研究与开发

星地边缘计算网络中基于 CA3C的异构任务卸载方法研究

刘治国[1,2]，金晓勇[1,2]，汪林[3]，潘成胜[4]

（1. 大连大学信息工程学院， 辽宁 大连 116622；
2. 大连大学通信与网络重点实验室， 辽宁 大连 116622；
3. 大连大学环境与化学工程学院， 辽宁 大连 116622；
4. 南京信息工程大学电子与信息工程学院， 江苏 南京 210044）

摘　要：卫星边缘计算弥补了地面网络覆盖面有限的缺点，能够为稀疏网络环境下的用户提供高质量服务。但卫星边缘服务器资源受限以及用户的差异需求为制定灵活有效的任务卸载方法带来了困难。针对因异构任务卸载决策不合理而导致的卸载时延过高与能耗过大的问题，融合软件定义网络（software defined network，SDN）的卫星-地面边缘计算网络（satellite-ground edge computing network，SGECN）架构的系统模型下，提出一种基于卷积异步优势 Actor-Critic（convolutional asynchronous advantage Actor-Critic，CA3C）算法的异构任务卸载方法，通过引入动态权重分配机制根据任务特征自适应地调整时延和能耗权重，并采用卷积神经网络改进了 A3C（asynchronous advantage Actor-Critic）算法的网络结构，解决了异构任务多样化需求以及算法收敛速度慢的问题。仿真结果表明，CA3C在显著降低异构任务卸载时延与能耗方面具有优越性能。

关键词：卫星边缘计算；软件定义网络；异构任务卸载；深度强化学习

中图分类号：TN927.2

文献标志码：A

doi: 10.11959/j.issn.1000−0801.2026124

Research on heterogeneous task offloading methods based on CA3C in satellite-ground edge computing networks

Liu Zhiguo[1,2], Jin Xiaoyong[1,2], Wang Lin[3], Pan Chengsheng[4]

1. School of Information Engineering, Dalian University, Dalian 116622, China
2. Key Laboratory of Communication and Network, Dalian University, Dalian 116622, China
3. School of Environmental and Chemical Engineering, Dalian University, Dalian 116622, China
4. School of Electronics and Information Engineering, Nanjing University of Information Science and Technology, Nanjing 210044, China

Abstract: Satellite edge computing is proposed to make up for limited coverage of the terrestrial network，it can provide high-quality service for users in sparse network environment. However, limited resources of satellite edge servers

收稿日期：XXXX−XX−XX；修回日期：XXXX−XX−XX
通信作者：刘治国，liuzhiguo_dldx@163.com
基金项目：国家自然科学基金资助项目（No.61931004）
Foundation Item: The National Natural Science Foundation of China (No.61931004)

and diverse demands of users make it difficult to formulate flexible and effective task offloading methods. Aiming at the problems of high offloading delays and excessive energy consumption caused by unreasonable heterogeneous task offloading strategy, a heterogeneous task offloading method based on the convolutional asynchronous advantage Actor-Critic (CA3C) algorithm was proposed based on the system model of satellite-ground edge computing network (SGECN) architecture combined with software defined network (SDN). Delay and energy consumption weights were adaptively adjusted according to task characteristics by introducing a dynamic weight distribution mechanism, and a convolutional neural network was used to improve the network structure of A3C, the problems of diverse demands of heterogeneous task and slow convergence speed of the algorithm were solved. Simulation results show that the CA3C method performed better in effectively reducing the offloading time delay and energy consumption of heterogeneous task.

Key words: satellite edge computing, software defined network, heterogeneous task offloading, deep reinforcement learning

0 引言

随着人工智能与物联网的发展，各类智能设备会产生大量计算密集型与时延敏感型任务。然而用户端设备受限于其有限资源，难以按时完成任务。通过云计算技术将任务卸载到地面云，可以解决用户端设备计算能力有限的问题[1]。然而地面云通常距离用户较远，导致任务传输时延较高，会显著降低用户的体验质量。边缘计算（edge computing，EC）通过在靠近用户的地面基站部署边缘服务器，使得计算任务能够就近卸载，能够有效减少任务传输时延并改善用户的体验质量[2]。

然而由于经济和技术的限制，全球仅20%的陆地被地面网络覆盖，沙漠、海洋等偏远地区因缺少地面基站而无法接入地面网络[3]。此外，地面基站易受暴雨、地震等自然灾害的影响导致受灾区域用户无法使用地面网络，因此仅依靠地面网络难以满足全球用户的互联需求[4]。卫星网络由于其对地覆盖范围广且不受地面环境的影响，在应急通信、导航定位以及灾区救援等领域发挥着重要作用[5]。其中低轨道（low-earth orbit，LEO）卫星因其轨道高度低，在数据传输时延和路径损耗等方面表现优异，已成为当前研究热点。当用户不在地面网络服务范围内时，可借助LEO卫星将任务转发至地面云。然而当大量用户通过这种方式将数据传输到地面云时，存在高时延和网络阻塞问题[6]，通过将边缘计算服务器集成于低轨卫星上，偏远地区用户可将任务直接卸载到卫星边缘服务器，从而有效缓解高时延和网络拥塞等问题[7]。

目前已有许多关于卫星边缘计算中任务卸载与资源分配的研究，现有研究[8-12]大多采用最优化理论、元启发式算法或博弈论等方法。例如，Tang[8]等针对任务卸载决策问题，设计了一种两阶段优化方法，第一阶段采用非合作博弈论方法解决用户竞争问题，第二阶段通过凸优化方法实现带宽与计算资源的联合分配。Fang[12]等针对多低轨卫星场景下的计算卸载问题，将时延与能耗最小化目标分解为两个子问题：先采用凸优化方法进行资源分配，再通过局部搜索获取卸载问题的次优解。以上方法普遍依赖多轮迭代来求解，然而在周期性动态变化的卫星网络环境下，一旦系统状态发生变化，算法需要重新迭代导致求解效率低，难以满足卫星边缘计算对实时性的要求。

现有研究尝试将深度强化学习应用于卫星边缘计算的任务卸载与资源分配问题[13-19]。例如，Yang[15]等针对边缘服务器的过载问题，提出了一种基于D3PG（dueling double deterministic policy

gradients）的协同资源分配与计算卸载算法，采用 Double Dueling Q-learning 架构改进 DDPG 框架，解决了因 Q 值估计不准确导致的策略振荡与收敛缓慢问题；Zhou[19]等提出一种时空负载感知的自适应任务卸载方法，融合人口、经济因子及星上任务队列来评估时空负载，以解决卫星边缘计算中任务卸载的负载均衡问题。以上研究虽能区分不同场景下任务对时延与能耗的差异需求，但优化目标通常为预定义的时延与能耗的固定成本函数，其中时延和能耗的权重参数均为静态设定的固定值，未能充分考虑异构任务对服务质量的多样化需求。

