基于 DRL 的联邦学习节点选择方法

贺文晨[1]，郭少勇[1]，邱雪松[1]，陈连栋[2]，张素香[3]

（1. 北京邮电大学网络与交换技术国家重点实验室，北京 100876； 2. 国网河北信息通信分公司，河北 石家庄 050011；
3. 国家电网有限公司信息通信分公司，北京 100761）

摘 要：为了应对设备差异化计算能力及非独立同分布数据对联邦学习性能的影响，高效地调度终端设备完成模型聚合，提出了一种基于深度强化学习的设备节点选择方法。该方法考虑异构节点的训练质量和效率，筛选恶意节点，在提升联邦学习模型准确率的同时，优化训练时延。首先，根据联邦学习中模型分布式训练的特点，构建基于深度强化学习的节点选择系统模型。其次，考虑设备训练时延、模型传输时延和准确率等因素，提出面向节点选择的准确率最优化问题模型。然后，将问题模型构建为马尔可夫决策过程，并设计基于分布式近端策略优化的节点选择算法，在每次训练迭代前选择合理的设备集合完成模型聚合。仿真实验表明，所提方法显著提高了联邦学习的准确率和训练速度，且具有良好的收敛性和稳健性。

关键词：联邦学习；模型聚合；节点选择；深度强化学习；准确率

中图分类号：TP911.1

文献标识码：A

DOI: 10.11959/j.issn.1000−436x.2021111

Node selection method in federated learning based on deep reinforcement learning

HE Wenchen[1], GUO Shaoyong[1], QIU Xuesong[1], CHEN Liandong[2], ZHANG Suxiang[3]

1. State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
2. Hebei State Grid Information & Telecommunication Branch, Shijiazhuang 050011, China
3. State Grid Information & Telecommunication Branch, Beijing 100761, China

Abstract: To cope with the impact of different device computing capabilities and non-independent uniformly distributed data on federated learning performance, and to efficiently schedule terminal devices to complete model aggregation, a method of node selection based on deep reinforcement learning was proposed. It considered training quality and efficiency of heterogeneous terminal devices, and filtrate malicious nodes to guarantee higher model accuracy and shorter training delay of federated learning. Firstly, according to characteristics of model distributed training in federated learning, a node selection system model based on deep reinforcement learning was constructed. Secondly, considering such factors as device training delay, model transmission delay and accuracy, an optimization model of accuracy for node selection was proposed. Finally, the problem model was constructed as a Markov decision process and a node selection algorithm based on distributed proximal strategy optimization was designed to obtain a reasonable set of devices before each training iteration to complete model aggregation. Simulation results demonstrate that the proposed method significantly improves the accuracy and training speed of federated learning, and its convergence and robustness are also well.

Keywords: federated learning, model aggregation, node selection, deep reinforcement learning, accuracy

收稿日期：2021−01−05；**修回日期**：2021−04−16

通信作者：邱雪松，xsqiu@bupt.edu.cn

基金项目：国家自然科学基金资助项目（No.62071070）；教育部区块链核心计划基金资助项目（No.2020KJ010802）；河北省重点研发计划基金资助项目（No.20310103D）

Foundation Items: The National Natural Science Foundation of China (No.62071070), Key Project Plan of Blockchain in Ministry of Education of the People's Republic of China (No.2020KJ010802), The Key Research and Development Program of Hebei Province (No.20310103D)

1　引言

随着边缘智能[1]概念的提出，越来越多的智能化应用将在边缘侧训练和执行。传统的云智能[2]采用将原始数据上传至云中心进行模型训练的方式，存在高传输时延、用户隐私泄露等弊端。为解决这一问题，基于联邦学习（FL, federated learning）的分布式模型训练架构应运而生。

在基于 FL 的分布式训练架构下，边缘侧终端设备可以利用自身采集数据在本地执行训练任务，然后将训练好的本地模型参数上传至云服务器进行模型聚合。相比直接上传原始训练数据，该架构选择上传训练之后的模型参数，能有效降低数据传输成本，同时保护用户隐私[3]。然而，终端设备上的数据集大小往往是不同的，数据也可能不满足独立同分布特性，这使本地模型的训练质量存在差异[4]。同时，边缘侧终端设备并不是完全可信的，存在一些恶意节点篡改训练结果，上传错误参数进而降低 FL 性能。此外，终端设备多样异构的计算资源和传输时间对 FL 的效率也具有较大影响[5]。因此，如何合理选择设备集合参与模型聚合，以提高 FL 效率和准确率成为一个亟待解决的问题。

由于能提供有效的隐私保护和高效的模型训练方式，FL 得到了越来越多的关注。Shi 等[6]提出了一种带宽分配和设备调度的联合优化模型，并通过解耦为 2 个子问题来提高 FL 效率，但该方法仅根据训练时间来选择设备，忽略了设备的本地训练质量。Ren 等[7]设计了一个新的概率调度框架来调度多个边缘设备参与 FL 模型聚合，该框架能有效提高模型训练的准确率，但是对设备异质的计算能力和训练时间考虑不足，可能会导致较大的时延。Chen 等[8]构建了一个无线资源分配和节点选择的联合优化问题，并提出了一种依概率选择节点的方法。Wu 等[9]设计了一个多层 FL 协议，依概率引入区域松弛因子后完成节点选择。但上述方案依赖概率进行节点选择，忽略了节点本身计算、通信能力等方面的差异。Kang 等[10]引入声誉作为衡量移动设备可靠性和可信度的指标，并设计了一个基于声誉的可靠 FL 设备选择方案，从而有效地保证模型精度和可靠性。Lu 等[11]揭示了本地训练方法和不进行节点筛选的 FL 训练方法在训练精度和时延等方面的不足，在此基础上提出了一种用于车联网中资源共享的 FL 方案，该方案综合考虑训练时间和精度，通过选择精确度高、训练速度快的设备完成模型聚合。但上述方法均忽略了非独立同分布数据带来的影响。Yoshida 等[12]考虑非独立同分布数据对训练性能的影响，设计了启发式算法解决终端设备和数据选择问题，但其节点选择算法的性能还有待改进。此外，由于资源分配和能耗管理也对 FL 性能有很大影响，有许多针对这方面的研究工作已陆续展开[13-16]，通过优化终端设备的无线、计算资源分配和能耗来支撑 FL。但上述工作偏向于提高资源利用率及设备节能，难以兼顾 FL 本身性能。另一方面，在针对诸如节点选择等 NP 问题时，孟洛明等[17]基于禁忌搜索算法进行求解，并在有限时间内获取近似最优解。李枝灵等[18]设计了一种基于免疫算法的接入点选择方法，以提高求解效率。但上述方法缺少学习能力，难以适应复杂且动态变化的边缘网络环境。已有许多文献[19-21]采用如 Q 学习、深度 Q 网络等深度学习算法进行求解，但这些方法存在学习率确定难、收敛速度慢等问题。因此，在 FL 的设备节点选择过程中，仍存在以下问题需要进一步解决：1) 忽略终端设备异构的数据质量及训练能力；2) 面对复杂动态的网络环境，缺乏高效的方法获取最优节点集合。

