Journal of System Simulation

Volume 36 | Issue 9 Article 12

9-15-2024

Edge Surveillance Task Offloading and Resource Allocation Algorithm Based on DRL

Chao Li
School of Computer Science, Hubei University of Technology, Wuhan 430000, China

Jiabao Li
School of Computer Science, Hubei University of Technology, Wuhan 430000, China

Caichang Ding
School of Computer and Information Science, Hubei Engineering University, Xiaogan 432000, China

Zhiwei Ye
School of Computer Science, Hubei University of Technology, Wuhan 430000, China

See next page for additional authors

Edge Surveillance Task Offloading and Resource Allocation Algorithm Based on DRL

Abstract

Abstract: For the resource limitation of intensive surveillance tasks in edge computing, a surveillance task offloading and resource allocation algorithm based on DRL is proposed. With the optimization objectives of surveillance task delay and recognition accuracy, the joint decision objective optimization solution of task offloading, wireless channel allocation, and image compression rate was modeled as a Markov decision process. To address the problem of slow and unstable algorithm convergence due to the high volatility of training samples caused by the dynamic nature of wireless channels and the randomness of surveillance tasks, an attention mechanism is used to jointly encode channel states and surveillance task information from multi-slot state sequences. By capturing the dependency relationships between multi-slot state sequences, the representation ability of network state and the robustness of the algorithm are improved. Experimental results show that the proposed algorithm outperforms traditional reinforcement learning algorithm and heuristic algorithm in improving recognition accuracy and reducing task computation delay.

Keywords

surveillance task, mobile edge computing, DRL, task offloading, resource allocation, attention mechanism

Authors

Chao Li, Jiabao Li, Caichang Ding, Zhiwei Ye, and Fangwei Zuo

Recommended Citation

Li Chao, Li Jiabao, Ding Caichang, et al. Edge Surveillance Task Offloading and Resource Allocation Algorithm Based on DRL[J]. Journal of System Simulation, 2024, 36(9): 2113-2126.

第 36 卷第 9 期 系统仿真学报© Vol. 36 No. 9
2024 年 9 月 Journal of System Simulation Sept. 2024

基于DRL的边缘监控任务卸载与资源分配算法

李超[1]，李贾宝[1]，丁才昌[2*]，叶志伟[1]，左方威[1]
(1. 湖北工业大学 计算机学院，湖北 武汉 430000；2. 湖北工程学院 计算机与信息科学学院，湖北 孝感 432000)

摘要：为解决边缘计算环境下密集型监控任务资源受限的问题，提出一种基于DRL的监控任务卸载与资源分配算法。以监控任务时延和识别精度为优化目标，将监控系统中的任务卸载、无线信道分配和图像压缩率的联合决策目标优化求解建模为马尔可夫决策过程；针对无线信道动态性和监控任务随机性引起的训练样本波动性较大，导致算法收敛速度慢和不稳定，采用Transformer注意力机制对多时隙序列的信道状态和监控任务信息进行联合编码。编码后的状态信息能够捕捉多时隙状态序列之间的依赖关系，提升网络状态的表征能力，并以此提高算法鲁棒性。实验结果表明：与传统强化学习算法和启发式算法相比，该算法在降低任务计算时延的同时能够有效提高识别精度。

关键词：监控任务；移动边缘计算；深度强化学习；任务卸载；资源分配；注意力机制
中图分类号：TP391.9 文献标志码：A 文章编号：1004-731X(2024)09-2113-14
DOI: 10.16182/j.issn1004731x.joss.23-0576
引用格式: 李超, 李贾宝, 丁才昌, 等. 基于DRL的边缘监控任务卸载与资源分配算法[J]. 系统仿真学报, 2024, 36(9): 2113-2126.
Reference format: Li Chao, Li Jiabao, Ding Caichang, et al. Edge Surveillance Task Offloading and Resource Allocation Algorithm Based on DRL[J]. Journal of System Simulation, 2024, 36(9): 2113-2126.

Edge Surveillance Task Offloading and Resource Allocation Algorithm Based on DRL

Li Chao[1], *Li Jiabao*[1], *Ding Caichang*[2*], *Ye Zhiwei*[1], *Zuo Fangwei*[1]
(1. School of Computer Science, Hubei University of Technology, Wuhan 430000, China;
2. School of Computer and Information Science, Hubei Engineering University, Xiaogan 432000, China)

Abstract: For the resource limitation of intensive surveillance tasks in edge computing, a surveillance task offloading and resource allocation algorithm based on DRL is proposed. *With the optimization objectives of surveillance task delay and recognition accuracy, the joint decision objective optimization solution of task offloading, wireless channel allocation, and image compression rate was modeled as a Markov decision process.* To address the problem of slow and unstable algorithm convergence due to the high volatility of training samples caused by the dynamic nature of wireless channels and the randomness of surveillance tasks, *an attention mechanism is used to jointly encode channel states and surveillance task information from multi-slot state sequences.* By capturing the dependency relationships between multi-slot state sequences, the representation ability of network state and the robustness of the algorithm are improved. Experimental results show that the proposed algorithm outperforms traditional reinforcement learning algorithm and heuristic algorithm in improving recognition accuracy and reducing task computation delay.

收稿日期：2023-05-16 修回日期：2023-07-16
基金项目：国家自然科学基金(61902116)；湖北省教育厅科学技术研究计划中青年人才(Q20202705)；湖北省大学生创新训练计划(S202210500096)
第一作者：李超(1982-)，男，讲师，博士，研究方向为深度学习与边缘计算。
通讯作者：丁才昌(1980-)，男，副教授，博士，研究方向为智能信息处理。

Keywords: surveillance task; mobile edge computing; DRL; task offloading; resource allocation; attention mechanism

0 引言

随着监控设备传感器性能的提升，监控系统被广泛应用于无人机、海上交通检测、智能汽车等[1-5]众多领域。与此同时，新的计算模式移动边缘计算(mobile edge computing, MEC)逐渐兴起，为计算密集和时延敏感的监控系统的进一步发展提供了关键技术[6-8]。然而，由于接入边缘监控网络的设备数量急剧增长，计算任务数量也随之增长，不仅会导致边缘服务器过载，还会耗费大量的带宽资源。因此，如何在有限的通信资源和计算资源下满足监控系统的时延敏感性是需要解决的关键问题。

在边缘监控系统中，监控设备将时延敏感的计算任务卸载到资源有限的边缘服务器进行计算，所以，任务卸载和资源分配是保证边缘计算场景下监控系统低时延的重点[9]。学者们提出了许多边缘计算场景下计算任务卸载和资源分配问题的算法，其中，经典的方法主要包括基于启发式的算法和基于深度强化学习的算法。

启发式算法是基于具体分析或计算经验等方面的启示，对优化问题的实例能够较快地给出问题的可行解。常见的基于启发式的算法包括贪婪算法、遗传算法、粒子群算法等[10-12]。Wang 等[13]针对车辆边缘计算场景中资源有限的问题提出了一种在线启发式算法，根据任务需求做出实时卸载决策以充分利用系统资源效率。Guo 等[14]研究了工业物联网中，移动设备电池容量有限导致任务卸载不可靠的问题，将问题表述为混合整数非线性规划问题，并提出了一种基于贪婪策略的启发式算法，以最小化系统开销为目标进行任务卸载决策和资源分配决策。Xu 等[15]针对传统启发式算法收敛慢等问题，将遗传算法与蚁群算法相结合，提出了一种基因蚁群融合算法的卸载策略，提升了算法的收敛速度。启发式算法在处理任务卸载和资源分配问题上能够高效地求解问题且易于实现，但是其建模过程复杂，无法有效地使用先验知识和全局信息，在求解过程中容易陷入局部最优点从而无法求得全局最优解[16]。

DRL 通过与环境的交互学习来处理决策问题，利用 DNN 处理用户信息和数据度量，并且能够基于信道环境和边缘节点状态学习长期最优的任务卸载和资源分配策略。Yan 等[17]提出一种基于强化学习算法的视频监控人脸识别框架，将本地无法执行的计算任务卸载到边缘执行，以低系统时延为目标，学习最优的计算任务卸载和资源分配方案。Zhou 等[18]提出了一种基于 DRL 的方法，以最小化 MEC 系统的能量消耗为目标，解决了动态的多用户 MEC 系统中计算卸载和资源分配的联合优化问题。Chen 等[19]研究了一种基于 DRL 的动态资源管理算法，解决了信息物联网中由于边缘节点的负载动态不确定带来的任务卸载问题和任务生成的动态性带来的有限资源管理问题。Tang 等[20]针对高负载的边缘节点可能会导致计算任务延迟较高的问题，提出了一种基于无模型的 DQN 深度强化学习分布式算法，以最小化系统长期成本进行任务的卸载决策，以保证系统计算任务的低时延。

