 10 个 RSU 和 20 辆空闲服务车辆, 恶意边缘节点占比设置为 10%、20%、40%, 终端车辆

规模设置为 20、40、60、80、100、120.

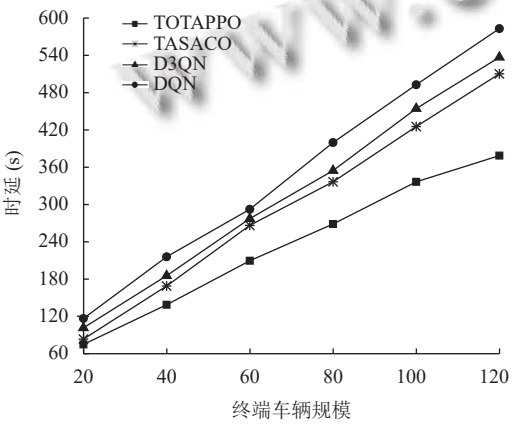
在不同终端车辆规模下, 时延和能耗的对比结果如图 6 和图 7 所示. 实验结果表明, 随着终端车辆规模的增加, 4 种方法的时延、能耗整体呈上升趋势, 这是由于边缘节点的计算资源被更多任务共享, 从而导致延迟和能耗的增加.



(a) 恶意节点比例10%

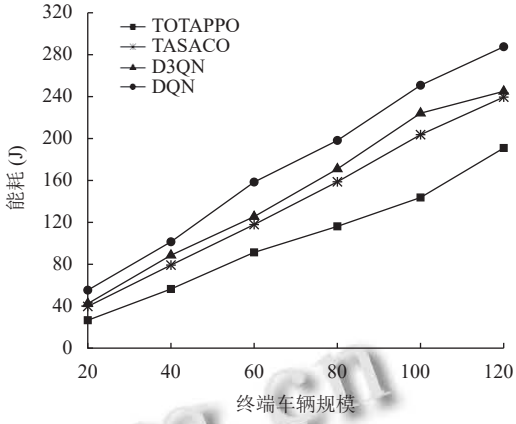


(b) 恶意节点比例20%

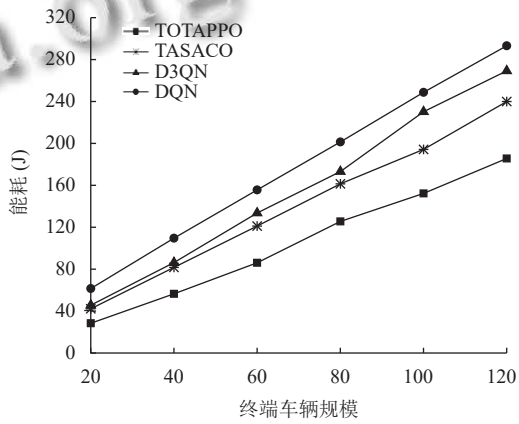


(c) 恶意节点比例40%

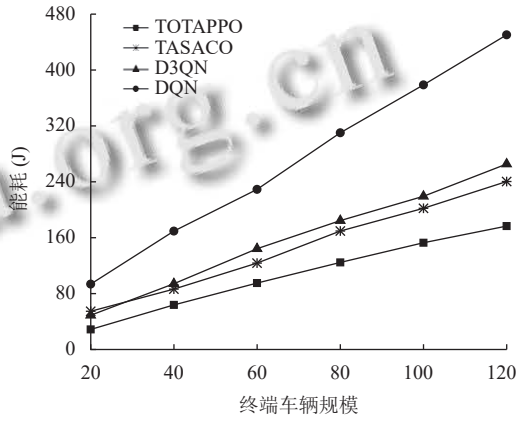
图 6 不同终端车辆规模下的时延



(a) 恶意节点比例10%



(b) 恶意节点比例20%



(c) 恶意节点比例40%

图 7 不同终端车辆规模下的能耗

从图 6 和图 7 中可进一步观察到, 随着恶意边缘节点数量的增加, 时延和能耗也在不断上升, 这主要是因为恶意节点的存在导致可用计算资源减少, 任务卸载受限, 从而系统整体时延和能耗增加. 在相同终端车辆规模下, DQN 算法的时延和能耗始终处于较高水平; D3QN 算法相较于 DQN 算法减少了 Q 值的高估, 充分利用边缘节点, 在一定程度上减少了时延和能耗. TA-

SACO 评估了边缘节点的可靠性, 减少因恶意节点导致的资源浪费, 进一步降低了系统的时延和能耗. TOTAPPO 算法不仅充分利用空闲边缘节点, 还考虑了节点的可信度, 有效解决了恶意节点干扰和资源浪费的问题, 且 PPO 算法通过限制策略更新幅度以及样本重复利用的策略, 提高了训练的稳定性和策略优化效率, 从而在复杂动态车联网环境中显著提升任务卸载的时延与能耗.