为解决以上问题，本文主要的研究工作如下。

1) 首先，建立了基于深度强化学习（DRL, deep reinforcement learning）的 FL 分布式训练系统架构，实现恶意节点的筛查和异构设备节点的选择。其次，构建面向节点选择的准确率最优化问题模型，该问题以最小化每次 FL 迭代过程中参与设备的总体损失函数为目标，并满足包含传输和计算时延的约束。

2) 设计了基于分布式近端策略优化（DPPO, distributed proximal policy optimization）的节点选择算法。将 FL 中设备节点选择问题构建为马尔可夫决策过程（MDP, Markov decision process），定义动作、状态空间和奖励函数。基于多线程和 PPO 算法思想，设计了基于 DPPO 的节点选择算法对优化问题进行求解。

3) 基于多种数据集和多样化训练任务，对所提最优化问题模型和算法进行了仿真实验验证。结果表明，本文所提模型和算法在面对差异化数据质量和设备训练能力时，具有更好的准确率和时延性能，同时有良好的收敛性和稳健性。

2 系统模型

本文构建的系统架构如图 1 所示。FL 任务实现流程主要包括模型的本地训练、参数上传、模型聚合以及参数下发。与传统 FL 分布式训练架构不同，本文基于 DRL 的节点选择对模型聚合模块进行改进，在权值聚合之前，基于 DRL 的节点选择能合理选择具备计算能力强、训练质量高的设备参与模型聚合，进而有效提高 FL 性能。

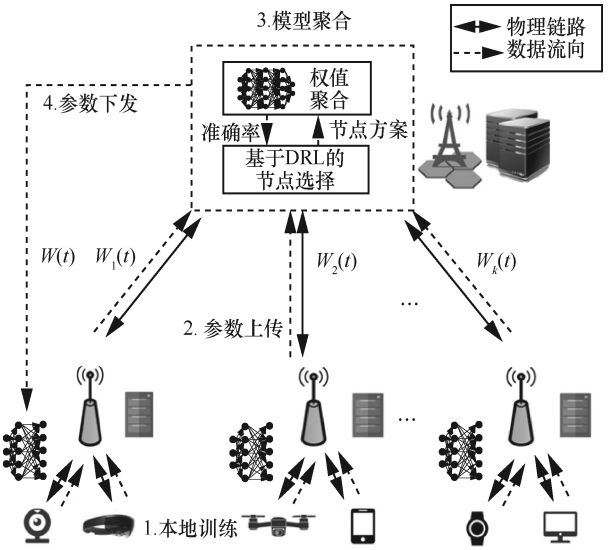


图 1 基于 DRL 的 FL 架构

2.1 网络架构

网络由终端设备、微基站、宏基站和对应的移动边缘计算（MEC, mobile edge computing）服务器组成。宏基站内的 MEC 服务器具有强大的计算和通信资源。令 Z 表示微基站内 MEC 服务器集合，每一个 MEC 服务器 $z \in Z$ 具有一定的计算能力，并通过与其相连的基站来覆盖数个终端设备。终端设备的集合用 D 表示，令 $H_{z,d}=\{x_{z,d},y_{z,d}\}$ 表示被 MEC 服务器 z 覆盖的终端 d 的数据集。针对诸如路径选择、图像识别等学习任务 $i \in I$，其目的是从终端设备的数据集合 $H_{z,d}=\{x_{z,d},y_{z,d}\}$ 中学习与任务相关的模型 M。本文定义 FL 任务 i 的属性集合为 $\Omega_i=\{Z_i,D_i,C_i,M_i^0\}$，其中，$Z_i$ 和 D_i 分别表示与任务 i 相关的 MEC 服务器和终端设备的集合，C_i 为该 FL 模型计算数据集中一组数据所需的 CPU 周期数，M_i^0 为该 FL 任务的初始模型。具体系统参数设置如表 1 所示。

2.2 FL 训练机制

本地训练。对于一个 FL 任务 $i \in I$，定义与该任务相关的总数据集为

$$H_i=\sum_{z\in Z_i}\sum_{d\in D_i}H_{z,d} \tag{1}$$

表 1 系统参数

参数	含义
Z	MEC 服务器集合
H_i	FL 任务 i 相关的总数据集
$H_{z,d}$	MEC 服务器 z 覆盖的终端 d 的数据集
$\lvert H_i\rvert$	FL 任务 i 相关的总数据集大小
Ω_i	FL 任务 i 的属性集合
M_i^0	FL 任务 i 的初始模型
C_i	FL 任务 i 计算一组数据所需的 CPU 周期数
$l_{z,d}^i$	设备 d 在进行 FL 任务 i 训练时的损失函数
$\omega_{z,d}^n$	设备 d 在进行第 n 次训练时的模型参数
A_i	FL 任务 i 测试数据集的损失函数之和
r_i^d	FL 任务 i 在设备 d 与微基站间的传输速率
r_i^z	FL 任务 i 在微基站 z 与汇聚服务器间的传输速率
$c_{z,d}$	设备 d 执行 FL 任务时的 CPU 频率
t_i^{tra}	FL 任务 i 的总传输时延
t_i^{com}	FL 任务 i 在设备上的计算时延

终端设备 d 在执行 FL 任务 i 的本地训练时的损失函数 $l_{z,d}^i(x_{z,d},y_{z,d};\omega_{z,d})$ 定义为它在样本数据集 $H_{z,d}$ 上的预测值与实际值之差，因此 FL 任务 i 在所有数据集上的损失函数可以定义为

$$L^i(\omega)=\frac{1}{|H_i|}\sum_{z\in Z_i}\sum_{d\in D_i}l_{z,d}^i(x_{z,d},y_{z,d};\omega_{z,d}) \tag{2}$$