由于卫星体积的限制，星上资源极为有限，若在星上训练模型会增加额外开销，集中训练–分布执行的方法[20-21]被应用来减少星上训练模型的系统消耗。例如，Zhang[20]等针对大规模低轨卫星网络中的星间协同计算问题，提出一种 COMA 算法，利用集中训练和分布执行有效降低了系统能耗与时延。Qin[21]等针对多用户协同卸载问题，提出了一种混合卸载框架，在保持分布式决策灵活性的同时，通过集中式 Critic 实现了全局优化。以上研究虽然将模型的训练和执行分开，有效减少了系统开销，但是低轨卫星缺少系统全局信息，其制定任务的效率会受到影响。本文通过将 SDN 与 GEO 卫星结合，利用 GEO 卫星的广域覆盖能力收集系统全局信息，系统可以制定更高效的任务卸载和资源分配策略。卫星边缘计算相关研究汇总见表 1。

本文工作贡献可以总结如下。

（1）设计了一种融合 SDN 的星地边缘计算网络架构 SGECN，其具备全局信息采集与集中控制能力，可显著提升任务卸载效率。

（2）针对 SGECN 中的异构任务卸载问题，优化目标是在资源约束下最小化任务卸载的时延和能耗，在未知任务类型的先验知识情况下，为每个异构任务设计了定制的卸载成本函数。

（3）提出了一种基于卷积异步优势 Actor-Critic（convolutional asynchronous advantage Actor- Critic，CA3C）算法，利用卷积神经网络改进了 A3C（asynchronous advantage Actor-Critic）的网络结构，加快了模型收敛速度。在仿真实验中，将 CA3C 与其他方法进行了比较，实验结果表明 CA3C 在减少任务卸载的时延和能耗方面优于其他方法。

1　系统模型

1.1　星地边缘计算网络架构

融合 SDN 的星地边缘计算（SGECN）架构如图 1 所示，SGECN 架构由部署 SDN 控制器的 GEO 卫星网络、部署边缘服务器的 LEO 卫星网络、地面用户以及地面云组成。假设用户位于偏远地区，其终端设备计算资源有限，仅能承担简单的计算任务；LEO 卫星搭载有边缘服务器，用户可将计算任务卸载至与其直连的 LEO 卫星，或通过星间链路中继至其他 LEO 卫星节点进行处理；地面云配备资源充足的云服务器，地面用户

表 1　卫星边缘计算相关研究汇总

文献	算法核心	优化目标	系统架构	是否考虑任务异构性
[8]	ADMM	能耗	LEO+地面云+本地	否
[13]	DDPG	能耗	LEO+地面云+本地	否
[15]	D3PG	时延	LEO+地面云+本地	否
[17]	SGTO-A	能耗+时延	卫星云+LEO+地面云	否
[19]	ATO-SLA	能耗+时延	LEO+本地	否
[20]	COMA	能耗	LEO	否
本文	CA3C	能耗+时延	LEO+地面云+本地	是

可通过LEO卫星中继转发将任务卸载到地面云。GEO卫星上部署有SDN控制器，并通过OpenFlow协议实现对卫星网络路由的集中控制，得益于地球同步轨道的广域覆盖特性，仅需3颗GEO卫星即可实现对LEO卫星网络的全域覆盖，将SDN控制器部署于GEO卫星上可以对系统信息进行集中管理和控制。

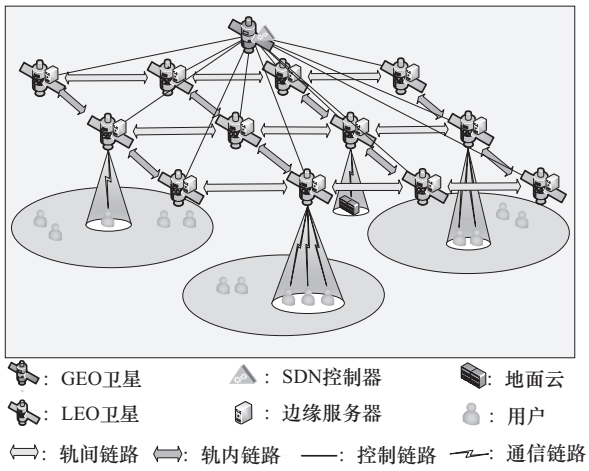


图1 融合SDN的星地边缘计算（SGECN）架构

与现有研究类似[17-19]，SGECN系统采用时隙化运行机制，将系统时间划分为等长时隙并于每个时隙起始时刻执行任务卸载操作。设SGECN中的LEO卫星集合为$L=\{l_1,l_2,\cdots,l_n,\cdots,l_N\}$，为了表示星上资源状态，对于任意LEO卫星表示为$l_n=(F_n,\text{remain}_f_n,\text{memory}_n,\text{wait}_Q_n)$，其中$F_n$表示星上边缘服务器的全部计算资源，$\text{remain}_f_n$表示剩余计算资源，$\text{memory}_n$表示存储容量，$\text{wait}_Q_n$表示当前等待处理的任务队列。假设在$t$时刻，地面多个用户端设备产生不可分割的计算任务，用户集合记为$U=\{1,2,\cdots,m,\cdots,M\}$，对于用户$m$的计算任务$\text{task}=(D_m,C_m,T_m)$，其中$D_m$[bit]表示任务数据量，$C_m$[cycle/bit]表示处理单位数据所需的中央处理器（central processing unit，CPU）周期数，T_m[s]表示任务最大容忍时延。每个用户任务可以在本地设备上执行，也可以卸载到LEO卫星上由边缘服务器处理，或者通过LEO卫星转发到地面云处理，因此有$N+2$个节点，为了表示用户m的任务具体卸载到哪个节点进行处理，使用决策变量$x_{m,j}\in\{0,1\}$来表示任务卸载决策。当$x_{m,j\in\{1,\cdots,n,\cdots,N\}}=1$时，表示任务被卸载到LEO卫星$n$上处理；当$x_{m,j\in\{N+1\}}=1$时，表示任务在本地执行；当$x_{m,j\in\{N+2\}}=1$时，表示任务被卸载到地面云处理。