相比启发式算法，DRL 算法不依赖于先验知识，能够通过与环境交互学习到长期的最优解，而且 DRL 算法可以根据实际需求进行自适应调整，启发式算法需要在先验知识的基础上构建模型，在特定场景下的适应性较差。但是，上述研究在利用 DRL 解决任务卸载和资源分配等问题时，只考虑了当前系统状态，忽略了多个系统状态关系对于算法性能的影响，而且由于强化学习中环境状态的动态性和计算任务的随机性会导致模型收敛速度较慢和稳定性差的问题。因此，本

DOI: 10.16182/j.issn1004731x.joss.23-0576

文构建了一个边缘监控人脸识别系统仿真模型，提出一个DRL的边缘监控任务卸载和资源分配算法SAC-MSE (multi-state-encoder)。

1 系统模型

1.1 网络模型

边缘监控人脸识别仿真系统由一个后端MEC服务器、多个继电器和多个连接到不同继电器的前端摄像头组成，其结构如图1所示。继电器的集合表示为$\forall k \in \{1,2,\cdots,K\}$，连接到继电器$k$的摄像头的集合表示为$\forall c \in \{1,2,\cdots,C_k\}$。系统时间被分为$i$个等长的时间间隙表示为$\forall t \in \{t_1,t_2,\cdots,t_i\}$，将时隙作为系统时间单位，决策模块会在一个时隙内对任务计算卸载、无线信道分配和图像压缩比选择进行联合决策。前端摄像头传感器具有计算能力，采用轻量级的人脸识别算法执行识别任务, MEC服务器的计算能力强大，采用高复杂度高精度的CNN算法。

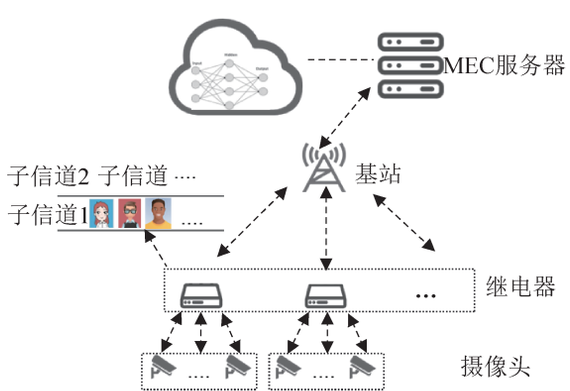

图1 基于MEC的监控识别系统
Fig. 1 MEC based surveillance and recognition system

将系统前端从视频流捕获图像的时间和前端算法识别的时间称为前端处理图像时间，并将其设定为一个常量t_{front}，将后端CNN识别算法识别图像的时间设称为后端处理图像时间，定为一个常量t_{back}。由于带宽资源的限制，如果将所有图像都传输到后端MEC服务器处理会导致边缘服务器超负载，从而造成过高的时延，不能满足系统对于实时性的要求，因此，系统处理待识别图像时共分为2个步骤：

(1) 图像如果通过前端识别算法也能得到可信的识别结果，则直接在前端输出，从而节省带宽资源。图像由前端算法处理的识别结果的正确性的置信度水平表示为$\beta \in (0,1]$，代表图像由前端直接识别成功的可信程度。当置信度大于设定阈值时认为前端的识别结果可信直接输出结果，反之则认为前端识别结果不可信，将待识别图像卸载到后端MEC服务器进行识别。

(2) 当前端识别结果不可信时，系统将图像通过无线信道压缩传输到MEC服务器进行处理以节省带宽资源，系统拟采用CRF(constant rate factor)码率控制对图像进行压缩，CRF压缩率越低，图像质量越高，压缩后数据量越大。前端的每个摄像头中都有一个待识别图像的缓冲队列，这些图像是摄像头从监控视频流中捕捉到的待识别图像任务。考虑到信道资源的有限性，当多个无线传输信道同时被占用时，待传输的图像在队列中进行等待，在下一个时隙做出决策，直到传输信道可用为止。

系统的决策模块会将所有摄像头的图像信息作为输入，同时考虑通信信道条件，对识别任务卸载、无线信道分配和压缩率选择进行决策。若图像需要传输到后端MEC服务器，将图像进行压缩传输，后端MEC服务器通过无线信道从前端获取到压缩后的图像后，对其进行解压缩，通过CNN识别算法生成最终的识别结果。将在时隙t中连接在继电器k的摄像头c的缓冲队列大小表示为$Q_{k,c,t}$,前端摄像头缓冲队列遵循先来先服务(first come first service, FCFS)原则，t时隙中连接在继电器k的摄像头c的缓冲队列的队头图像的信息表示为一个三元组$q_{k,c,t}=\{s_{k,c,t},\beta_{k,c,t},t_{k,c,t,\text{w}}\}$，其中，$s_{k,c,t}$为队头图像的数据大小，$\beta_{k,c,t}$为队头图像的前端处理置信度，$t_{k,c,t,\text{w}}$为队头图像在队列中的等待时间。

1.2 传输模型

当图像在前端识别的置信度较低时，需要通过无线信道压缩传输到后端MEC服务器进行识别。将无线传输信道的总带宽表示为B，系统的无线传输信道被划分为若干个相等且带宽大小为B_{sub}的子信道供不同的图像识别任务进行传输。传输模型中假设分配给继电器的子信道数量与继电器连接的数量成正比，将分配给继电器k的子信道集合表示为$\forall n \in \{n_1, n_2, \cdots, n_{N_{\text{sub}}}\}$。将$t$时隙子信道$n$的无线信道增益表示为$h_{n,t}$，考虑到无线信道传输会伴随着信号衰落，系统采用了Rayleigh衰落信道。由此，可将时隙t下子信道n的传输速率表示为

$$r_{n,t} = B_{\text{sub}} \text{lb}\left(1 + \frac{Ph_{n,t}}{p_{\text{N}}}\right) \tag{1}$$

式中：P为继电器的固定传输功率；p_{N}为背景噪声功率。

1.3 计算模型

图像识别任务在每个时隙中都要决定其卸载策略：前端直接输出识别结果、缓冲队列中等待、传输至MEC服务器进行识别。将k继电器下的摄像头c队列队头图像在时隙t的卸载策略表示为

$$d_{k,c,t} = \begin{cases} -1, \text{前端输出识别结果} \\ 0, \text{缓冲队列等待} \\ 1, \text{后端 MEC 服务器识别} \end{cases} \tag{2}$$

前端直接输出识别结果所耗费的总时间$t_{k,c,t,\text{f}}$由前端处理图像时间t_{front}和图像在摄像头缓冲队列中的等待时间$t_{k,c,t,\text{w}}$组成，将其表示为

$$t_{k,c,t,\text{f}} = t_{\text{front}} + t_{k,c,t,\text{w}} \tag{3}$$

由于子信道的数量有限，多个摄像头的队列竞争传输信道资源时，可能会出现传输信道被占用的情况。用$x_{n,t} \in \{0, 1\}$来表示t时隙下子信道n的占用情况，$x_{n,t} = 0$表示信道未被占用，$x_{n,t} = 1$表示信道被占用。将后端MEC服务器使用CNN算法图像识别时间表示为一个常量t_{back}，将$u_{k,c,n,t} \in \{0, 1\}$表示为t时隙继电器k下的摄像头c的队头图像是否占用子信道n来进行传输。系统规定每个时隙中一个信道只能用于一个图像识别任务传输，将其表示为

$$\sum_{c=1}^{C_K} u_{k,c,n,t} \leqslant x_{n,t} \tag{4}$$

将图像的压缩时间和解压缩时间设定为一个常量t_{cpic}。在图像进行压缩时，要选取不同的压缩率来进行压缩，将t时隙继电器k下摄像头c图像压缩率决策表示为$\forall m_{k,c,t} \in \{1, 2, \cdots N_m\}$。由此，将待处理图像在所分配的子信道上的传输时间表示为

$$T_{k,c,t,\text{r}} = \frac{g(s_{k,c,t}, m_{k,c,t})}{B_{\text{sub}} \text{lb}\left(1 + \dfrac{Ph_{n,t}}{p_{\text{N}}}\right)} = \frac{g(s_{k,c,t}, m_{k,c,t})}{r_{n,t}} \tag{5}$$