其中，ω 表示当前要训练的模型的权值，$|H_i|$ 表示该任务数据集大小。FL 的目的是通过最小化任务的损失函数 $L^i(\omega)$ 来优化全局模型参数，表示为

$$\omega=\arg\min L^i(\omega) \tag{3}$$

本文的 FL 的参数更新方法为随机梯度下降（SGD, stochastic gradient descent），即每次随机选择数据集中的一条数据 $\{x_{z,d},y_{z,d}\}$ 进行更新。这种方法大大降低了计算量，但由于其随机性使本地模型需要进行足够的本地训练量以保证模型质量。模型参数的更新表示为

$$\omega_{z,d}^n=\omega_{z,d}^{n-1}-\eta\nabla l(\omega_{z,d}^{n-1}) \tag{4}$$

其中，η 表示参数更新时的学习率，$n \in N$ 表示训练的迭代次数。

模型聚合。当上传的本地模型达到一定数量

或者迭代次数 N 后，宏基站处的 MEC 服务器将对得到的本地模型执行全局模型聚合，具体的权值聚合表示为

$$\omega_g' = \omega_g + \sum_{z\in Z_i}\sum_{d\in D_i}\frac{\left|H_{z,d}\right|\left(\omega_{z,d}' - \omega_{z,d}\right)}{\left|H_i\right|} \tag{5}$$

其中，$\left|H_{z,d}\right|$ 表示终端设备 d 参与 FL 任务的数据集大小。可以看出，具备更大数据集的终端设备能得到更大的权值。

2.3　节点选择问题描述

设备节点的选择受诸多因素影响。首先，终端设备差异化的计算和通信能力直接影响本地训练和数据传输时延。其次，终端设备上携带的数据集大小不同，数据也可能不满足独立同分布的特性，这使本地模型的训练质量存在差异。因此，本文构建了面向节点选择的准确率最优问题模型。

准确率。对于一个 FL 任务 $i\in I$，其训练质量定义为聚合后的全局模型在测试数据集上的测试准确率，本文使用测试数据集的损失函数之和表示测试准确率，即

$$A_i = L^i(x_{\text{test}}, y_{\text{test}}; \omega_g) \tag{6}$$

时延。FL 每一次模型聚合的总时延包括数据在终端设备上的训练时延和在链路上的传输时延。FL 任务 i 的参数数据在终端设备与微基站间以及微基站与宏基站间传输速率可分别表示为

$$\begin{aligned} r_i^d &= B_d \text{lb}\left(1+\frac{p_d G_d}{N_0 B_d}\right), d\in D_i \\ r_i^z &= B_z \text{lb}\left(1+\frac{p_z G_z}{N_0 B_z}\right), z\in Z_i \end{aligned} \tag{7}$$

其中，B_d 和 B_z 分别表示设备与微基站间以及微基站与宏基站间的可用带宽，G_d 和 G_z 分别表示设备和微基站的信道增益，p_d 和 p_z 分别表示设备和微基站的发射功率，N_0 表示噪声功率谱密度。

因此，设备将本地参数上传至模型汇聚服务器的总传输时间为

$$t_i^{\text{tra}} = \frac{\left|\omega_{z,d}'\right|}{r_i^d} + \frac{\left|\omega_{z,d}'\right|}{r_i^z}, z\in Z_i, d\in D_i \tag{8}$$

其中，$|\omega_{z,d}'|$ 表示终端设备 d 待上传的本地模型参数大小。终端设备的计算时延可以表示为

$$t_i^{\text{com}} = \frac{\left|H_{z,d}\right| C_i}{c_{z,d}}, z\in Z_i, d\in D_i \tag{9}$$

其中，$\left|H_{z,d}\right| C_i$ 表示完成终端 d 上的 FL 任务 i 所需的 CPU 周期数目，$c_{z,d}$ 表示终端设备执行 FL 任务时的 CPU 频率。FL 每一学习回合的总时延由时延最大的终端设备决定，因此，总时延定义为

$$T_i = \max_{z\in Z_i, d\in D_i}\left(t_i^{\text{com}} + t_i^{\text{tra}}\right) \tag{10}$$

综上，面向节点选择的准确率最优化问题模型可以表示为

$$\begin{aligned} &\min\left\{\frac{1}{|I|}\sum_{i\in I} A_i\right\} \\ &\text{s.t. } T_i \leqslant T_{\text{req}}, i\in I \end{aligned} \tag{11}$$

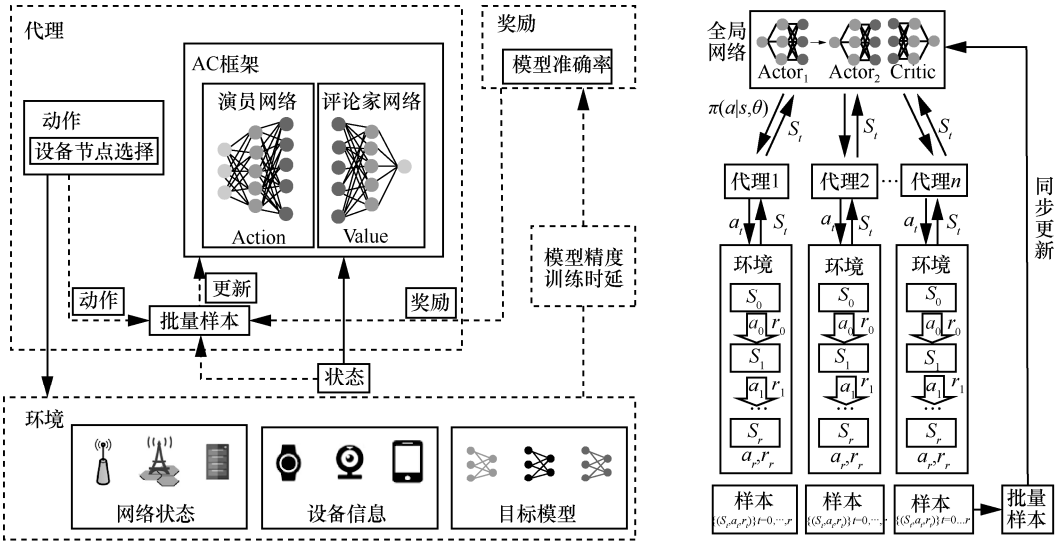
对于一个 FL 任务 $i\in I$，节点选择问题可以概括为每次迭代时选择节点集 $D_i\in D$，使本次训练的准确率最优，即总损失函数最小，同时将训练和传输时延控制在一定范围内。可以看出，上述问题属于典型的 NP 问题。

3　基于 DRL 的 FL 节点选择方法

3.1　算法机理描述

在复杂多变的边缘网络中，节点选择策略需要随着环境状态信息的变化而发生改变，基于 DRL 的节点选择框架能通过不断与环境的交互，学习节点选择策略以获得最大回报[22-23]。本文提出的基于 DRL 的节点选择框架如图 2(a)所示，包括 3 个部分：环境、代理和奖励。环境主要包括网络状态、终端设备以及目标模型信息。代理与环境进行交互，从一个状态出发，根据自己的策略分布选择动作，并获得奖励。代理获得的动作、奖励及环境状态组成批量样本来更新演员−评论家（AC, actor-critic）网络。