(2) 不同任务大小的平均时延和平均能耗对比

仿真环境设置 8 个 RSU 和 8 辆空闲服务车辆, 任务大小设置为{1.0, 1.1, 1.2, 1.3, 1.4, 1.5} Mb.

在不同任务大小下的平均时延和平均能耗对比结果如图 8 和图 9 所示.

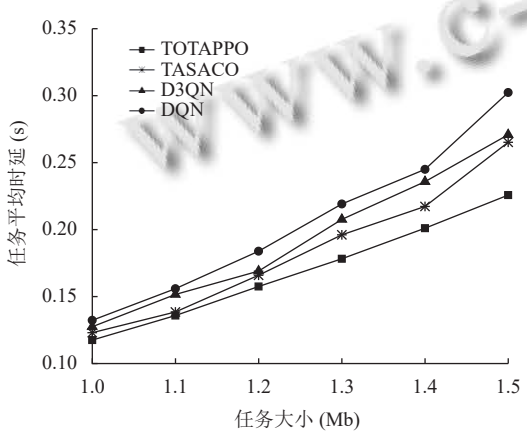


图 8 不同任务大小下的平均时延

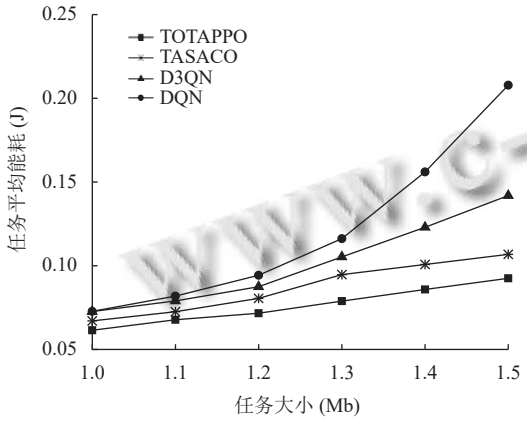


图 9 不同任务大小下的平均能耗

实验结果表明, 随着任务大小的增加, 4 种方法的任务平均时延和能耗呈上升趋势, 主要原因在于数据量增大导致任务在传输和计算过程中消耗更多资源. 在相同数据规模下, TOTAPPO 始终处于较低的平均时延、能耗. 其中, TOTAPPO 的任务平均时延相较于 DQN 方法最少下降 12.56%, 最多下降 33.95%; TOTAPPO 的任务平均能耗相较于 TASACO 方法最少下降 9.2%, 最多下降 17.45%. 这得益于 V2V 链路的协同负载, 以及动态反馈信任模型和近端策略优化算法的结合, 智能体能够更合理地选择高可信、低延迟的边缘节点进行任务卸载决策.

4 结论与展望

本文针对 VEC 中可信任务卸载的关键问题, 提出了一种基于信任感知和 PPO 算法的任务卸载模型. 首先利用空闲服务车辆的计算资源, 构建 V2V 通信链路协同负载, 缓解路侧单元的压力. 其次, 构建融合多源赋权和奖惩机制的动态反馈信任模型, 准确评估边缘节点的可信度. 最后, 将任务卸载建模为 MDP, 利用 PPO 算法优化目标函数. 仿真结果表明, 该模型在提高卸载效率的同时提升了系统的安全性, 为 VEC 任务卸载决策提供了可行方案.

在未来的工作中, 考虑到当前系统模型在通信链路方面仍面临安全问题, 且车联网中存在大量隐私敏感的数据, 未来可以专注于联邦学习机制, 实现分布式隐私保护下的模型协同训练, 提升 VEC 系统的安全性.

参考文献

1 赵振博, 任雪容, 付青坤. 改进鲸鱼优化算法的车联网计算卸载. 计算机系统应用, 2024, 33(4): 123–132. [doi: 10.15888/j.cnki.csa.009478]

2 Wang X, Lv JH, Slowik A, *et al*. Augmented intelligence of things for priority-aware task offloading in vehicular edge computing. IEEE Internet of Things Journal, 2024, 11(22): 36002–36013. [doi: 10.1109/JIOT.2024.3408157]

3 Fan WH, Su Y, Liu J, *et al*. Joint task offloading and resource allocation for vehicular edge computing based on V2I and V2V modes. IEEE Transactions on Intelligent Transportation Systems, 2023, 24(4): 4277–4292. [doi: 10.1109/TITS.2022.3230430]

4 Dai XX, Xiao Z, Jiang HB, *et al*. UAV-assisted task offloading in vehicular edge computing networks. IEEE Transactions on Mobile Computing, 2024, 23(4): 2520–2534. [doi: 10.1109/TMC.2023.3259394]

5 Zhou YB, Chai ZY, Li YL, *et al*. Parking vehicle-assisted task offloading in edge computing: A dynamic multi-objective evolutionary algorithm with multi-strategy fusion response. Swarm and Evolutionary Computation, 2025, 94:

101900. [doi: 10.1016/j.swevo.2025.101900]
6 Gu K, Liu ZL, Jia WJ. Location-aware reliable task cooperative-computation scheme under fog computing-based IoVs. IEEE Transactions on Intelligent Transportation Systems, 2025, 26(1): 425–442. [doi: 10.1109/TITS.2024.3485241]
7 Guo HZ, Chen XS, Zhou XY, *et al*. Trusted and efficient task offloading in vehicular edge computing networks. IEEE Transactions on Cognitive Communications and Networking, 2024, 10(6): 2370–2382. [doi: 10.1109/TCCN.2024.3412394]
8 王亚丽, 娄世豪. 基于改进遗传算法的可信边缘计算任务卸载方法. 郑州大学学报 (理学版), 2025, 57(3): 72–80. [doi: 10.13705/j.issn.1671-6841.2023246]
9 Zhao W, Shi K, Liu Z, *et al*. DRL connects Lyapunov in delay and stability optimization for offloading proactive sensing tasks of RSUs. IEEE Transactions on Mobile Computing, 2024, 23(7): 7969–7982. [doi: 10.1109/TMC.2023.3342102]
10 何杰, 马强. 基于深度强化学习的 C-V2X 任务卸载研究. 计算机工程, 2024, 50(12): 200–212. [doi: 10.19678/j.issn.1000-3428.0068425]
11 Chen Y, Li KX, Wu Y, *et al*. Energy efficient task offloading and resource allocation in air-ground integrated MEC systems: A distributed online approach. IEEE Transactions on Mobile Computing, 2024, 23(8): 8129–8142. [doi: 10.1109/TMC.2023.3346431]
12 Zhou H, Wang ZN, Min GY, *et al*. UAV-aided computation offloading in mobile-edge computing networks: A Stackelberg game approach. IEEE Internet of Things Journal, 2023, 10(8): 6622–6633. [doi: 10.1109/JIOT.2022.3197155]
13 Suzuki A, Kobayashi M, Oki E. Multi-agent deep reinforcement learning for cooperative computing offloading and route optimization in multi cloud-edge networks. IEEE Transactions on Network and Service Management, 2023, 20(4): 4416–4434. [doi: 10.1109/TNSM.2023.3267809]
14 王晨旭, 王凯月, 王梦勤. 基于半监督和自监督图表示学习的恶意节点检测. 软件学报, 2025, 36(5): 2288–2307. [doi: 10.13328/j.cnki.jos.007211]
15 Zhang Y, Zhu KG, Zhao X, *et al*. Research on resource allocation technology in highly trusted environment of edge computing. Journal of Parallel and Distributed Computing, 2023, 178: 29–42. [doi: 10.1016/j.jpdc.2023.03.011]
16 Zhang LS, Guo HZ, Zhou XY, *et al*. Trusted task offloading in vehicular edge computing networks: A reinforcement learning based solution. Proceedings of the 2023 IEEE Global Communications Conference. Kuala Lumpur: IEEE, 2023. 6711–6716. [doi: 10.1109/GLOBECOM54140.2023.10437191]
17 Kong WP, Li XY, Hou LY, *et al*. A reliable and efficient task offloading strategy based on multifeedback trust mechanism for IoT edge computing. IEEE Internet of Things Journal, 2022, 9(15): 13927–13941. [doi: 10.1109/JIOT.2022.3143572]
18 石琼, 段辉, 师智斌. 基于深度强化学习的可信任务卸载方案. 计算机工程, 2024, 50(8): 142–152. [doi: 10.19678/j.issn.1000-3428.0069352]
19 Yu Y, Lu QC, Fu YS. Dynamic trust management for the edge devices in industrial Internet. IEEE Internet of Things Journal, 2024, 11(10): 18410–18420. [doi: 10.1109/JIOT.2024.3361914]
20 Wang X, Wang SB, Gao X, *et al*. AMTOS: An ADMM-based multilayer computation offloading and resource allocation optimization scheme in IoV-MEC system. IEEE Internet of Things Journal, 2024, 11(19): 30953–30964. [doi: 10.1109/JIOT.2024.3416171]
21 Schulman J, Wolski F, Dhariwal P, *et al*. Proximal policy optimization algorithms. arXiv:1707.06347, 2017.
22 Schulman J, Moritz P, Levine S, *et al*. High-dimensional continuous control using generalized advantage estimation. arXiv:1506.02438, 2018.
23 Gao J, Kuang ZF, Gao J, *et al*. Joint offloading scheduling and resource allocation in vehicular edge computing: A two layer solution. IEEE Transactions on Vehicular Technology, 2023, 72(3): 3999–4009. [doi: 10.1109/TVT.2022.3220571]
24 Wu HN, Yang XM, Bu ZY. Task offloading with service migration for satellite edge computing: A deep reinforcement learning approach. IEEE Access, 2024, 12: 25844–25856. [doi: 10.1109/ACCESS.2024.3367128]
25 孔文萍. 面向边缘计算的可信资源分配与任务卸载的关键技术研究 [博士学位论文]. 北京: 北京邮电大学, 2022.

(校对责编: 张重毅)