式中：$g(s, m)$为以m为压缩率，压缩原大小为s的图像得到的压缩图像大小。由于后端回传的结果数据量远小于前端传输数据量，所以将其忽略不计。后端处理图像耗费的总时间为$t_{k,c,t,\text{tb}}$，由t_{front}、$t_{k,c,t,\text{w}}$、$t_{k,c,t,\text{tr}}$、t_{cpic}和t_{back}组成，将其表示为

$$t_{k,c,t,\text{tb}} = t_{\text{front}} + t_{k,c,t,\text{w}} + t_{k,c,t,\text{tr}} + t_{\text{cpic}} + t_{\text{back}} \tag{6}$$

将系统的图像处理时间统一表示为

$$t_{k,c,t,\text{process}} = t_{\text{front}} + t_{k,c,t,\text{w}} + \epsilon(t_{k,c,t,\text{tr}} + t_{\text{cpic}} + t_{\text{back}}) \tag{7}$$

式中：$\epsilon \in \{0, 1\}$为是否将图像识别任务卸载到MEC服务器进行处理，ϵ取0时代表直接由前端输出。

1.4 缓冲队列模型

如果传输子信道处于可用的状态，每一个时隙中前端摄像头的缓冲队列都会有队头图像离开缓冲队列，也会有新的待处理图像加入队列。在时隙t下，$t+1$时隙的队列长度为

$$\varpi_{k,c,t+1} = \varpi_{k,c,t} + \tau_{k,c,t} + \kappa_{k,c,t} \tag{8}$$

式中：入队标志$\tau_{k,c,t} \in \{0, 1\}$，代表有无新图像任务进入缓冲队列，考虑到监控系统中任务到达的随机性，本文将仿真环境中的图像根据随机概率入队。离队标志$\kappa_{k,c,t} \in \{-1, 0\}$，代表有或无待处理图像离开缓冲队列，当$\kappa_{k,c,t}$取值为0时，代表着没有

DOI: 10.16182/j.issn1004731x.joss.23-0576

图像离开队列，意味着待处理图像在当前时隙处于等待状态。

2 问题表述

2.1 状态、动作和奖励函数

由于时变的无线信道条件无法预测和识别任务到达随机，将该问题表述为马尔可夫决策过程。在每个时隙的开始，前端设备观察其状态(图像任务大小、队列信息、信道增益等)，然后进行任务卸载和信道分配等决策。根据决策动作，算法得到相应奖励，如果任务在前端处理，则代表着低延迟和高奖励，如果卸载到后端MEC服务器，则代表着高精度和相对较高时延。算法的目标是通过平衡低时延和高识别精度，从状态到行动的策略映射来最大化其预期的长期奖励。马尔可夫决策过程中的关键要素即状态、动作、奖励。

(1) 状态：状态是一个反映系统整体网络环境的空间，边缘监控系统的状态由连接到继电器的摄像头缓冲队列图像信息、子信道的分配信息、子信道的信道增益等信息组成。系统状态为$S_{k,t}=(q_{k,c,t},h_{N_k,t},u_{N_k,t})$，其中，$q_{k,c,t}=\{q_{k,1,t},q_{k,2,t},\cdots,q_{k,C_k,t}\}$为图像的信息集合即图像大小、图像前端识别置信度、图像等待时间等；$h_{N_k,t}=\{h_{1,t},h_{2,t},\cdots,h_{N,t}\}$为分配给继电器$k$的所有子信道的信道增益集合；$N_k$为分配给继电器$k$的所有子信道个数；$u_{N_k,t}=\{u_{1,t},u_{2,t},\cdots,u_{N,t}\}$为分配给继电器$k$的所有子信道占用传输情况集合，系统规定算法在一个时隙开始的时候获取系统状态信息。

(2) 动作：MDP(Markov decision process)中由智能体来根据系统状态反馈出相应的动作，其实质是将状态空间映射到动作空间。在本文系统中，智能体的动作有3个部分：图像卸载决策、资源分配决策和压缩率选择决策。$A_{k,t}=(a_{k.1,t},a_{k.2,t},\cdots,a_{k,c,t},a_{k,C_k,t})$表示动作，其中，$a_{k,c,t}=(d_{k,c,t},o_{k,c,t},m_{k,c,t})$为队头图像的动作。$d_{k,c,t}\in\{-1,0,1\}$为队头图像的卸载决策。$o_{k,c,t}=\{o_{k,c,1,t},o_{k,c,2,t},\cdots,o_{k,c,n,t},o_{k,c,N_k,t}\}$为子信道分配决策，其中，$o_{k,c,n,t}\in\{0,1\}$表示将子信道$n$是否分配给摄像头$c$传输。$m_{k,c,t}\in\{1,2,\cdots,N_m\}$表示队头图像的传输压缩率决策，当识别任务直接由前端处理时图像无需压缩，所以$m_{k,c,t}$无效。

(3) 奖励函数：一个合理的奖励函数能够提升算法效率，加快算法收敛速度，本文针对不同的卸载策略制定了不同的奖励函数[21]。

$$R_{k,c,t}=\begin{cases}J/\mathrm{e}^{\mu t_{k,c,t,\mathrm{f}}},d_{k,c,t}=-1\\ J/\mathrm{e}^{\mu t_{k,c,t,\mathrm{w}}},d_{k,c,t}=0\\ J/\mathrm{e}^{\mu t_{k,c,t,\mathrm{b}}}+\rho,d_{k,c,t}=1\end{cases}\tag{9}$$

式中：$d_{k,c,t}=-1$时图像前端直接输出获取奖励；$d_{k,c,t}=0$时图像停留缓冲队列等待获取奖励，由于图像不在前端直接输出，所以做出等待决策的图像一定是倾向于传输到后端MEC进行处理；$d_{k,c,t}=1$时图像传输到后端MEC服务器获取奖励。使用参数μ平衡处理图像时间和传输图像时间，为了调整算法决策的趋势，在奖励函数中引入了超参数ρ。如果倾向于将任务卸载到后端处理，则ρ值更大，卸载到后端的动作得到更高奖励。ρ的设定对于算法至关重要，不合理的ρ值会严重影响任务卸载效率和资源利用率。即时的奖励函数与优化的目标有密切关系，为了平衡识别精度和系统时延对算法决策的影响，本文在奖励函数中使用$J=\{0,1\}$表示图像是否识别成功，成功取值1，否则取值0。图像在前端直接输出结果时，J的取值取决于前端摄像头，若卸载到后端，则J的取值取决于后端MEC服务器。

2.2 优化目标

SAC-MSE算法目标是为了让算法获得更高的奖励以降低系统时延提升识别精度，本文通过式(9)来反映优化目标。当不同决策延迟越高时奖励函数分母值越大，奖励函数值越小，因此，算法通过学习更低延迟的决策，获取更高的奖励值。

奖励函数中J是图像识别成功的标志，若识别失败则为0，无法获取到奖励值，因此算法会学习识别成功率高的决策，提升系统识别精度。将连接到继电器k的所有摄像头在时隙t获得的奖励总和表示为

$$R_{k,t,\text{tol}}=\sum_{c=1}^{C_k}R_{k,c,t} \tag{10}$$

为了求解MDP的最优解，用π^*表示在当前给定状态S下选择的最优动作，通过学习最大化长期奖励的动作π^*来达到优化目的。根据Bellman公式，将继电器k在时隙t中做出的最优动作表示为

$$\pi^*=\underset{A_{k,t}}{\text{argmax}}\left(R_{k,t,\text{tol}}+\gamma\sum_{S_{k,t+1}}P_{t,t+1,\text{trans}}V(S_{k,t+1})\right) \tag{11}$$