1.2 成本模型

1.2.1 本地处理任务

（1）本地处理任务的时延

假设用户端设备的计算能力为f_m^{local}，计算任务在本地设备处理的计算时延为：

$$T_m^{\text{local}}=\frac{D_mC_m}{f_m^{\text{local}}} \tag{1}$$

（2）本地处理任务的能耗

用户端设备通常是由电池供电来维持正常的运行，而电池的寿命和容量是有限的，有必要考虑计算任务在本地设备处理的能耗，本地设备处理任务的能耗主要为计算能耗，如式（2）所示：

$$E_m^{\text{local}}=\omega_m\left(f_m^{\text{local}}\right)^2D_mC_m \tag{2}$$

其中，ω_m表示能耗因子，其值与用户设备所采用的CPU芯片架构相关[22-23]。

1.2.2 LEO卫星处理任务

假设用户m将任务卸载到LEO卫星，目标卸载卫星可能是与用户直连的卫星，也可能是LEO卫星网络中的其他卫星。为区分直连卫星和目标卸载卫星，使用n'表示直连卫星，使用n表示目标卸载卫星。由于任务被处理后的数据量很小，一般可以忽略向地面用户返回任务结果的时延[24]。

（1）LEO卫星处理任务的时延

① 星地传输和传播时延

当任务被卸载到直连卫星n'并由其处理时，直连卫星就是目标卸载卫星$(n'=n)$，地面用户通过上行链路将任务传输到直连卫星n'的传输和传

播时延如式（3）和式（4）所示：

$$t_{m,n'}^{\text{sat,trans}}=\frac{D_m}{R_{m,n'}^{\text{up}}} \tag{3}$$

$$t_{m,n'}^{\text{sat,prop}}=\frac{2d_{m,n'}}{\text{c}} \tag{4}$$

其中，$R_{m,n'}^{\text{up}}$表示上行链路数据传输速率；$d_{m,n'}$表示地面用户m与直连卫星n'的距离；c表示光速，其值为3×10^8m/s。

② 星间传输和传播时延

当地面用户与直连卫星n'相关联，然后通过直连卫星将任务转发到其他LEO卫星$n(n\in L\backslash\{n'\})$上时，传输时延主要源于星间链路的数据传输过程。部署于GEO卫星上的SDN控制器凭借其全局网络视图，能够实时获取整个LEO卫星网络的拓扑状态，SDN控制器通过运行最短路径算法为需要中继的任务计算出一条最优星间传输路径，并将任务传输的路由策略下发至相关的LEO卫星执行，由此产生的传输和传播时延如式（5）和式（6）所示：

$$t_{n',n}^{\text{sat,isl\_trans}}=\frac{D_m}{R_{n',n}^{\text{isl}}} \tag{5}$$

$$t_{n',n}^{\text{sat,isl\_prop}}=\frac{2d_{n',n}}{\text{c}} \tag{6}$$

其中，$R_{n',n}^{\text{isl}}$为星间链路数据传输速率；$d_{n',n}$为直连卫星n'与目标卸载卫星n之间的星间距离。

③ 星上排队和计算时延

当地面用户将任务卸载到目标卸载卫星n上，根据星上剩余资源情况，任务可能需要在任务队列wait_Q_n进行排队等待分配计算资源，任务队列按照先来先服务的规则为任务分配计算资源，排队时延和计算时延如式（7）和式（8）所示：

$$t_{m,n}^{\text{sat,wait}}=\begin{cases}0\text{，wait\_}Q_n=\varnothing\ \&\ \text{remain\_}f_n\neq\varnothing\\ \text{wait\_time}_{m,n}\text{，otherwise}\end{cases} \tag{7}$$

$$t_{m,n}^{\text{sat,comp}}=\frac{D_mC_m}{f_{m,n}^{\text{sat}}} \tag{8}$$

其中，$f_{m,n}^{\text{sat}}$为目标卸载卫星n分配给任务task$_m$的计算资源。

综上所述，任务task$_m$卸载到LEO卫星n上处理的总时延为：

$$T_{m,n}^{\text{sat}}=\begin{cases}t_{m,n'}^{\text{sat,trans}}+t_{m,n'}^{\text{sat,prop}}+t_{m,n}^{\text{sat,wait}}+t_{m,n}^{\text{sat,comp}},n'=n\\ t_{m,n'}^{\text{sat,trans}}+t_{m,n'}^{\text{sat,prop}}+t_{n',n}^{\text{sat,isl\_trans}}+t_{n',n}^{\text{sat,isl\_prop}}+\\ t_{m,n}^{\text{sat,wait}}+t_{m,n}^{\text{sat,comp}},n'\neq n\end{cases} \tag{9}$$

（2）LEO卫星处理任务的能耗

LEO卫星作为边缘计算节点，其能量供给完全依赖于太阳能，能源预算严格受限[8]，因此有必要考虑卫星边缘计算过程中的能耗。地面用户向LEO卫星卸载任务时，产生的能耗主要包含星地传输能耗、星间传输能耗及星上任务计算能耗，其中星地链路和星间链路的传输能耗占主要部分，执行任务的计算能耗作为一种持续且基础的系统开销，不仅直接消耗有限的能量，还会影响星上电子系统的寿命与可靠性。

① 星地传输能耗

任务task$_m$传输到直连卫星n'产生的能耗如式（10）所示：

$$e_{m,n'}^{\text{sat,trans}}=p_{m,n}\frac{D_m}{R_{m,n'}^{\text{up}}} \tag{10}$$

其中，$p_{m,n}$表示上行链路传输功率。

② 星间传输能耗

直连卫星n'将任务转发到其他LEO卫星n处理时，星间传输能耗如式（11）所示：

$$e_{n',n}^{\text{sat,isl\_trans}}=p_{n',n}\frac{D_m}{R_{n',n}^{\text{isl}}} \tag{11}$$

其中，$p_{n',n}$表示星间链路传输功率。

③ 星上任务计算能耗

任务task$_m$卸载到LEO卫星n上时，星上任务计算能耗如式（12）所示：

$$e_{m,n}^{\text{sat,comp}}=\omega_n(f_{m,n}^{\text{sat}})^2D_mC_m \tag{12}$$

其中，ω_n表示能耗因子，其值与星上边缘服务器所采用的CPU芯片架构相关。

综上所述，任务task$_m$卸载到LEO卫星n上处理的总能耗$E_{m,n}^{\text{sat}}$如式（13）所示：

$$E_{m,n}^{\text{sat}}=\begin{cases}e_{m,n'}^{\text{sat, trans}}+e_{m,n}^{\text{sat, comp}}, n'=n\\ e_{m,n'}^{\text{sat, trans}}+e_{n',n}^{\text{sat, isl\_trans}}+e_{m,n}^{\text{sat, comp}}, n'\neq n\end{cases} \quad (13)$$