边缘网络中参与 FL 训练的终端设备往往数量众多，在应对节点选择问题时，传统的 AC 算法由于学习率难以确定，易导致收敛速度过慢或过早收敛等弊端，同时算法收敛性能也有待提高。因此本文基于多线程与 PPO 算法设计的思想，设计了基于 DPPO 的节点选择算法，如图 2(b)所示。PPO 作为一种基于 AC 框架的强化学习算法，通过采用正则项的方式限制策略更新幅度，解决了传统策略梯度更新步长难以确定的问题[24]。为进一步提高收敛速度，基于 DPPO 的节点选择算法使用多个线程在环境中收集数据，且多个线程共享一个全局 PPO 网络。



(a) 基于DRL的节点选择框架 (b) DPPO算法流程

图 2 基于 DRL 的 FL 节点选择方法

本文首先将 FL 节点选择问题表述为一个 MDP 模型，然后设计了基于 DPPO 的节点选择算法对问题进行了求解，具体设计如下。

3.2 MDP 模型

状态空间。t 时刻环境状态 s_i^t 可由一个四元组 $s_i^t=\{\Phi_i,C_i^t,H_i^{t-1},a_i^{t-1}\}$ 表示，其中，Φ_i 表示 FL 任务 i 的信息，C_i^t 表示终端设备在 t 时刻可用于 FL 任务 i 的资源，H_i^{t-1} 表示终端设备在上一时刻的数据集，a_i^{t-1} 表示上一时刻的节点选择方案。

动作空间。在进行每步动作选择时，代理只被允许采用一种节点选择方案，将 FL 任务 i 在 t 时刻的节点选择方案建模为一个 0-1 二进制向量 $\boldsymbol{a}_i^t=\{\beta_{i,1},\beta_{i,2},\beta_{i,3},\cdots,\beta_{i,d}\},\beta_{i,d}\in\{0,1\}$，其中，$\beta_{i,d}$=1 表示编号为 d 的设备在此次节点选择中被选中，反之则表示未被选中。因此，经节点选择后权值聚合表示为

$$\omega_g^i{}'=\omega_g^i+\sum_{z\in Z_i}\sum_{d\in D}\frac{\left|H_{z,d}\right|\left(\omega_{z,d}{}'-\omega_{z,d}\right)}{\left|H_i\right|}\beta_{i,d} \tag{12}$$

奖励函数。当代理根据某个节点选择策略执行某步动作后，环境信息会随之变化并得到一个用于评价本次行为的奖励值。本文考虑基于 FL 的测试准确率设计奖励函数，并设置最大时延作为每步动作选择的约束，奖励函数表示为

$$r_i^t=\frac{-1}{\sum_{d\in D}\beta_{i,d}}L^i(x_{\text{test}},y_{\text{test}};\omega_g^i) \tag{13}$$

上述执行动作来源是一个策略 π，π 是状态空间到动作空间的一个映射，即

$$a_i^t=\pi\left(s_i^t\right) \tag{14}$$

MDP 模型的目标是得到一个优化策略，即在相应的状态根据该策略采用相应动作后，使强化学习的目标−累积回报的期望最大，即求解

$$\pi^*=\arg\max_{\pi}\mathrm{E}\left[\sum_{t=0}^{\infty}\sigma^t r_i^t\right] \tag{15}$$

其中，σ^t 为折扣因子，其值随时间增加而减小。

3.3 基于 DPPO 的 FL 节点选择算法

全局 PPO 网络中包含 2 个 Actor 网络（Actor_1 和 Actor_2）以及一个 Critic 网络。Actor_1 代表当前最新的策略 π 并负责指导各线程与环境交互。Critic 网络根据代理执行节点选择动作后获得的奖励对当前策略进行评判，并通过损失函数的反向传播实现对 Critic 网络中的参数进行更新。Actor_2 代表旧策略 π_{old} 训练 circle 步后，使用 Actor_1 的参数对 Actor_2 进行更新。重复上述过程直至收敛。

相较于传统策略梯度算法，PPO 首先对算法梯度进行改进，策略梯度的原始参数更新方程为

$$\theta_{\text{new}}=\theta_{\text{old}}+\alpha\nabla_\theta J \tag{16}$$

其中，θ_{old} 和 θ_{new} 分别表示更新前后的策略参数，α 表示学习率，$\nabla_\theta J$ 表示目标函数梯度。PPO 将新策略的回报函数分解为旧策略对应的回报函数加其他项，为实现回报函数的单调不减，只需保证新策略中的其他项大于或等于 0，表示为

$$J(\tilde{\pi})=J(\pi)+\mathrm{E}_{s_0,a_0,\cdots,\tilde{\pi}}\left[\sum_{t=0}^{\infty}\sigma^t\mathrm{Adv}_{\pi}\left(s_i^t,a_i^t\right)\right] \quad (17)$$

其中，J 表示当前策略的回报函数，π 表示旧策略，$\tilde{\pi}$ 表示新策略，$\mathrm{Adv}_{\pi}(s_i^t,a_i^t)$ 表示优势函数。基于上述分析可知[25]，PPO 的优化目标是通过对参数 θ 进行更新以满足

$$\max_{\theta}\mathrm{E}\left[\frac{\pi_{\theta(a|s)}}{\pi_{\theta_{\mathrm{old}}(a|s)}}\mathrm{Adv}_{\theta_{\mathrm{old}}}\left(s_i^t,a_i^t\right)\right] \quad (18)$$

其中，$\pi_{\theta(a|s)}$ 为基于策略 π 在状态 s 下采取动作 a 的概率，且 $D_{\mathrm{KL}}^{\max}\left(\theta_{\mathrm{old}},\theta\right)\leqslant\kappa$，$D_{\mathrm{KL}}^{\max}\left(\theta_{\mathrm{old}},\theta\right)$ 表示旧策略参数与新策略参数之间相对熵的最大值，相对熵用于度量 θ_{old} 和 θ 这 2 个参数的概率分布之间的相似度，进而控制策略的更新幅度。

$$L^{\mathrm{KLPEN}}(\theta)=\mathrm{E}_t\left[\frac{\pi_{\theta(a|s)}}{\pi_{\theta_{\mathrm{old}}(a|s)}}\mathrm{Adv}_{\theta_{\mathrm{old}}}\left(s_i^t,a_i^t\right)-\lambda\mathrm{KL}\left[\pi_{\theta_{\mathrm{old}}},\tilde{\pi}_{\theta}\right]\right] \quad (19)$$