式中：$R_{k,t,\text{tol}}$为在$S_{k,t}$的状态下做出动作$A_{k,t}$获得的奖励值；γ为折扣回报率，表示未来预期奖励与当前奖励的折扣关系。$P_{t,t+1,\text{trans}}$为状态转移概率，表示从时隙t的状态转移到$t+1$时隙状态的概率；$V(S_{k,t+1})$为状态价值函数，表示状态$S_{k,t+1}$的长期价值。

$$V(S_{k,t+1})=E\left[\sum_{k=1}^{\infty}\gamma_k R_{k,t+1+k}\middle|S_{k,t+1}=S\right] \tag{12}$$

监控识别系统中，由于视频流中的图像捕捉不规律导致图像识别任务的随机性比较大，不同动作对环境状态的长期影响使状态转移概率无法预测，加之时变的无线信道和识别任务到达的随机性使奖励也难以预测，传统的MDP解决方法并不适用。因此，本文使用基于DRL的算法来选择最优动作，在部分网络信息已知的情况下，通过平衡单次奖励和长期奖励找到最优解。

3 算法设计

本文提出了一种带有多状态编码器的深度强化学习算法SAC-MSE，其结构如图2所示，第一部分是带有MSE的SAC网络，通过将多个历史网络状态表示为信息标志，利用MSE对信息标志提取深层次的语义信息作为SAC网络的编码状态输入，通过与环境的交互来学习任务卸载和信道分配策略。第二部分是F-Net网络，采用监督学习的方式图像任务卸载到后端MEC服务器时的压缩率决策。

3.1 多状态编码器 Multi-State-Encoder

边缘监控系统中，由于无线信道的动态性和监控任务的随机性会引起训练样本波动性较大，进而导致算法收敛速度慢和不稳定的问题。受文献[22]的启发，在时变的无线信道中，Transformer能够将强化学习中多个历史状态之间的变化关系进行建模，而且多头自注意力层对随机的系统状态更具鲁棒性。

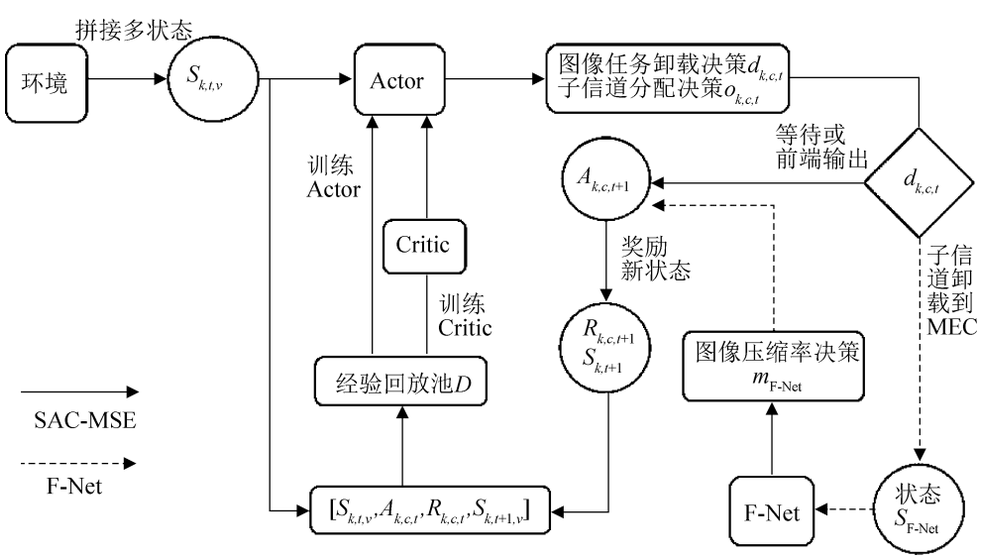

图2 SAC-MSE算法流程
Fig. 2 SAC-MSE algorithm flow ohart

DOI: 10.16182/j.issn1004731x.joss.23-0576

本文设计了一个MSE多状态编码器来学习多个状态之间的深层次的语义信息，MSE的结构如图3所示。

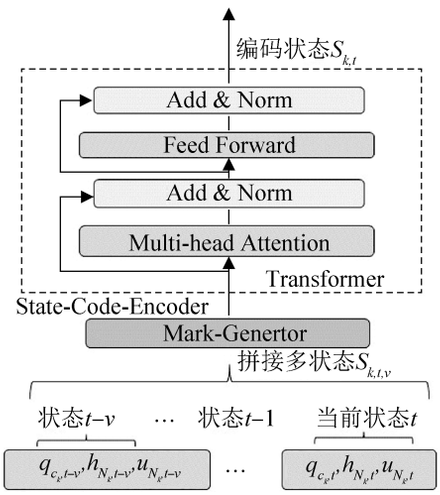

图3　Multi-State-Encoder结构
Fig. 3　Multi-State-Encoder structures

与传统的使用多层感知器(MLP)[23]或递归神经网络(RNN)[24]作为网络架构的DRL算法不同，MSE由Mark-Generator和State-Code-Encoder两部分组成。对t时刻的多状态进行编码时，首先将当前状态和输入的v个历史状态拼接为多状态$S_{k,t,v}$，然后，Mark-Generator使用层归一化Layer Normalization将$S_{k,t,v}$转化成紧凑的语义标志并对这些语义标志之间进行建模，为了实现多头自注意力，受Vision Transformer启发，将语义标志作为State-Code-Encoder的输入得到编码状态$S_{k,t}$。本文使用了一个标准的Transformer Encoder[25]作为State-Code-Encoder，每一层都有一个标准的架构，由一个标准化的多头自注意模块和一个标准化的前馈网络组成，与传统的MLP和RNN相比优点如下。

(1) 更好的表征能力：能够学习到时序的多个历史状态之间深层次的语义信息，从而更好地建模状态之间的长期依赖关系，得到更优的状态表示，提升算法的性能与收敛速度。

(2) 更高的鲁棒性：能够自适应地学习到不同的状态关系，对随机的系统状态具有更高的鲁棒性。

(3) 更好的通用性：能够将混合状态空间中多个网络状态编码成唯一的特征空间，其可以用作其他的通信系统中，作为一个通用的特征提取框架。

3.2 SAC

演员评论家(Actor-Critic)算法是价值学习和策略学习的结合，分别通过用2个神经网络来近似状态价值函数中的2个部分，用Policy Network近似策略函数，Value Network近似动作价值函数。当前流行的Actor-Critic算法包括优势演员评论家算法(advantage actor-aritic, A2C)、深度确定性策略梯度算法(deep deterministic policy gradient, DDPG)等。

SAC算法是Actor-Critic的改进算法，目标是找到一个策略函数π来最大化长期预期奖励，为了防止算法过早收敛，SAC算法在奖励中加入了熵项$H(\pi(a|s))=-\ln\pi(a|s)$[26]。通过最大化熵项来增加动作采样的随机性。$\pi(a|s)$表示在状态s下策略函数π选择动作a的概率值。在给定动作a和状态s的情况下，本文将预期奖励Soft Q Value函数定义为

$$Q^{\pi}(s,a)=E^{\pi}\left\{\sum_{t=0}^{\infty}\gamma^{k,t}\left[R_{k,t}-\alpha\ln\pi(A_{k,t}\left|S_{k,t}\right.)\right]\left|S_{k,0}=s,A_{k,0}=a\right.\right\} \tag{13}$$

式中：$-\alpha\ln\pi(A_{k,t}\left|S_{k,t}\right.)$为动作熵；$\alpha$为熵的权重，其作用是控制动作采样的随机性；$\gamma$为折扣因子，表示长期奖励的折扣。算法的学习过程是以最大化$Q_{\omega}^{\pi_{\phi}}(S,A)$为目标来更新Actor网络参数$\phi$和评估网络Critic参数$\omega$，这2个参数在系统时间开始时按照标准正态分布进行随机初始化。

3.2.1 Critic网络

Critic网络将MSE编码后的状态S和动作A作为输入，反馈出$Q^{\pi}(S,A)$的期望，代表着当前环境下做出动作A获取的长期价值[27]，其结构如图4所示。用神经网络$Q(S_{k,t},A_{k,t};\omega)$来近似$Q^{\pi}(S_{k,t},A_{k,t})$，通过式(14)更新网络$Q(S,A;\omega)$：

$$L_{\text{critic}}(\omega)=E\left[\frac{1}{2}\left(Q(S_{k,t},A_{k,t};\omega)-\hat{Q}(S_{k,t},A_{k,t};\omega)\right)^2\right] \tag{14}$$