1.2.3 地面云处理任务

用户将任务卸载到地面云的过程如下：首先用户将任务上传到直连卫星，然后直连卫星将任务转发给覆盖地面云的LEO卫星，最后将任务通过下行链路传输到地面云。

（1）地面云处理任务的时延

相对于将任务卸载到LEO卫星上处理，在地面云处理任务额外增加了LEO卫星将任务转发到地面云的传输和传播时延，如式（14）和式（15）所示：

$$t_m^{\text{cloud, trans}}=\frac{D_m}{R_{n,\text{cloud}}^{\text{down}}} \quad (14)$$

$$t_m^{\text{cloud, prop}}=\frac{2d_{n,\text{cloud}}}{\text{c}} \quad (15)$$

其中，$R_{n,\text{cloud}}^{\text{down}}$为下行链路数据传输速率；$d_{n,\text{cloud}}$为LEO卫星与地面云之间的距离。

假设地面云的计算能力为f^{cloud}，地面云处理任务的计算时延如式（16）所示：

$$t_{m,n}^{\text{cloud, comp}}=\frac{D_mC_m}{f^{\text{cloud}}} \quad (16)$$

综上所述，地面云处理任务的总时延T_m^{cloud}如式（17）所示：

$$T_m^{\text{cloud}}=t_{m,n'}^{\text{sat, trans}}+t_{m,n'}^{\text{sat, prop}}+t_{n',n}^{\text{sat, isl\_trans}}+t_{n',n}^{\text{sat, isl\_prop}}+t_m^{\text{cloud, trans}}+t_m^{\text{cloud, prop}}+t_m^{\text{cloud, comp}} \quad (17)$$

（2）地面云处理任务的能耗

由于地面云有充足的电力供应，一般可以忽略在地面云处理任务的计算能耗[25]，需要考虑LEO卫星将任务转发到地面云的传输能耗，如式（18）所示：

$$e_m^{\text{cloud, trans}}=p_{n,\text{cloud}}\frac{D_m}{R_{n,\text{cloud}}^{\text{down}}} \quad (18)$$

其中，$p_{n,\text{cloud}}$为下行链路传输数据的传输功率。

地面云处理任务的总能耗E_m^{cloud}如式（19）所示：

$$E_m^{\text{cloud}}=e_{m,n'}^{\text{sat, trans}}+e_{n',n}^{\text{sat, isl\_trans}}+e_m^{\text{cloud, trans}} \quad (19)$$

1.3 总成本模型

基于以上计算，完成任务task_m所需要的总时延T_m^{total}以及总能耗E_m^{total}如式（20）和式（21）所示：

$$T_m^{\text{total}}=\sum_{j=1}^{N}x_{m,j}T_{m,n}^{\text{sat}}+x_{m,N+1}T_m^{\text{local}}+x_{m,N+2}T_m^{\text{cloud}} \quad (20)$$

$$E_m^{\text{total}}=\sum_{j=1}^{N}x_{m,j}E_{m,n}^{\text{sat}}+x_{m,N+1}E_m^{\text{local}}+x_{m,N+2}E_m^{\text{cloud}} \quad (21)$$

SGECN系统中的任务具有异构性，不同任务对时延和能耗的偏好不同，采用固定权重来分配时延和能耗权重参数的方法并不能充分考虑异构任务对卸载成本的差异化需求。针对以上问题，本文采用一种动态权重分配机制[26]，具体表现为根据任务的最大容忍时延构建任务的时延权重参数，根据任务的数据量和计算强度构建任务的能耗权重参数。任务的最大容忍时延代表任务对时延方面的要求，而计算量和计算强度与处理任务所需能耗相关。

用户m的时延权重参数w_m^{time}的构建如式（22）所示，任务的最大容忍时延T_m越小，其对时延要求越高，构建任务task_m的时延权重参数w_m^{time}使用的是对最大容忍时延T_m进行反向归一化处理后的数据：

$$w_m^{\text{time}}=\frac{1}{2}\left[1+\tanh(1-T_m^{\text{norm}})\right] \quad (22)$$

用户m的能耗权重参数w_m^{energy}的构建如式（23）所示，任务task_m的计算量D_m和计算强度C_m与处理任务的能耗呈正相关，计算量和计算强度越大，处理任务所需的能耗就越多，其中C_m^{norm}和D_m^{norm}分别表示对C_m和D_m进行归一化处理后的数据：

$$w_m^{\text{energy}}=\frac{1}{2}\left\{1+\tanh\left[\frac{1}{2}(C_m^{\text{norm}}+D_m^{\text{norm}})\right]\right\} \quad (23)$$

综合以上计算，处理任务task_m的总成本cost_m如式（24）所示，其中$T_m^{\text{total, norm}}$和$E_m^{\text{total, norm}}$

分别表示对总时延 T_m^{total} 和总能耗 E_m^{total} 进行归一化处理后的数据：

$$\text{cost}_m = w_m^{\text{time}} T_m^{\text{total, norm}} + w_m^{\text{energy}} E_m^{\text{total, norm}} \quad (24)$$

本文优化目标是最小化任务卸载过程的总成本 cost_m，可以表示为：

$$\begin{aligned}
&\min \sum_{m=1}^{M} \text{cost}_m \\
&\text{s.t. C1: } x_{m,j} \in \{0,1\}, \\
&\quad \forall m \in M, j \in \{1,2,\cdots,N,N+1,N+2\} \\
&\text{C2: } \sum_{j=1}^{N+2} x_{m,j} = 1, \forall m \in M \\
&\text{C3: } f_{m,n}^{\text{sat}} \leqslant F_n, \forall m \in M, n \in \{1,2,\cdots,N\} \\
&\text{C4: } T_m^{\text{total}} \leqslant T_m, \forall m \in M \\
&\text{C5: } \sum_{j=1}^{N} x_{m,j} D_m \leqslant \text{memory}_n, \forall m \in M, n \in \{1,2,\cdots,N\}
\end{aligned} \quad (25)$$

其中，C1 表示任务的卸载决策 $x_{m,j}$ 为二进制变量；C2 表示任务 task_m 只能卸载到一个节点进行处理；C3 表示 LEO 卫星分配给任务的计算资源 $f_{m,n}^{\text{sat}}$ 不能超过最大计算资源 F_n；C4 表示处理用户任务 task_m 的总时延 T_m^{total} 不能超过最大容忍时延 T_m，否则任务处理超时；C5 表示卸载到 LEO 卫星的任务总数据量不能超过 LEO 卫星的最大数据存储容量 memory_n。