在考虑约束条件后，PPO 中基于拉格朗日乘数法的初始策略更新如上所示。为解决超参数 λ 难以确定的问题，本文考虑使用 t 时刻的新策略与旧策略的比值衡量策略的更新幅度，表示为

$$\mathrm{ratio}_t(\theta)=\frac{\pi_{\theta(a_t|s_t)}}{\pi_{\theta_{\mathrm{old}}(a_t|s_t)}} \quad (20)$$

当策略未发生变化时，$\mathrm{ratio}_t(\theta)=1$。用裁剪函数 clip 对新旧策略之间的更新幅度进行限制，改进后的策略更新方式为

$$L^{\mathrm{CLIP}}(\theta)=\mathrm{E}_t\left[\min\left(\mathrm{ratio}_t(\theta)\mathrm{Adv}_t,\mathrm{clip}\left(\mathrm{ratio}_t(\theta)\right),1-\varepsilon,1+\varepsilon\right)\mathrm{Adv}_t\right] \quad (21)$$

其中，$\varepsilon\in[0,1]$ 是一个超参数，裁剪函数将 $\mathrm{ratio}_t(\theta)$ 的值约束在区间 $[1-\varepsilon,1+\varepsilon]$ 内。

基于上述对 PPO 的分析，结合多线程的思想，提出了基于 DPPO 的 FL 节点选择算法，主要分为多线程交互和全局网络更新 2 个过程。

1）多线程交互

步骤 1　将初始状态输入 Actor_1 网络中，各线程基于策略 π_{old} 选择一个动作与环境进行交互，即 $a_i^t=\pi\left(s_i^t\right)$。

步骤 2　各线程分别与环境连续交互多次，收集包含动作、状态和奖励的样本，并将批量样本同步传输至全局 PPO 网络处。

2）全局网络更新

步骤 1　全局 PPO 网络使用式(22)计算每个时间步的优势函数，即

$$\mathrm{Adv}_t=\sum_{j>t}\sigma^{j-t}r_i^t-V_{\varphi}\left(s_i^t\right) \quad (22)$$

其中，V 为状态值函数，φ 为 Critic 网络参数。

步骤 2　利用 $L(\phi)=-\sum_{t=1}^{T}\left(\sum_{j>t}\sigma^{j-t}r_i^t-V_{\varphi}\left(s_i^t\right)\right)^2$ 计算 Critic 网络的损失函数，并反向传播更新 Critic 网络参数 φ。

步骤 3　利用 $L^{\mathrm{CLIP}}(\theta)$ 与优势函数对 Actor_1 网络的参数进行更新。

步骤 4　circle 步后使用 Actor_1 中的网络参数更新 Actor_2 的参数。

步骤 5　循环步骤 1~步骤 4，直至模型收敛。

全局网络模型收敛后，可指导代理根据不同的环境状态得出相应的动作，进而选择合理的节点集合参与 FL 聚合。详细过程如算法 1 所示。

算法 1　基于 DPPO 的节点选择算法

输入　网络的初始状态、FL 任务信息

输出　节点选择方案

1) 初始化网络、设备及任务信息，随机初始化系统状态和全局网络参数

2) for episode $\in\{1,\cdots,\mathrm{EP}\}$

3)　　for sub_episode $\in\{1,\cdots,\mathrm{EP}_s\}$

4)　　各代理根据全局 PPO 策略 $\alpha_{i,t}=\pi(s_t)$ 执行节点选择动作 $\alpha_{i,t}$

5)　　各代理根据式(13)获得奖励 r_t 和下一个状态 s_{t+1}，并将当前状态、动作和奖励存为样本

6)　　更新当前网络及设备状态信息

7)　　end for

8)　　各代理将收集的数据同步上传至全局网络处

9)　　根据式(22)优势函数和式(21)更新 Actor_1 网络参数 θ

10)　　根据 $L(\phi)$ 反向传播更新 Critic 网络参数 φ

11)　　if sub_episode%circke == 0

12)　　使用 Actor_1 中的参数对 Actor_2 进行更新；

13)　　end if

14)end for

4 仿真分析

4.1 实验设置

本文在 Python 3.8 和 TensorFlow 2.3.1 环境下对算法进行了仿真验证。实验模拟了 MEC 环境中，多类终端设备进行分布式 FL 训练的场景。场景包含一个汇聚服务器、10 个 MEC 服务器以及每个 MEC 服务器下 10~80 台的终端设备。MEC 场景中的终端设备用处理器为 AMD Ryzen 7 4800U、配置为 8 核 16 GB 的计算机来模拟。为体现终端差异化计算能力，实验中采用虚拟化 docker 技术随机分配计算机中[10%, 100%]的核数用于模型训练。

实验首先选择 MNIST 数据集作为训练数据。将数据集分割为每组 100~2 000 个，并分配给终端节点作为本地数据集。采用卷积神经网络作为 FL 的训练模型，并将模型结构设置为 2 层卷积层和 4 层全连接层。每经过 5 次本地迭代或者本地迭代时间超过最大允许本地迭代时间时，系统进行一次全局参数合成。为体现所提方法的稳健性，实验中设置了恶意节点来模拟训练质量差的设备，该类节点可能不训练模型，而是随机生成模型参数并将其上传，实验中把这个概率随机设置在 80%~100%。通过节点上独立同分布数据的比例来表征数据质量，该比例在[80%,100%]随机设置。此外，本文还选取 CIFAR 数据集，并将卷积神经网络改为 5 层卷积层和 3 层全连接层，对算法进行了验证。

DPPO 算法中使用 4 个线程与外部环境进行交互，奖励折扣系数设置为 0.9。Actor 网络和 Critic 网络的学习率分别设置为 0.000 1、0.000 2，且每当代理训练 100 个回合就使用 Actor_1 中的参数对 Actor_2 进行更新。为实现对策略更新幅度的控制，clip() 中的超参数设为 0.2。具体实验参数的设置如表 2 所示。

选取 2 个算法作为本文所提算法（FL-DPPO）的对比。1) FL-Greedy：该算法在 FL 每次迭代训练中选择全部设备节点进行模型汇聚。2) Local Training：不采用 FL 机制，仅在本地设备上进行模型训练。

4.2 结果分析

实验从准确率、损失函数、时延等多个角度对 3 种算法进行了分析。MNIST 数据集属于分类问题，因此实验中的准确率可定义为分类正确的数量占总样本数的比例。

表 2 仿真参数设置

参数类型	参数描述	设置
网络与模型参数	MEC 数	10
	终端数/MEC	80
	终端核数	[10%, 100%]
	本地数据集	[100, 2 000]
	本地迭代（MNIST/CIFAR）	5
	卷积层（MNIST/CIFAR）	2/5
	全连接层（MNIST/CIFAR）	4/3
	节点不训练概率	[80%,100%]
	独立同分布数据比例	[80%,100%]
DPPO 参数	代理数 Agents	4
	训练步数	1 000
	Actor α	0.000 1
	Critic α	0.000 2
	奖励折扣因子 σ	0.9
	限制步长 ε	0.2
	策略更新步数 circle	100
	最小样本数 Batch-size	64