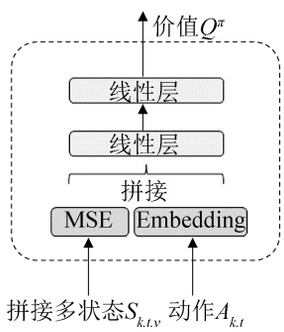

图4 Critic网络结构
Fig. 4 Critic network structures

为了防止高估问题，网络的更新用到了一个Target Soft$Q$[28]，将其表示为$\hat{Q}(S_{k,t},A_{k,t};\omega)$，采用经验回放池来更新Critic。每一步训练会将经验存储到经验池中，在开始训练前先进行经验收集，经验池存满后，在每次迭代中，从经验池中选取一批经验通过最小化损失函数$L_{\text{critic}}(\omega)$来更新Critic网络。

3.2.2 Actor网络

Actor表示为神经网络$\pi(a|s;\phi)$，其中，ϕ为神经网络参数，$\pi(a|s;\phi)$的功能是输出相应的图像任务卸载、子信道分配和的联合决策，其结构如图5所示。

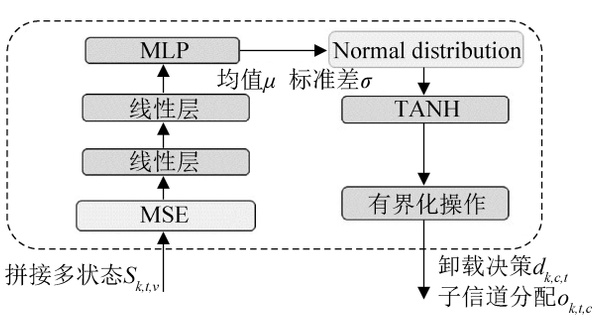

图5 Actor网络结构
Fig. 5 Actor network structures

Actor网络以多历史状态$S_{k,t,v}$为输入，经过MSE后，由MLP输出均值$\mu_{k,t,a}$和标准差$\sigma_{k,t,a}$，然后以$\mu_{k,t,a}$和$\sigma_{k,t,a}$生成正态分布的无界动作，通过tanh激活函数将动作值进行非线性映射，将无界动作进行有界化操作得到有界动作[22]，将神经网络π输出的动作表示为

$$A_{k,t}(d_{k,t},o_{k,t})=\xi\times\tanh\left(\mathcal{N}(\mu_{k,t,a},\sigma_{k,t,a})\right)+\mathcal{G} \tag{15}$$

式中：ξ、$\mathcal{G}$为有界化参数。Actor网络的更新与Critic网络相似，在每次迭代中，从经验回放池中抽取一批经验通过最小化函数$L_{\text{actor}}(\phi)$更新网络：

$$L_{\text{actor}}(\phi)=E\left[\alpha\ln\pi_{\phi}(A_{k,t}|S_{k,t})-Q_{\omega}^{\pi}(S_{k,t},A_{k,t})\right] \tag{16}$$

式中：α为熵的权重；ϕ为Actor网络的参数。

3.3 F-Net

为了将联合决策进行解耦，本文提出了F-Net以确定传输图像的压缩率，在更新SAC-MSE网络之前，用监督学习的方式异步训练F-Net网络。F-Net通过监督学习直接从带有标签的数据中学习，而不需要进行试错和奖励信号的反馈，因此，能够加快算法的训练和决策过程，提高算法效率。F-Net的训练流程如图6所示。

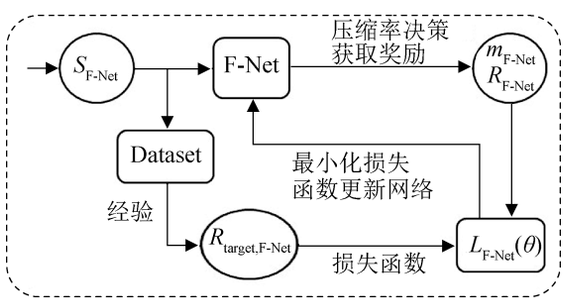

图6 F-Net训练流程
Fig. 6 F-Net training process

F-Net以当前状态即信道条件和图像信息等作为输入，输出压缩率决策并由奖励函数反馈奖励。将F-Net的状态空间定义为$S_{\text{F-Net}}=(s,\beta,h)$，其中，$s$表示图像大小，$\beta$表示图像前端识别的置信度，$h$表示信道增益。将动作空间定义为$m_{\text{F-Net}}=\{1,2,\cdots,N_m\}$，图像可选的压缩率个数为$N_m$。奖励函数为

$$R_{\text{F-net}}=J/\mathrm{e}^{\mu t_{k,c,t,\text{tb}}}+\rho \tag{17}$$

奖励函数$R_{\text{F-Net}}$表示在图像压缩率确定的情况下，传输到后端MEC服务器所获取的奖励。根据数据集中不同信道条件下的图像的模拟经验，能够得到不同状态下动作的最优动作选择，并假设

DOI: 10.16182/j.issn1004731x.joss.23-0576

每个状态的目标奖励表示为$R_{\text{target, F-Net}}$，通过最小化损失函数(18)更新F-Net。

$$L_{\text{F-Net}}(\theta)=E\left[(R_{\text{F-Net}}-R_{\text{target, F-Net}})^2\right] \tag{18}$$

3.4 算法描述

没训练SAC-MSE算法的伪代码如算法1所示。

算法1：SAC-MSE算法训练过程

初始化：系统状态$S_{k,0}$、F-Net网络参数θ、Actor网络参数ϕ、Critic网络参数ω，设定F-Net、SAC-MSE经验回放池D、$D_{\text{F-Net}}$，设定F-Net最大训练步数$W_{\text{max, F-Net}}$，SAC-MSE最大训练轮数$W_{\text{max, SAC-MSE}}$

while $W<W_{\text{max, F-Net}}$ do

for images in training set do

获取状态$S_{\text{F-Net}}$

F-Net根据$\text{argmax}_A Q(S_{\text{F-Net}},A;\theta)$

返回动作图像压缩率决策$m_{\text{F-Net}}$

执行$m_{\text{F-Net}}$获取奖励$R^{F\text{-net}}$

获取target奖励$R_{\text{target, F-Net}}$与奖励$R_{\text{F-Net}}$一起存入$D_{\text{F-Net}}$

end for

$D_{\text{F-Net}}$中抽取经验并根据式(18)更新网络参数

end while

while $W<W_{\text{max, SAC-MSE}}$ do

获取拼接历史状态$S_{k,t,v}$，并通过MSE编码得到编码状态$S_{k,t}$

Actor输入状态$S_{k,t}$由(15)输出卸载决策$d_{k,c,t}$，信道分配决策$o_{k,c,t}$，若卸载到后端则由F-Net输入图像信息输出压缩率决策$m_{k,c,t}$

执行动作并获取奖励$R_{k,t}$与下一时刻状态$S_{k,t+1}$

将$[S_{k,t},A_{k,t},R_{k,t},S_{k,t+1}]$存入$D$

每z步从D中抽取一批经验，根据式(14)更新Critic网络，根据式(16)更新Actor网络

end while

4 实验与仿真

4.1 实验环境与参数

本文在windows 10系统下使用Python3.7和Pytorch框架对MEC边缘监控系统进行仿真实验。首先，实验建立了边缘监控仿真环境，并实现了传统强化学习DQN算法、SAC算法、传统启发式GREEDY算法和任意策略4种算法与SAC-MSE算法进行对比实验。然后，通过实验分析参数ρ对各个算法性能的影响程度。最后，实验对不同系统状态下SAC-MSE算法决策的合理性进行分析，系统部分参数设定如表1所示。

表1 仿真参数设定

Table 1 Simulation parameter settings

参数	预设值
继电器的数量K	10
连接到继电器的摄像头数量C	3
传输功率P/dBm	15
噪声功率p_N/dBW	−97
总带宽B/MHz	7
学习率α	1×10^{-5}
折扣率γ	0.9
子信道带宽B_{sub}/KHz	700
有界化参数ξ	1.5
有界化参数$\mathcal{G}$	0.5
奖励函数参数ρ	16
历史状态个数v	2