2　基于 CA3C 的异构任务卸载方案

2.1　CA3C-SGECN 系统架构

CA3C-SGECN 系统架构如图 2 所示，数据层由 LEO 卫星网络、地面用户和地面云组成，控制层由 GEO 卫星和 SDN 控制器组成。SDN 控制器主要实现 3 个功能：网络拓扑管理、路由管理和任务卸载决策管理。网络拓扑管理和路由管理主要负责拓扑相关和路由相关的服务，任务卸载决策管理的作用主要是将收集到的网络拓扑信息以及各节点信息作为状态输入智能体 CA3C，网络经过训练收敛输出动作为任务的最佳卸载位置以及计算资源分配方案。CA3C 采用异步并行训练机制，包含一个全局网络（Global Network）和若干线程网络（Worker Network）。其中全局网络部署于 SDN 控制器中，通过接收线程网络上传的参数来更新其网络；若干线程网络由资源丰富的地面训练中心负责训练。

CA3C-SGECN 实现任务卸载分为 3 个阶段，具体描述如下。

第一阶段是信息收集与状态感知阶段（图 2 中的□ 步骤）：地面用户发起任务卸载请求，SDN 控制器实时接收数据层的信息包括任务信息、低轨卫星的状态信息等，然后 SDN 控制器将这些信息传输至地面训练中心作为状态数据来训练线程网络。

第二阶段是智能体 CA3C 的在线训练与策略更新阶段（图 2 中的步骤□ ~□）：信息采集阶段所获取的状态数据输入智能体神经网络，其输出的卸载决策作为指令下发至 LEO 卫星，将任务卸载到指定节点进行处理，系统根据反馈计算奖励值，用于更新神经网络参数，循环执行步骤□ ~ 。该阶段采用异步并行训练机制，若干线程网络并行探索与学习，并通过梯度上传与参数下载实现全局网络的持续优化。由于卫星网络环境具有时变特性，CA3C 采用在线学习机制，能够在系统运行过程中不断适应环境变化，保持策略的实时性与有效性。

第三阶段是动态决策与任务执行阶段：SDN 控制器依据实时采集的信息调用 CA3C 模型实时生成任务卸载与资源分配策略，并将其下发至 LEO 卫星执行。该过程在每个时隙动态进行，从而确保系统在时变网络环境下仍能维持较高性能。

2.2　CA3C 算法设计

状态空间：包括两部分，第一部分为任务请求矩阵 $\mathbf{s\_task} = [\text{task}_1, \text{task}_2, \cdots, \text{task}_m, \cdots, \text{task}_M]^{\text{T}}$，第二部分为 LEO 卫星的状态矩阵 $\mathbf{s\_sat} = [l_1, l_2, \cdots, l_n, \cdots, l_N]^{\text{T}}$。

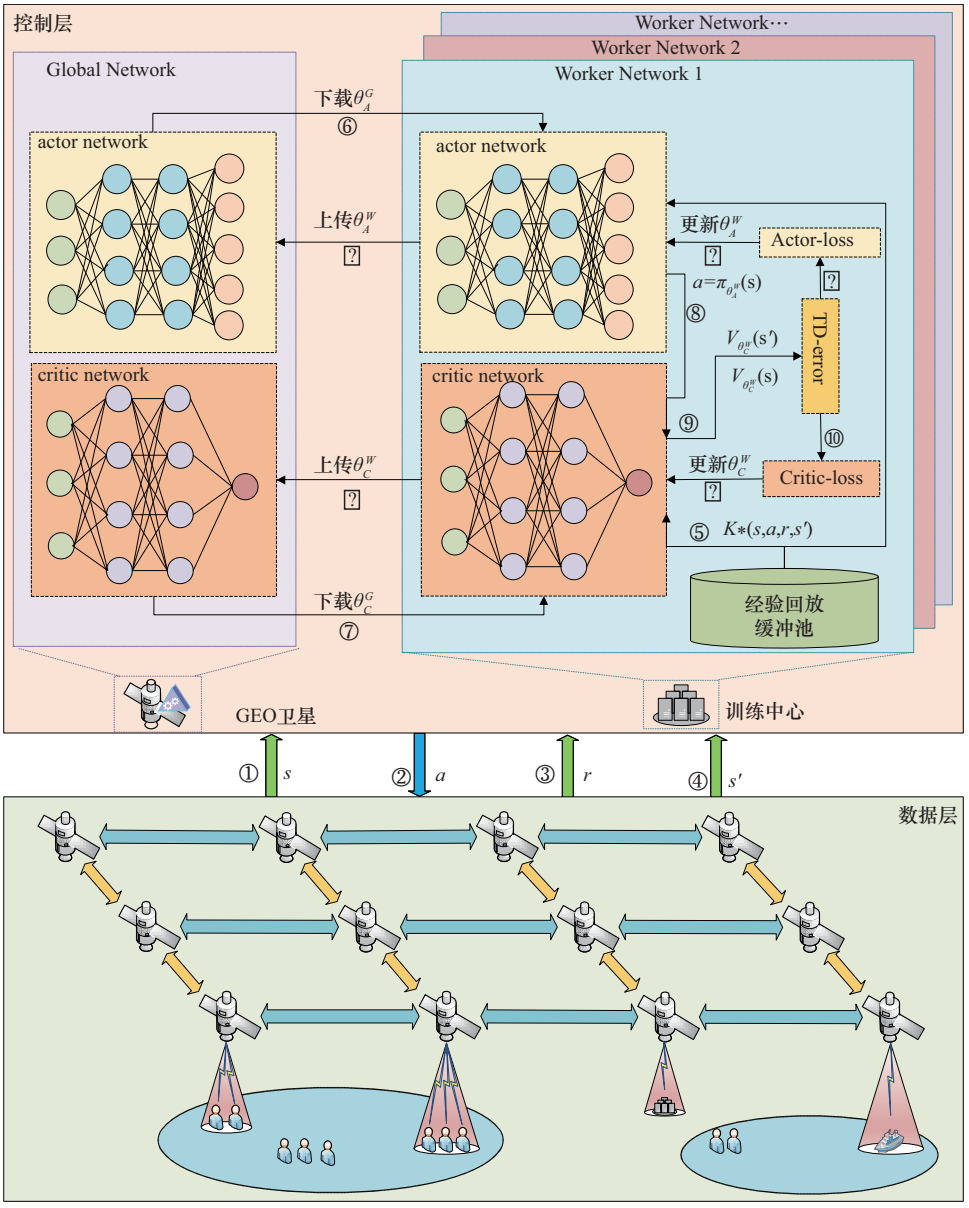


图2　CA3C-SGECN系统架构

动作空间：定义动作空间$\boldsymbol{a}=(m,j,f_{m,j})$，其中$m$表示任务请求编号；$j$表示任务的卸载节点编号；$f_{m,j}$表示卸载节点为任务分配的计算资源。若$j\in\{1,2,\cdots,N\}$表示任务被卸载到LEO卫星$n$，分配的计算资源为$f_{m,n}^{\text{sat}}$；若$j=N+1$表示任务在本地执行，分配的计算资源为$f_m^{\text{local}}$；若$j=N+2$表示任务在地面云处理，分配的计算资源为$f^{\text{cloud}}$。