图 3 给出了每个 MEC 下有 10%的恶意设备节点时 3 种算法准确率的变化情况。从图 3 中可以看出，3 种机制在训练初期得到的模型准确率较低，这说明模型的训练精度需要足够的训练次数来保证。当迭代次数达到 10 次时，3 种机制训练得到的模型准确率趋于稳定，FL-DPPO、FL-Greedy 和 Local Training 的准确率分别稳定在 0.94、0.87 和 0.7 附近。FL-DPPO 算法在应对少量恶意节点和差异化数据质量时仍能保持较好的训练性能，而 Local Training 很难保证训练质量。

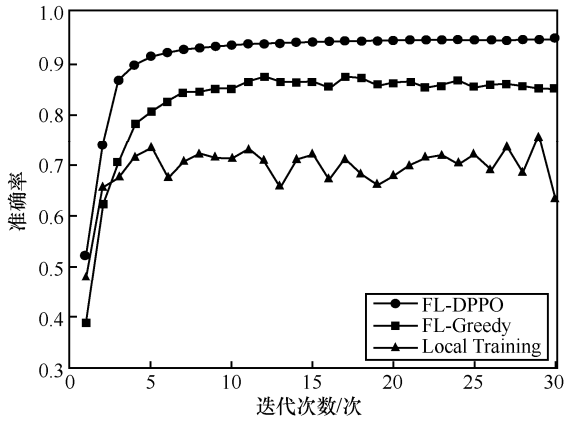


图 3 准确率对比（恶意设备节点占 10%）

图 4 是每个 MEC 下有 10%的恶意设备节点时 3 种算法损失函数的变化情况。FL-DPPO 算法相较于另外 2 种算法能更快地收敛，且损失函数值最小。Local Training 由于未采用 FL 机制，其损失函数始终无法收敛且明显高于 FL-DPPO 和 FL-Greedy。

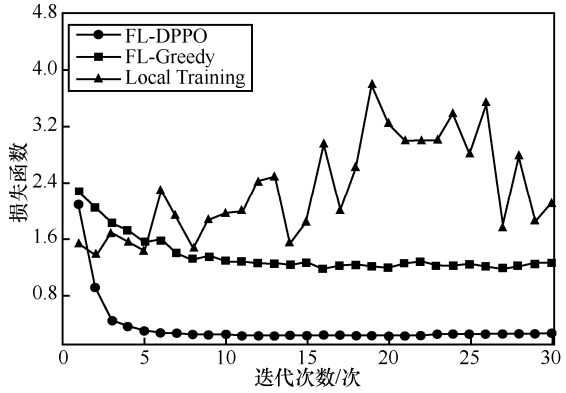


图 4　损失函数对比（恶意设备节点占 10%）

图 5 给出了每个 MEC 下有 40%的恶意设备节点时 3 种算法准确率的变化情况。从图 5 中可以看出，在应对较多恶意节点时，FL-DPPO 仍能快速收敛至最高的准确率（0.92）。FL-Greedy 受恶意节点的影响，获得的模型质量明显下降，保持在 0.71 左右，与 Local Training 的训练性能接近。本文所提 FL 机制具有兼顾数据质量和设备训练的能力，并可有效保证模型质量。

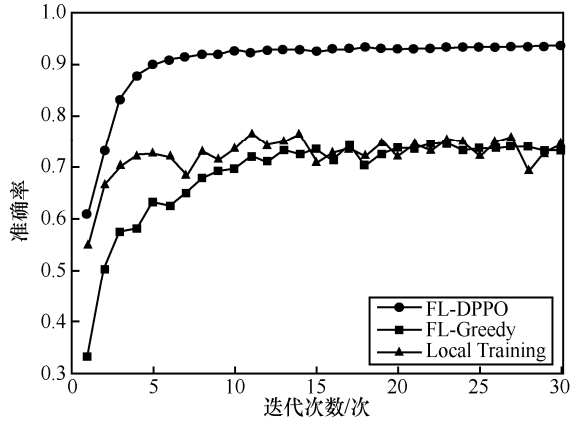


图 5　准确率对比（恶意设备节点占 40%）

图 6 是每个 MEC 下有 40%的恶意设备节点时 3 种算法损失函数的变化情况。与准确率的收敛情况类似，FL-DPPO 算法相较于另外 2 种算法能更快地收敛，且损失函数值最小。FL-Greedy 和 Local Training 由于恶意节点的存在，损失函数值始终较高。

对比上述 2 组仿真结果可以看出，相比于 FL-Greedy 和 Local Training，FL-DPPO 在面对不同数量的恶意节点时，始终能快速收敛至最高的准确率，因此可以得出本文所提方法具有良好的稳健性。

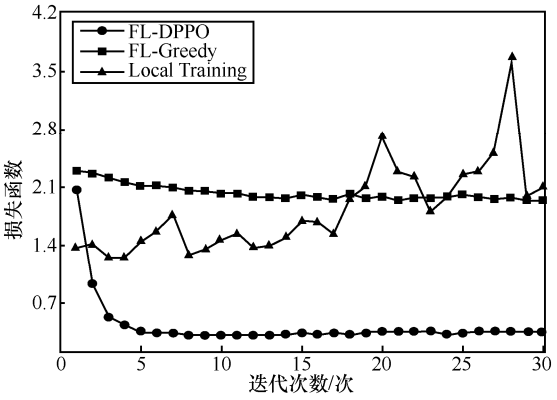


图 6　损失函数对比（恶意设备节点占 40%）

3 种算法的时延对比如图 7 所示。从图 7 中可以看出，FL-DPPO 算法在应对多种节点数目时都能保证较低的时延，这是由于该算法能有效选择训练质量高的设备节点进行模型汇聚。以节点数目 40 为例，3 种算法的时延值分别为 7.3 s、8.1 s 和 10 s，FL-DPPO 算法分别比 FL-Greedy 和 Local Training 降低了 9.9%和 27%。这说明本文所提算法能高效地完成 FL 训练。