本文通过Rayleigh衰落信道的Jakes衰落模型计算信道增益，继电器和基站的距离设为30 m，路径衰落系数设为3.5，假设小尺度衰落在10 ms内保持不变，在设定范围外根据Jakes衰落模型变化，以64个神经元的单层网络作为F-Net用监督学习方式训练，算法所使训练集和测试集分别由6 000和1 500对人脸图像信息组成[17]。

4.2 仿真实验结果

实验对各算法进行了1 800轮的训练，各算法的每轮平均奖励收敛过程如图7所示。

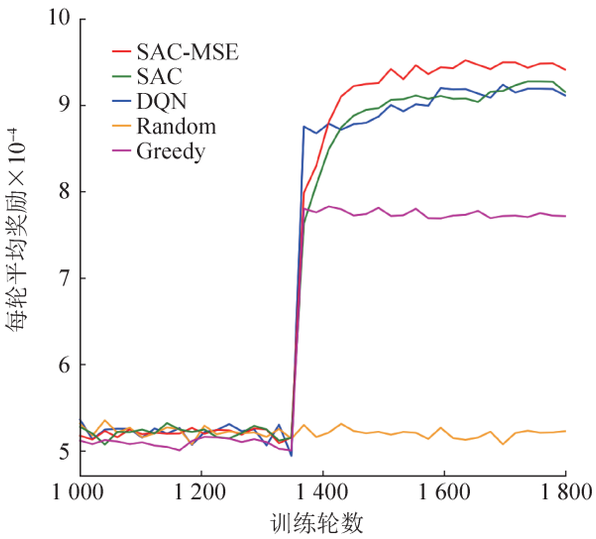

图7 各算法平均奖励收敛性
Fig. 7 Convergence of average reward

由图7可知，相比其他4种算法，SAC-MSE算法的每轮奖励之和收敛到了最高值，传统的SAC算法的平均奖励收敛值仅次于SAC-MSE。这说明通过多状态编码器MSE从多时序状态中提取到深层次的语义信息，从而提高了算法性能，证明了MSE的有效性。DQN算法的奖励比SAC略低，且收敛性相对较差，任意策略收敛的奖励值最小。传统启发式GREEDY算法收敛速度较快，但收敛值低于强化学习算法，这是因为传统启发式Greedy算法无法有效利用先验知识，容易陷入局部最优解。

图8~9分别为各个算法在平均系统延迟和识别精度2个性能指标的收敛结果，SAC-MSE在较短的回合数内收敛到了最优值，SAC算法的收敛值小于SAC-MSE，DQN算法即使在1 800轮结束也没有呈现出良好的收敛性。由图7~9可知，基于强化学习的算法在平均系统延迟和平均识别精度两方面的收敛值都优于Greedy算法。

为了调整算法决策的趋势，在奖励函数中引入了超参数ρ，一个合适的ρ可以避免决策偏向于前端或者后端，从而充分利用系统计算资源和带宽资源，提高系统效率。实验对不同超参数ρ下各个算法性能进行了对比，图10为各算法的识别精度对比。由图10可知，SAC-MSE的识别精度优与其他4种算法，而且SAC-MSE和SAC的总识别精度和ρ的值为正相关关系。随着ρ的提升，总识别精度也随之提升，这意味着有更多的图像识别任务被卸载到后端MEC服务器执行，因为后端识别算法的精度更高，所以系统识别精度也呈上升趋势。

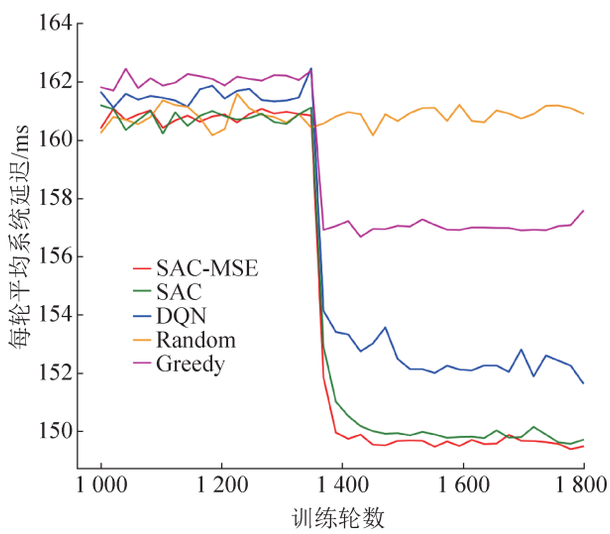

图8 各算法平均系统延迟收敛性
Fig. 8 Convergence of average system delay

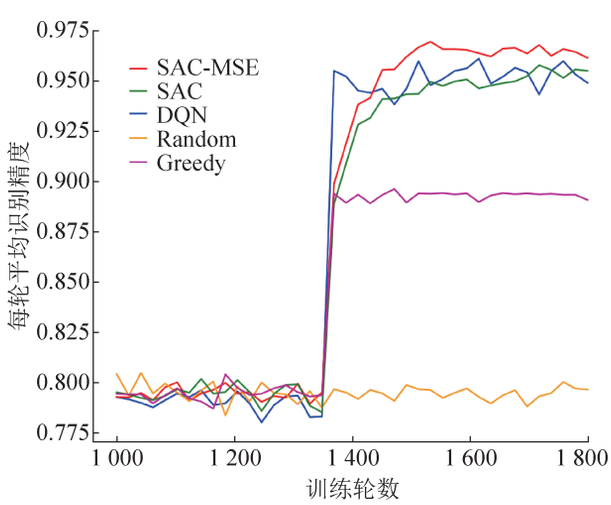

图9 各算法平均识别精度收敛性
Fig. 9 Convergence of recognition accuracy

由图11~12可知，随着ρ的提升DQN算法、SAC和SAC-MSE延迟会大幅变大，由于更多的图像识别任务卸载到后端，这样会造成更多任

DOI: 10.16182/j.issn1004731x.joss.23-0576

务抢占信道资源导致无线信道条件恶化，没有分配到信道资源的任务倾向于等待，这样会使系统延迟大大增加。但是，从图 11~12 中可以看出，ρ 对于 Random 策略和 Greedy 算法的性能没有显著的影响，这是由于 Random 策略动作选择是随机的，奖励函数的改变不会影响到动作的选择，所以也不会影响到 Random 策略的算法性能。Greedy 算法选取动作规则不同于强化学习算法，其根据当前系统状态选择一个符合优化目标的最优解，由于不具备从经验中学习的能力，也不考虑全局状态和长期奖励，所以,奖励函数的变化不会影响其动作的选择，ρ 的改变也不会影响其算法性能。根据上述实验得出结论:对基于强化学习算法来说，合适的 ρ 能够保证系统整体卸载效率，ρ 值过大则会严重影响算法性能。