奖励函数：本文优化目标是最小化任务卸载成本，深度强化学习的优化目标是最大化奖励值，因此将任务卸载成本的负数设为奖励值，同时为了平衡低轨卫星网络的负载，在奖励函数中添加负载均衡因子p，当LEO卫星网络平均负载上升时被激活，以抑制网络负载失衡。奖励函数如式（26）所示：

$$\boldsymbol{r}=\begin{cases}-\sum_{m=1}^{M}\text{cost}_m, & \text{if load}^{\text{ave}}\leqslant\text{load}^{\text{ave}},\\ -\sum_{m=1}^{M}\text{cost}_m-p, & \text{if load}^{\text{ave}}>\text{load}^{\text{ave}},\end{cases}\tag{26}$$

其中，load^{ave} 为当前时刻低轨卫星网络的平均负载；$\text{load}^{\text{ave}'}$ 为上一时刻低轨卫星网络的平均负载；p 为惩罚因子，其值被定义为当前时刻所有LEO卫星负载的标准差，用以衡量LEO卫星网络中各节点负载的不均衡程度。惩罚因子 p 越大，表明负载越集中于少数LEO卫星，LEO卫星网络的负载状态就越不均衡，因此对奖励施加的惩罚越大，智能体在制定卸载决策时会主动避免将负载集中到少数LEO卫星上，从而有效实现LEO卫星网络的负载均衡。惩罚因子 p 如式（27）所示：

$$p=\frac{1}{N}\sqrt{\sum_{n=1}^{N}(\text{load}_n-\text{load}^{\text{ave}})^2} \tag{27}$$

其中，load_n 为低轨卫星 n 的负载，根据低轨卫星上待处理的任务队列 $\text{wait\_Q}_n^{\text{sat}}$ 经过归一化处理得到。

LEO卫星网络的平均负载 load^{ave} 如式（28）所示：

$$\text{load}^{\text{ave}}=\frac{1}{N}\sum_{n=1}^{N}\text{load}_n \tag{28}$$

2.3　CA3C算法设计

A3C算法是一种将Actor-Critic架构与分布式训练方法相结合的深度强化学习算法，能够适应动态变化的环境，在卫星拓扑时变的星地边缘计算网络中处理任务卸载具备优势，这种异步训练框架能够提高网络训练效率，加快网络收敛速度[27]。本文将A3C中的深度神经网络（DNN）改进为卷积神经网络（CNN），传统DNN使用全连接层连接神经元，存在参数量过大导致计算缓慢以及网络容易过拟合的问题，CNN通过卷积操作提取数据局部特征，能够以较低计算开销处理大规模数据，从而显著提升网络收敛速度。

CA3C的全局网络和线程网络的结构相同，由Actor网络和Critic网络组成，Actor网络负责输出任务卸载与资源分配策略对应的动作，而Critic网络则对该策略的性能进行评估，然后Actor网络据此更新其动作的概率分布以优化策略，最终达到生成最优策略的目的。

Global Actor网络和Worker Actor网络采用相同的网络结构，Actor网络结构如图3所示。网络以状态 $\boldsymbol{s}$ 作为输入，经过卷积层然后通过激活函数进行非线性变换后，通过全连接层和激活函数进一步处理，最终输出动作 $\boldsymbol{a}$。

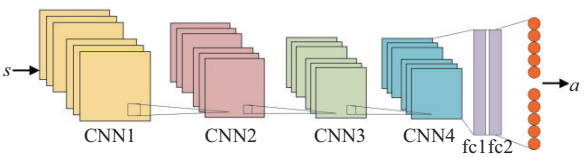


图3　Actor网络结构

Global Critic网络和Worker Critic网络的网络结构是相同的，Critic网络结构如图4所示。输入状态 $\boldsymbol{s}$ 和动作 $\boldsymbol{a}$，经过卷积层然后通过激活函数进行非线性变换，通过全连接层和激活函数，最终输出状态价值函数 $V(\boldsymbol{s})$。

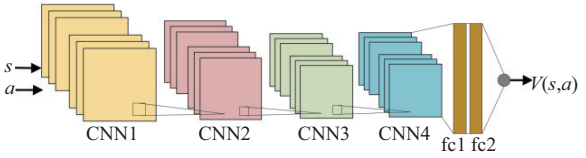


图4　Critic网络结构

CA3C结构如图2所示，记Global Actor网络为 $\pi_{\boldsymbol{\theta}_{\text{A}}^{\text{G}}}$，网络参数为 $\boldsymbol{\theta}_{\text{A}}^{\text{G}}$；记Worker Actor网络为 $\pi_{\boldsymbol{\theta}_{\text{A}}^{\text{W}}}$，网络参数为 $\boldsymbol{\theta}_{\text{A}}^{\text{W}}$；记Global Critic网络为 $V_{\boldsymbol{\theta}_{\text{C}}^{\text{G}}}$，网络参数为 $\boldsymbol{\theta}_{\text{C}}^{\text{G}}$；记Worker Critic网络为 $V_{\boldsymbol{\theta}_{\text{C}}^{\text{W}}}$，网络参数为 $\boldsymbol{\theta}_{\text{C}}^{\text{W}}$。CNN卷积层参数见表2。

表2　CNN卷积层参数

卷积层	输入通道数	输出通道数	卷积核	步长
一层	7	14	（3,3）	（2,2）
二层	14	28	（3,3）	（2,2）
三层	28	56	（3,3）	（2,2）
四层	56	112	（3,3）	（2,2）

CA3C算法流程如图5所示，首先在数据采样阶段，将状态$\boldsymbol{s}$输入Worker Network的Actor网络，Actor网络初始化参数$\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}}$并根据状态$\boldsymbol{s}$生成动作$\boldsymbol{a}$，智能体执行动作后得到环境反馈的奖励值$\boldsymbol{r}$，状态更新为下一时刻的状态$\boldsymbol{s'}$，重复采样直到状态达到终态（任务为最后一个任务）或达到最大迭代次数，收集经验$(\boldsymbol{s},\boldsymbol{a},\boldsymbol{r},\boldsymbol{s'})$存放在经验回放区。