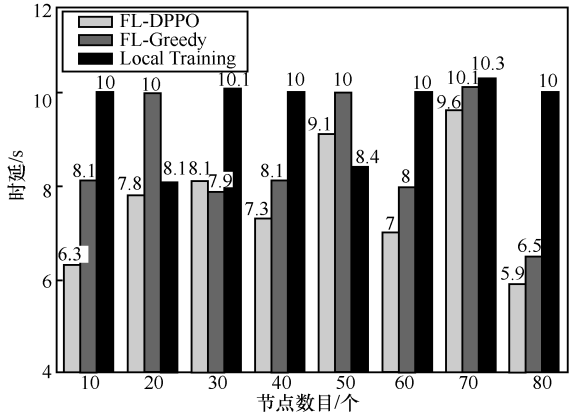


图 7　时延对比

图 8 是 3 种算法在不同的节点数目情况下获得的模型准确率。FL-DPPO 算法在应对多个节点数目时都能获得最高的准确率。以 40 个节点为例，3 种算法的准确率分别为 0.95、0.78 和 0.23，FL-DPPO 算法的准确率分别比 FL-Greedy 和 Local Training 提高了 17.9%和 75.8%。2 组数据同时说明本文所提方法在节点规模方面有着良好的扩展性能。

图 9 表示 FL-DPPO 算法的收敛特性。从图 9 中可以看出，准确率随着 DRL 训练步数的增加逐渐变大，当 Episode=40 时，算法在 150 步左右收敛得到最大准确率。当 Episode=1 时，算法也能在 500 步左右收敛。这说明 FL-DPPO 算法具有良好的收敛性能，在应对复杂的状态环境和高维的动作空间时有良好的表现。

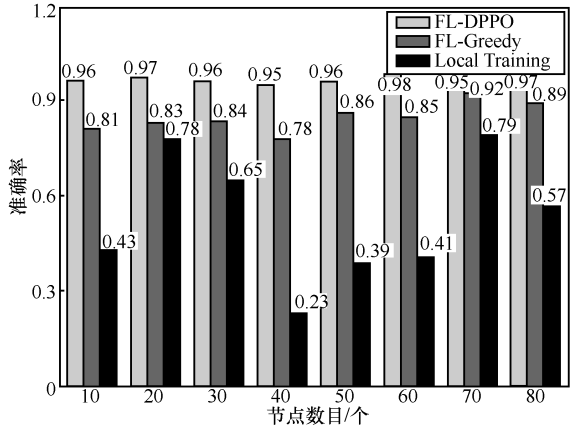


图 8　准确率对比

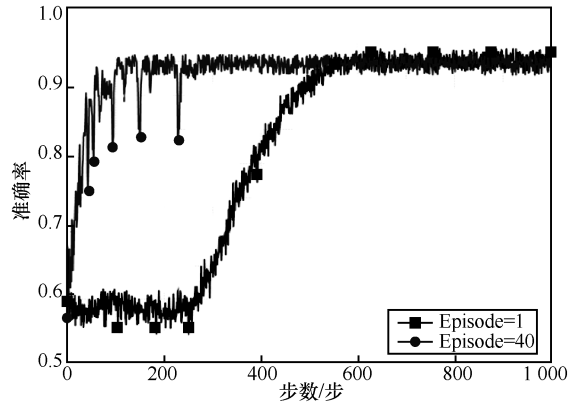


图 9　算法收敛性

接下来，采用 CIFAR 数据集对 3 种算法进行了对比和验证。图 10 给出了每个 MEC 下有 20%的恶意设备节点时 3 种算法准确率的变化情况。从图 10 中可以看出，相比于 MNIST 数据集，CIFAR 数据集的训练次数明显增多。当迭代次数达到 60 次时，3 种机制训练得到的模型准确率趋于稳定，FL-DPPO、FL-Greedy 和 Local Training 的准确率分别稳定在 0.75、0.62 及 0.55。FL-DPPO 算法在应对恶意节点和差异化数据质量时仍能保持较好的训练性能，而 Local Training 很难保证训练质量。

图 11 是每个 MEC 下有 20%的恶意设备节点时 3 种算法损失函数的变化情况。FL-DPPO 算法相较于另外 2 种算法能更快地收敛，且损失函数值最小。Local Training 由于未采用 FL 机制，其损失函数始终无法收敛且高于另外两者。

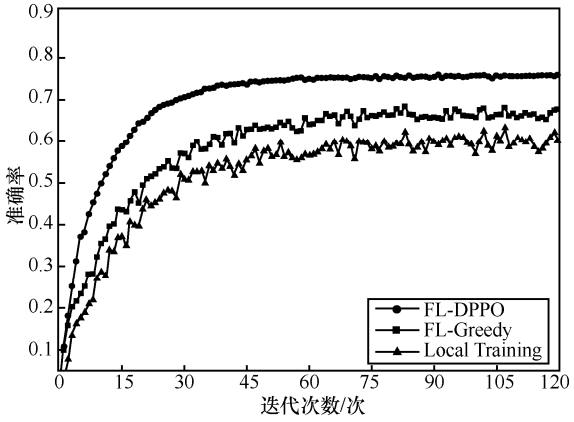


图 10　准确率对比

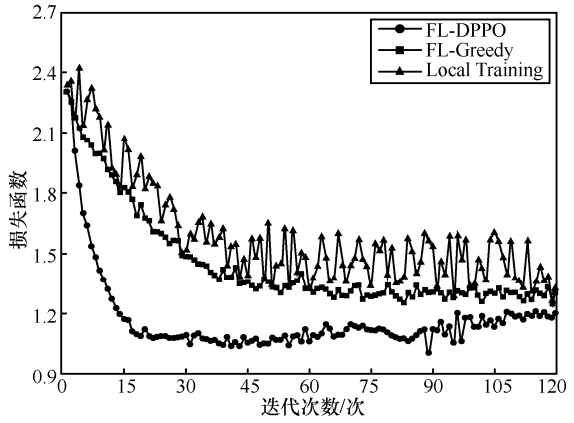


图 11　损失函数对比

5　结束语

基于深度强化学习方法，本文提出了 FL 系统中设备节点选择方法，在兼顾设备训练能力和数据质量的情况下，有效提高了 FL 学习的效率和性能。首先，根据 FL 特点，提出基于 DRL 的节点选择系统模型。其次，考虑设备训练时延、模型传输时延和准确率等因素，构建面向节点选择的准确率最优化问题模型。最后，将问题模型构建为 MDP 模型，并设计基于分布近端策略优化的节点选择算法，在每次训练迭代前选择合理的设备集合完成模型聚合。仿真实验结果表明，所提方法显著提高了 FL 的准确率和训练速度，且具有良好的收敛性和稳健性，为在网络边缘侧执行 FL 提供了一种有效的解决方案。

参考文献：

[1]　ZHOU Z, CHEN X, LI E, et al. Edge intelligence: paving the last mile

of artificial intelligence with edge computing[J]. Proceedings of the IEEE, 2019, 107(8): 1738-1762.