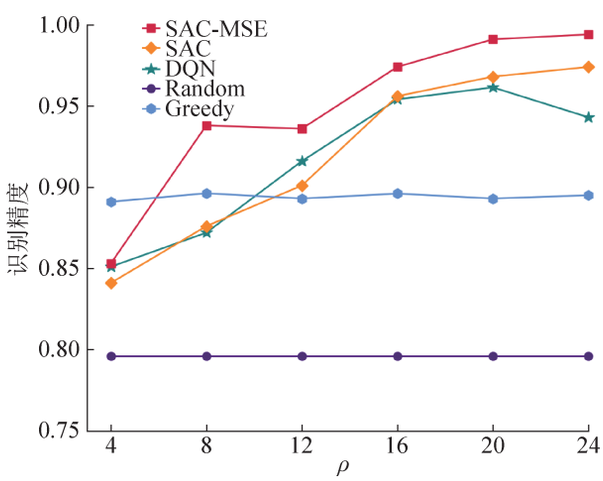

图 10　不同 ρ 下各算法识别精度

Fig. 10　Average recognition accuracy of various algorithms under different ρ

本文在相同参数下对于 4 种算法进行了测试，获取了平均性能指标，结果如表 2 所示。

由表 3 可知，SAC-MSE 的精度达到所有算法中的最高值 0.974，前端延迟和后端延迟也均低于其他 4 种算法。

为了验证 MSE 的有效性，实验对不同 Self-Attention heads 数量的算法性能进行了比较，此处超参数 $\rho=10$，结果如表 3 所示。

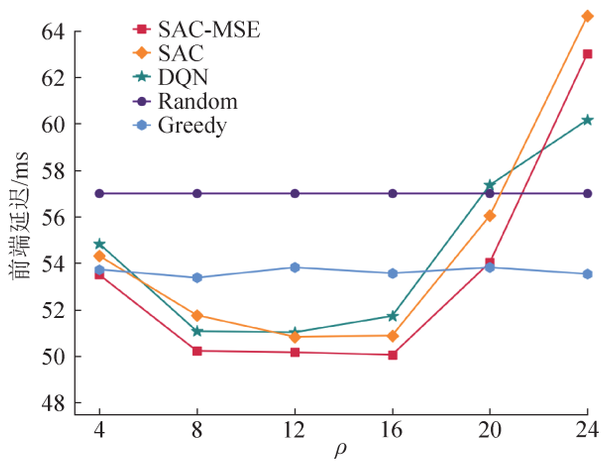

图 11　不同 ρ 下各算法平均前端延迟

Fig.11　Average front delay of various algorithms under different ρ

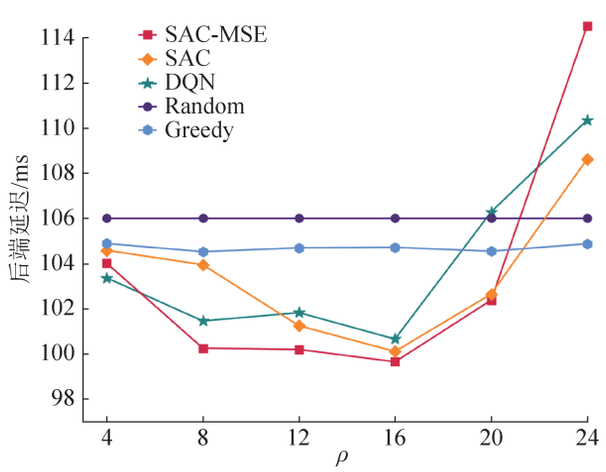

图 12　不同 ρ 下各算法平均后端延迟

Fig.12　Average back delay of various algorithms under different ρ

表 2　各算法性能比较

Table 2　Performance comparison of various algorithms

算法	精度	前端延迟/ms	后端延迟/ms
SAC-MSE	0.974	50.06	99.60
SAC	0.958	50.24	99.84
Random	0.798	55.83	104.76
DQN	0.942	53.56	99.75
Greedy	0.895	54.62	103.31

由表 3 可知，随着 Self-Attention heads 数量的提升，算法的总精度有所提升，系统的前端延迟有所下降。这是因为增加 Self-Attention heads 数量可以提高模型对不同状态信息的关注能力，从而

捕捉更多的局部特征和全局关系，更多注意力头数还可以增加模型的并行性，提高模型的训练和推理效率。

表3 不同head数量SAC-MSE算法性能
Table 3 Performance of SAC-MSE under different heads

head数	精度	前端延迟/ms	后端延迟/ms
1	0.945 4	50.275 6	99.702 5
2	0.947 6	50.192 7	99.695 1
7	0.949 8	50.164 1	99.781 2

上述实验结果验证了MSE对于算法做出更优决策有重要作用，然而，随着Self-Attention heads数量增加，也会增加模型的参数复杂度，从而增加模型的计算量和存储量。因此，本文在实验中选择了合理的Self-Attention heads数量。

4.3 不同系统状态下算法分析

实验通过不同系统状态下算法的决策动作概率对算法决策的合理性进行分析。为了便于统计数据，根据图像的大小和置信度大小，将图像任务总共分成5个集合，将图像从小到大分为5个集合，单位为bytes：size1=[0, 1 782)，size2=[1 782, 3 666)，size3=[3 666, 5 669)，size4=[5 669, 7 725)，size5=[7 725, ∞)。

将在集合中的图像任务卸载概率分为3种，分别为图像任务卸载到前端的概率A_{front}，图像任务在缓冲队列等待的概率A_{wait}与图像任务卸载到MEC后端的概率A_{back}。首先，实验分析了当信道可用时，图像的大小对于任务卸载动作的影响，统计数据结果如表4所示。

表4 不同图像大小集合下动作概率
Table 4 Action probability under different image size sets

集合	A_{front}	A_{wait}	A_{back}
size1	0.013 32	0.047 05	0.939 61
size2	0.024 68	0.047 76	0.927 54
size3	0.025 12	0.048 23	0.926 63
size4	0.033 69	0.048 49	0.917 80
size5	0.035 84	0.051 09	0.913 05

由表4可知，随着图像大小的增加，处于不同集合图像的A_{front}和A_{wait}会提升，A_{back}不断下降，这意味着如果图像越大，则由前端输出的概率就越大，其原因是图像大小会直接影响整个系统的延迟，为了使系统延迟维持在一个较低水平，算法会在保证识别精度的基础上，将数据量大的图像交由前端处理。随着图像卸载到前端的概率A_{front}的上升，图像之间竞争信道资源的现象也会得到缓解，从而提高系统效率。

针对卸载到后端的图像，实验分析了不同大小集合的图像选择不同压缩率的概率。将图像的5种压缩率从低到高依次表示为$m_1, m_2, \cdots, m_5$，数据结果如表5所示。由表5可知，图像变大时选择低压缩率的概率会减少，选择高压缩率的概率会增加。这是由于压缩率影响了总处理时间和后端识别精度。在不过多影响识别精度的情况下，算法倾向于选择较大的压缩率使图像压缩后数据量更小，从而减少传输延迟，节省无线信道资源。

表5 不同图像大小集合下选择压缩率的概率
Table 5 Probability of selecting compression rate for images under different size sets

集合	m_1	m_2	m_3	m_4	m_5
size1	0.915 80	0.032 97	0.017 86	0.015 94	0.017 41
size2	0.831 59	0.057 53	0.036 80	0.034 34	0.039 72
size3	0.815 36	0.067 58	0.041 89	0.037 31	0.037 84
size4	0.789 79	0.077 30	0.046 19	0.041 90	0.044 79
size5	0.778 31	0.075 43	0.052 24	0.043 26	0.050 75

实验对不同置信度的图像卸载动作进行了分析，将置信度从小至大分为5个集合：β_1=[0, 0.117)，β_2=[0.117, 0.139)，β_3=[0.139, 0.217)，β_4=[0.217, 0.314)，β_5=[0.314, 1)。

统计结果如表6所示，由表6可知，当图像的置信度越大时，其卸载到前端的概率就越大，这是因为置信度代表着图像由前端直接识别成功的可信程度，置信度越高则前端识别成功的概率越高。为了避免任务抢占信道，减少图像处理时间，

DOI: 10.16182/j.issn1004731x.joss.23-0576

置信度高的图像应该由前端的轻量级识别算法直接输出识别结果。

表6 不同图像置信度集合的动作概率

Table 6 Action probability of different image confidence sets

置信度	A_{front}	A_{wait}	A_{back}
β_1	0.002 07	0.048 37	0.949 55
β_2	0.003 53	0.053 77	0.942 68
β_3	0.004 82	0.063 72	0.931 44
β_4	0.015 13	0.055 18	0.929 67
β_5	0.032 58	0.066 93	0.900 47

5 结论

本文主要研究工作总结如下。

(1) 建立多摄像头的边缘监控人脸识别系统，以提升识别精度和降低系统延迟为目标，研究系统中的任务卸载、无线信道分配和图像压缩率选择的联合决策。由于时变的无线信道条件的无法预测性和识别任务到达的随机性，本文将该问题表述为一个马尔可夫决策过程。

(2) 为了避免巨大的状态空间和动作空间导致维数诅咒，本文将联合决策问题解耦：首先，提出基于深度强化学习算法SAC-MSE进行任务卸载和信道分配决策。然后，提出F-Net网络通过监督学习方式为卸载到后端的任务选择压缩率进行压缩传输，F-Net以当前系统状态条件和图像信息等作为输入，输出图像的压缩率决策。解耦决策问题能够有效降低状态空间的维度，加快算法的训练和决策过程。

(3) 提出多状态编码器MSE，对时序的多个系统状态信息充分利用，通过捕捉连续多时隙状态之间的依赖关系增强网络状态特征的表征能力，以此提高算法对于随机系统状态的鲁棒性，提升模型收敛速度。将MSE作为深度强化学习软演员评论家算法的主干网络，通过训练SAC-MSE，求解边缘视频监控中的识别任务卸载，信道资源分配问题。

实验结果表明：在算法收敛性方面，SAC-MSE算法收敛速度较快且识别精度和系统延迟都优于其它算法。超参数ρ对于算法性能至关重要，合适的ρ可以更好地提升算法效率。此外，SAC-MSE算法在不同图像大小，图像置信度等系统条件下也表现出了决策的合理性。由于本文算法主要对监控任务中的实时性和准确性进行研究，在仿真场景下假定监控设备和边缘设备都在能源充足的情况下进行，因此，暂未考虑能量消耗问题。另外，SAC-MSE算法添加了MSE作为主干网络，因此，其算法复杂度略高于传统强化学习算法。针对上述不足之处，未来将在保证监控任务实时性和准确性的基础上，从降低算法复杂度层面，开展对系统能量消耗优化的研究。

参考文献：

[1] Jiang Xiantao, Yu F R, Song Tian, et al. Intelligent Resource Allocation for Video Analytics in Blockchain-enabled Internet of Autonomous Vehicles with Edge Computing[J]. IEEE Internet of Things Journal, 2022, 9 (16): 14260-14272.