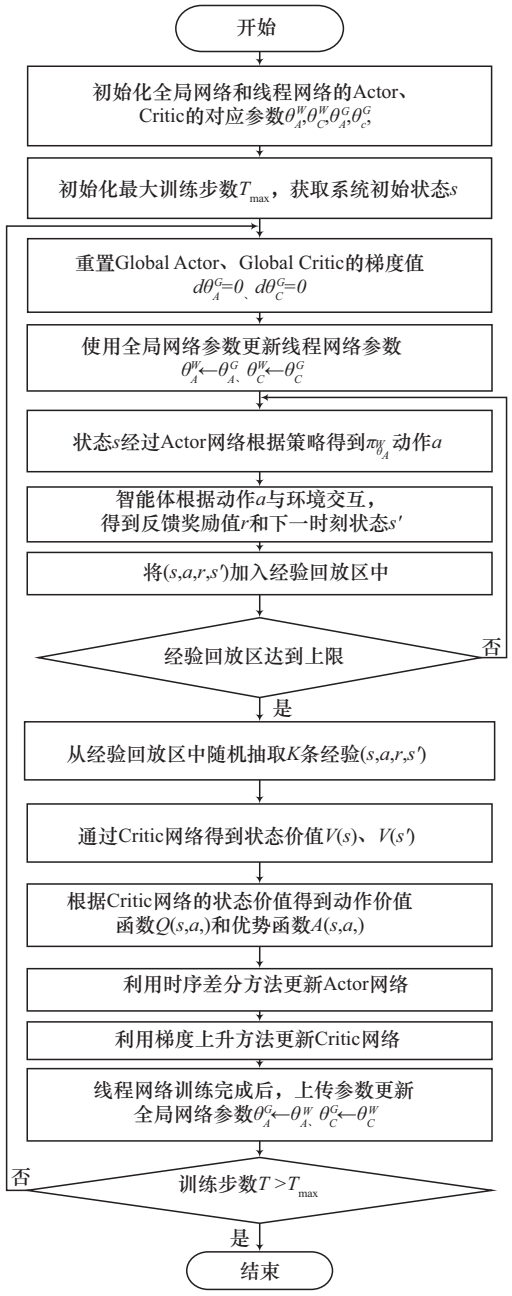


图5　CA3C算法流程

在网络训练阶段，每个智能体从经验回放区选取样本，然后对Worker Actor网络和Worker Critic网络的参数进行更新，Worker Network将更新后的参数$\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}}$和$\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}$上传到Global Network，分别更新Global Actor网络和Global Critic网络，同时将Global Actor和Global Critic网络的参数$\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{G}}$和$\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{G}}$下载到Worker Network来更新Worker Actor和Worker Critic网络并继续训练。假设采样数据为$(\boldsymbol{s},\boldsymbol{a},\boldsymbol{r},\boldsymbol{s'})$，网络的更新过程为，首先利用Worker Critic网络得到状态$\boldsymbol{s}$和状态$\boldsymbol{s'}$的状态价值函数$V_{\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}}(\boldsymbol{s})$和$V_{\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}}(\boldsymbol{s'})$，其作用主要是评估状态，然后根据Bellman方程计算动作价值函数，如式（29）所示：

$$Q(\boldsymbol{s},\boldsymbol{a})=\boldsymbol{r}+\gamma V_{\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}}(\boldsymbol{s'}) \tag{29}$$

其中，$Q(\boldsymbol{s},\boldsymbol{a})$为动作价值函数，其作用是评估在状态$\boldsymbol{s}$下的动作$\boldsymbol{a}$的好坏；$\gamma$表示折扣因子。

优势函数$A(\boldsymbol{s},\boldsymbol{a})$计算如式（30）所示，其作用是减少网络中非均匀的高估或低估问题：

$$A(\boldsymbol{s},\boldsymbol{a})=Q(\boldsymbol{s},\boldsymbol{a})-V_{\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}}(\boldsymbol{s}) \tag{30}$$

根据优势函数$A(\boldsymbol{s},\boldsymbol{a})$可得到Worker Actor网络的损失函数，如式（31）所示：

$$\mathrm{loss}_{\mathrm{w\_actor}}=\log\pi_{\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}}}(\boldsymbol{a})\left[Q(\boldsymbol{s},\boldsymbol{a})-V_{\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}}(\boldsymbol{s})\right] \tag{31}$$

Worker Actor网络的累计梯度计算如式（32）所示：

$$\mathrm{d}\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}}\leftarrow\mathrm{d}\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}}+\partial\left(\mathrm{loss}_{\mathrm{w\_actor}}\right)/\partial\left(\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}}\right) \tag{32}$$

Worker Critic网络的损失函数计算如式（33）所示：

$$\mathrm{loss}_{\mathrm{w\_critic}}=\left[Q(\boldsymbol{s},\boldsymbol{a})-V_{\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}}(\boldsymbol{s})\right]^2 \tag{33}$$

Worker Critic网络的累计梯度计算如式（34）所示：

$$\mathrm{d}\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}\leftarrow\mathrm{d}\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}+\partial\left(\mathrm{loss}_{\mathrm{w\_critic}}\right)/\partial\left(\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}}\right) \tag{34}$$

最后，利用线程网络的累计梯度对全局网络的参数$\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{G}}$和$\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{G}}$分别进行更新，如式（35）

和式（36）所示：

$$\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{G}}=\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{G}}-\mathrm{d}\boldsymbol{\theta}_{\mathrm{A}}^{\mathrm{W}} \tag{35}$$

$$\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{G}}=\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{G}}-\mathrm{d}\boldsymbol{\theta}_{\mathrm{C}}^{\mathrm{W}} \tag{36}$$

2.4　CA3C算法复杂度分析

利用CA3C算法制定任务卸载决策的时间复杂度主要受其神经网络结构和输入输出变量的维数影响。输入层大小主要由状态空间决定，状态空间包括任务请求矩阵和低轨卫星状态矩阵，维度分别为3和4，其中任务数量为M，低轨卫星数量为N，因此输入层大小为$3M+4N$；输出层大小主要由动作空间决定，包括LEO卫星、地面云以及本地，因此输出层大小为$N+2$。Actor和Critic网络的卷积结构相同。其卷积层的总时间复杂度可以表示为$C_{\mathrm{CNN}}=2\sum_{i=1}^{I}[K_i^2\times C_i^{\mathrm{in}}\times C_i^{\mathrm{out}}\times H_i\times W_i]$，其中$I$表示卷积层数，$K_i$表示第$i$层卷积核大小，$C_i^{\mathrm{in}}$表示第$i$层卷积的输入通道数，$C_i^{\mathrm{out}}$表示第$i$层卷积的输出通道数，$H_i\times W_i$表示第$i$层卷积输出的特征大小；全连接层的总时间复杂度可以表示为$C_{\mathrm{FC}}=2\sum_{f=1}^{F}[D_f^{\mathrm{in}}\times D_f^{\mathrm{out}}]$，其中$F$表示全连接层数，$D_f^{\mathrm{in}}$表示第$f$层的输入维度，$D_f^{\mathrm{out}}$表示第$f$层的输出维度。处理一个样本时，Actor网络的时间复杂度为$O(C_{\mathrm{CNN}}+C_{\mathrm{FC}}+|N+2|)$，Critic网络的时间复杂度为$O(C_{\mathrm{CNN}}+C_{\mathrm{FC}}+1)$，因此单一Worker Network处理一个样本的总时间复杂度为$O\left(2\times(C_{\mathrm{CNN}}+C_{\mathrm{FC}})+|N|\right)$，其单次更新的时间复杂度为$O\left(B\times(2\times(C_{\mathrm{CNN}}+C_{\mathrm{FC}})+|N|)\right)$，其中$B$为训练批次大小。由此可得到CA3C全局单次更新的时间复杂度为$O\left(W\times B\times(2\times(C_{\mathrm{CNN}}+C_{\mathrm{FC}})+|N|)\right)$，其中$W$为Worker Network的数量。