[2] LU Y L, HUANG X H, DAI Y Y, et al. Federated learning for data privacy preservation in vehicular cyber-physical systems[J]. IEEE Network, 2020, 34(3): 50-56.

[3] 陈兵, 成翔, 张佳乐, 等. 联邦学习安全与隐私保护综述[J]. 南京航空航天大学学报, 2020, 52(5): 675-684.
CHEN B, CHENG X, ZHANG J L, et al. Survey of security and privacy in federated learning[J]. Journal of Nanjing University of Aeronautics & Astronautics, 2020, 52(5): 675-684.

[4] WANG H, KAPLAN Z, NIU D, et al. Optimizing federated learning on non-IID data with reinforcement learning[C]//IEEE INFOCOM 2020 - IEEE Conference on Computer Communications. Piscataway: IEEE Press, 2020: 1698-1707.

[5] ABDULRAHMAN S, TOUT H, OULD-SLIMANE H, et al. A survey on federated learning: the journey from centralized to distributed on-site learning and beyond[J]. IEEE Internet of Things Journal, 2021, 8(7): 5476-5497.

[6] SHI W Q, ZHOU S, NIU Z S. Device scheduling with fast convergence for wireless federated learning[C]//2020 IEEE International Conference on Communications. Piscataway: IEEE Press, 2020: 1-6.

[7] REN J K, HE Y H, WEN D Z, et al. Scheduling for cellular federated edge learning with importance and channel awareness[J]. IEEE Transactions on Wireless Communications, 2020, 19(11): 7690-7703.

[8] CHEN M Z, POOR H V, SAAD W, et al. Convergence time minimization of federated learning over wireless networks[C]//2020 IEEE International Conference on Communications. Piscataway: IEEE Press, 2020: 1-6.

[9] WU W T, HE L G, LIN W W, et al. Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(7): 1539-1551.

[10] KANG J W, XIONG Z H, NIYATO D, et al. Incentive mechanism for reliable federated learning: a joint optimization approach to combining reputation and contract theory[J]. IEEE Internet of Things Journal, 2019, 6(6): 10700-10714.

[11] LU Y L, HUANG X H, ZHANG K, et al. Blockchain empowered asynchronous federated learning for secure data sharing in Internet of vehicles[J]. IEEE Transactions on Vehicular Technology, 2020, 69(4): 4298-4311.

[12] YOSHIDA N, NISHIO T, MORIKURA M, et al. Hybrid-FL for wireless networks: cooperative learning mechanism using non-IID data[C]//2020 IEEE International Conference on Communications. Piscataway: IEEE Press, 2020: 1-7.

[13] YANG Z H, CHEN M Z, SAAD W, et al. Energy efficient federated learning over wireless communication networks[J]. IEEE Transactions on Wireless Communications, 2021, 20(3): 1935-1949.

[14] ZENG T C, SEMIARI O, MOZAFFARI M, et al. Federated learning in the sky: joint power allocation and scheduling with UAV swarms[C]//2020 IEEE International Conference on Communications. Piscataway: IEEE Press, 2020: 1-6.

[15] TRAN H V, KADDOUM G, ELGALA H, et al. Lightwave power transfer for federated learning-based wireless networks[J]. IEEE Communications Letters, 2020, 24(7): 1472-1476.

[16] LUO S Q, CHEN X, WU Q, et al. HFEL: joint edge association and resource allocation for cost-efficient hierarchical federated edge learning[J]. IEEE Transactions on Wireless Communications, 2020, 19(10): 6535-6548.

[17] 孟洛明, 孙康, 韦磊, 等. 一种面向电力无线专网的虚拟资源优化分配机制[J]. 电子与信息学报, 2017, 39(7): 1711-1718.
MENG L M, SUN K, WEI L, et al. Optimal resource allocation mechanism for electric power wireless virtual networks[J]. Journal of Electronics & Information Technology, 2017, 39(7): 1711-1718.

[18] 李枝灵, 刘柱, 郭少勇, 等. 基于免疫算法的电力线通信网接入点规划方法[J]. 北京邮电大学学报, 2016, 39(S1): 104-108.
LI Z L, LIU Z, GUO S Y, et al. Access points location planning based on immune algorithm for power line communication network[J]. Journal of Beijing University of Posts and Telecommunications, 2016, 39(S1): 104-108.

[19] 赵海涛, 张唐伟, 陈跃, 等. 基于DQN的车载边缘网络任务分发卸载算法[J]. 通信学报, 2020, 41(10): 172-178.
ZHAO H T, ZHANG T W, CHEN Y, et al. Task distribution offloading algorithm of vehicle edge network based on DQN[J]. Journal on Communications, 2020, 41(10): 172-178.

[20] 喻鹏, 张俊也, 李文璟, 等. 移动边缘网络中基于双深度 Q 学习的高能效资源分配方法[J]. 通信学报, 2020, 41(12): 148-161.
YU P, ZHANG J Y, LI W J, et al. Energy-efficient resource allocation method in mobile edge network based on double deep Q-learning[J]. Journal on Communications, 2020, 41(12): 148-161.

[21] PAN S L, ZHANG Z Y, ZHANG Z W, et al. Dependency-aware computation offloading in mobile edge computing: a reinforcement learning approach[J]. IEEE Access, 2019, 7: 134742-134753.

[22] YANG Z Y, MERRICK K, JIN L W, et al. Hierarchical deep reinforcement learning for continuous action control[J]. IEEE Transactions on Neural Networks and Learning Systems, 2018, 29(11): 5174-5184.

[23] LIANG X Y, DU X S, WANG G L, et al. A deep reinforcement learning network for traffic light cycle control[J]. IEEE Transactions on Vehicular Technology, 2019, 68(2): 1243-1253.

[24] LIU C F, BENNIS M, DEBBAH M, et al. Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing[J]. IEEE Transactions on Communications, 2019, 67(6): 4132-4150.

[25] SCHULMAN J, LEVINE S, MORITZ P, et al. Trust region policy optimization[J]. arXiv Preprint, arXiv:1502.05477, 2015.

[作者简介]

贺文晨（1993− ），男，山东济南人，北京邮电大学博士生，主要研究方向为边缘智能、任务部署和资源分配。

郭少勇（1985− ），男，河北邢台人，博士，北京邮电大学副教授，主要研究方向为物联网与区块链。

邱雪松（1973− ），男，江西上饶人，博士，北京邮电大学教授、博士生导师，主要研究方向为网络与业务管理、物联网与区块链。

陈连栋（1987− ），男，山东莱州人，国网河北信息通信分公司副高级工程师，主要研究方向为网络信息安全和数据加密。

张素香（1973− ），女，河北衡水人，博士，国家电网有限公司信息通信分公司高级工程师，主要研究方向为电力系统通信、物联网和人工智能等。