[2] Sibi C Sethuraman, Pranav Kompally, Srikar Reddy. VISU: A 3-D Printed Functional Robot for Crowd Surveillance[J]. IEEE Consumer Electronics Magazine, 2021, 10(1): 17-23.

[3] Chen Xinqiang, Ling Jun, Wang Shengzheng, et al. Ship Detection from Coastal Surveillance Videos Via an Ensemble Canny-gaussian-morphology Framework[J]. The Journal of Navigation, 2021, 74(6): 1252-1266.

[4] Wan Shaohua, Xu Xiaolong, Wang Tian, et al. An Intelligent Video Analysis Method for Abnormal Event Detection in Intelligent Transportation Systems[J]. IEEE Transactions on Intelligent Transportation Systems, 2021, 22(7): 4487-4495.

[5] Song Chunhe, Xu Wenxiang, Wu Tingting, et al. QoE-driven Edge Caching in Vehicle Networks Based on Deep Reinforcement Learning[J]. IEEE Transactions on Vehicular Technology, 2021, 70(6): 5286-5295.

[6] Chen Y Y, Lin Y H, Hu Yuchen, et al. Distributed Real-time Object Detection Based on Edge-cloud Collaboration for Smart Video Surveillance Applications [J]. IEEE Access, 2022, 10: 93745-93759.

[7] Xu Zhi, Li Jingzhao, Zhang Mei. A Surveillance Video Real-time Analysis System Based on Edge-cloud and FL-YOLO Cooperation in Coal Mine[J]. IEEE Access, 2021, 9: 68482-68497.

[8] Michele Girolami, Piergiorgio Vitello, Andrea Capponi,

et al. A Mobility-based Deployment Strategy for Edge Data Centers[J]. Journal of Parallel and Distributed Computing, 2022, 164: 133-141.

[9] 张依琳, 梁玉珠, 尹沐君, 等. 移动边缘计算中计算卸载方案研究综述[J]. 计算机学报, 2021, 44(12): 2406-2430.
Zhang Yilin, Liang Yuzhu, Yin Mujun, et al. Survey on the Methods of Computation Offloading in Mobile Edge Computing[J]. Chinese Journal of Computers, 2021, 44 (12): 2406-2430.

[10] Sun Jin, Yin Lu, Zou Minhui, et al. Makespan-minimization Workflow Scheduling for Complex Networks with Social Groups in Edge Computing[J]. Journal of Systems Architecture, 2020, 108: 101799.

[11] Liao Zhuofan, Peng Jingsheng, Xiong Bing, et al. Adaptive Offloading in Mobile-edge Computing for Ultra-dense Cellular Networks Based on Genetic Algorithm[J]. Journal of Cloud Computing, 2021, 10(1): 1-16.

[12] Gao Tieliang, Tang Qigui, Li Jiao, et al. A Particle Swarm Optimization with Lévy Flight for Service Caching and Task Offloading in Edge-cloud Computing [J]. IEEE Access, 2022, 10: 76636-76647.

[13] Wang Junhua, Zhu Kun, Chen Bing, et al. Distributed Clustering-based Cooperative Vehicular Edge Computing for Real-time Offloading Requests[J]. IEEE Transactions on Vehicular Technology, 2022, 71(1): 653-669.

[14] Guo Min, Huang Xing, Wang Wei, et al. HAGP: A Heuristic Algorithm Based on Greedy Policy for Task Offloading with Reliability of MDs in MEC of the Industrial Internet[J]. Sensors, 2021, 21(10): 3513.

[15] Xu Fei, Qin Zengshi, Ning Linpeng, et al. Research on Computing Offloading Strategy Based on Genetic Ant Colony Fusion Algorithm[J]. Simulation Modelling Practice and Theory, 2022, 118: 102523.

[16] 杨来义, 毕敬, 苑海涛. 基于SAC算法的移动机器人智能路径规划[J]. 系统仿真学报, 2023, 35(8): 1726-1736.
Yang Laiyi, Bi Jing, Yuan Haitao. Intelligent Path Planning for Mobile Robots Based on SAC Algorithm[J]. Journal of System Simulation, 2023, 35(8): 1726-1736.

[17] Yan Kunpeng, Shan Hangguan, Sun Tengxu, et al. Reinforcement Learning-based Mobile Edge Computing and Transmission Scheduling for Video Surveillance[J]. IEEE Transactions on Emerging Topics in Computing, 2022, 10(2): 1142-1156.

[18] Zhou Huan, Jiang Kai, Liu Xunun, et al. Deep Reinforcement Learning for Energy-efficient Computation Offloading in Mobile-edge Computing[J]. IEEE Internet of Things Journal, 2022, 9(2): 1517-1530.

[19] Chen Ying, Liu Zhiyong, Zhang Yongchao, et al. Deep Reinforcement Learning-based Dynamic Resource Management for Mobile Edge Computing in Industrial Internet of Things[J]. IEEE Transactions on Industrial Informatics, 2021, 17(7): 4925-4934.

[20] Tang Ming, Wong V W S. Deep Reinforcement Learning for Task Offloading in Mobile Edge Computing Systems [J]. IEEE Transactions on Mobile Computing, 2022, 21 (6): 1985-1997.

[21] Hu Haoji, Shan Hangguan, Wang Chuankun, et al. Video Surveillance on Mobile Edge Networks-A Reinforcement-learning-based Approach[J]. IEEE Internet of Things Journal, 2020, 7(6): 4746-4760.

[22] Wang Shuoyao, Bi Suzhi, Zhang Yingjun. Deep Reinforcement Learning with Communication Transformer for Adaptive Live Streaming in Wireless Edge Networks[J]. IEEE Journal on Selected Areas in Communications, 2022, 40(1): 308-322.

[23] Christian Blad, Simon Bøgh, Carsten Skovmose Kallesøe. Data-driven Offline Reinforcement Learning for HVAC-systems[J]. Energy, 2022, 261, Part B: 125290.

[24] Wang Junpeng, Zhang Wei, Yang Hao, et al. Visual Analytics for RNN-based Deep Reinforcement Learning [J]. IEEE Transactions on Visualization and Computer Graphics, 2022, 28(12): 4141-4155.

[25] Vaswani A, Shazeer N, Parmar N, et al. Attention is All You Need[C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc., 2017: 6000-6010.

[26] Haarnoja T, Zhou A, Abbeel P, et al. Soft Actor-critic: Off-policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor[C]//Proceedings of the 35th International Conference on Machine Learning. Chia Laguna Resort, Sardinia, Italy: PMLR, 2018: 1861-1870.

[27] Wu Jingda, Wei Zhongbao, Li Weihan, et al. Battery Thermal- and Health-constrained Energy Management for Hybrid Electric Bus Based on Soft Actor-critic DRL Algorithm[J]. IEEE Transactions on Industrial Informatics, 2021, 17(6): 3751-3761.

[28] Chen Chunyu, Cui Mingjian, Li Fangxing, et al. Model-free Emergency Frequency Control Based on Reinforcement Learning[J]. IEEE Transactions on Industrial Informatics, 2021, 17(4): 2336-2346.

DOI: 10.16182/j.issn1004731x.joss.23-0576