3　仿真实验

3.1　仿真参数设置

仿真实验采用AGI STK 11.6.0模拟星地边缘计算网络环境，并基于python与pytorch实现神经网络的搭建与训练，低轨卫星网络采用由66颗卫星节点构成的铱星星座，铱星星座轨道数量为6条，每条轨道上分布有11颗卫星，轨道高度为780 km。GEO卫星数量设置为3颗以实现对铱星星座和地面的全覆盖，轨道高度为35 786 km。系统仿真模拟参数见表3[13,17,19,28]。

表3　系统仿真模拟参数

参数名称	值
本地设备计算速度	1 GHz
卫星边缘服务器计算速度	[5,25] GHz
地面云计算速度	300 GHz
任务数据量	[10，1 000] KB
任务计算强度	[0.5，3.0] Kcycles/bit
任务最大容忍时延	[0.5，10] s
本地设备能耗因子	2×10^{-18} J/cycles
卫星边缘服务器能耗因子	8×10^{-17} J/cycles
上行链路数据传输速率	300 Mb/s
星间链路数据传输速率	2.5 Gb/s
下行链路数据传输速率	1.5 Gb/s
上行链路传输功率	1 W
下行链路传输功率	5 W
星间链路传输功率	3 W
经验回放区容量	50 000
训练批量大小	128
折扣因子	0.9
学习率	0.001
训练次数	5 000

3.2　仿真结果分析

3.2.1　收敛性能分析

为评估CA3C算法的收敛性能，本节实验设计了不同线程网络数量训练场景，并对训练过程的奖励变化进行分析。为验证卷积神经网络结构相对于深度神经网络在A3C框架中的优势，实验也对比了A3C-CNN与A3C-DNN的奖励收敛表现。

不同线程网络数量的奖励值变化如图6所示，实验中测试了线程网络数量分别为1、2和3时的奖励值收敛情况。实验表明，当线程网络数量从

1 增加至 3 时，奖励收敛速度显著提升且最终收敛值更高，从单线程的−8.75 提升至三线程的−6.12，奖励均值提升了约 30.1%，说明多线程训练有效提升了策略性能；奖励方差也随线程网络数量增加而明显下降，从单线程的 8.21 下降至三线程的 7.17，降低了约 12.7%，表明训练过程的稳定性显著增强。单线程网络收敛缓慢且奖励波动较大，双线程网络在训练中期奖励趋于稳定，三线程网络在训练初期迅速收敛并维持在较高奖励水平，显著优于单线程和双线程的性能。因此，在后续实验中将线程网络数量设置为 3 个线程网络。

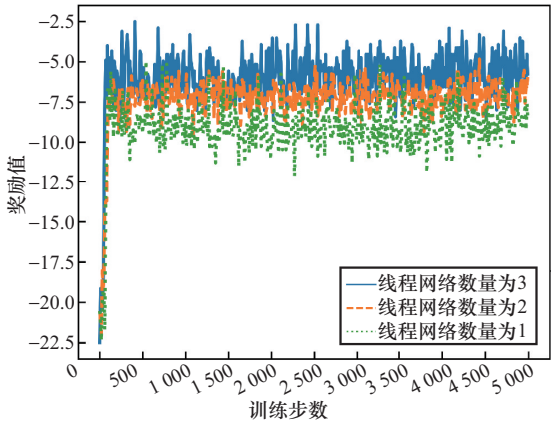


图 6　不同线程网络数量的奖励值变化

A3C-CNN 和 A3C-DNN 的奖励值变化如图 7 所示，为验证 CA3C 算法的收敛性，在超参数、线程网络数量等条件完全一致的前提下，实验测试了 A3C-CNN 和 A3C-DNN 的收敛情况，A3C-CNN 在 500 个训练步数左右就已经收敛，收敛奖励值稳定在−6.0 左右，而 A3C-DNN 则在 2 000 个训练步数左右才收敛，收敛奖励值则在−8.5 左右波动。A3C-CNN 算法的收敛速度更快，收敛奖励值更好，这一对比清晰表明 CNN 结构的引入显著提升了 A3C 的收敛性能。因为 CNN 能够自动提取局部特征，更精确的特征表示使 Actor 和 Critic 网络能够快速关联状态和动作价值，从而加快奖励收敛速度，其参数共享的特性也有效降低了模型过拟合的风险。A3C-CNN 和 A3C-DNN 的 reward 分析见表 4，A3C-CNN 的奖励均值为−6.12，比 A3C-DNN 提升了约 29.9%，A3C-CNN 的奖励方差为 7.17，比 A3C-DNN 降低了约 32.5%，表明 A3C-CNN 不仅能获得更优的策略性能，其稳定性也更好，能有效应对 SGECN 的高维与时变特性。

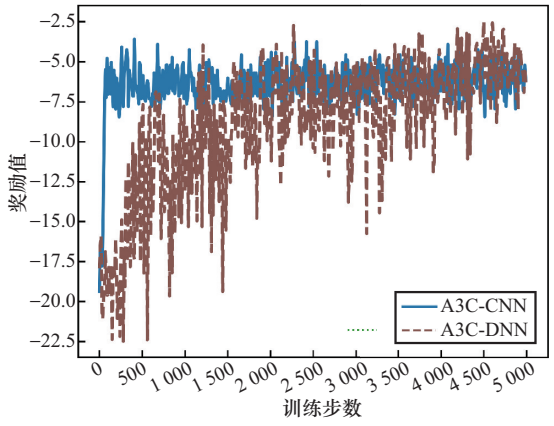


图 7　A3C-CNN 和 A3C-DNN 的奖励值变化

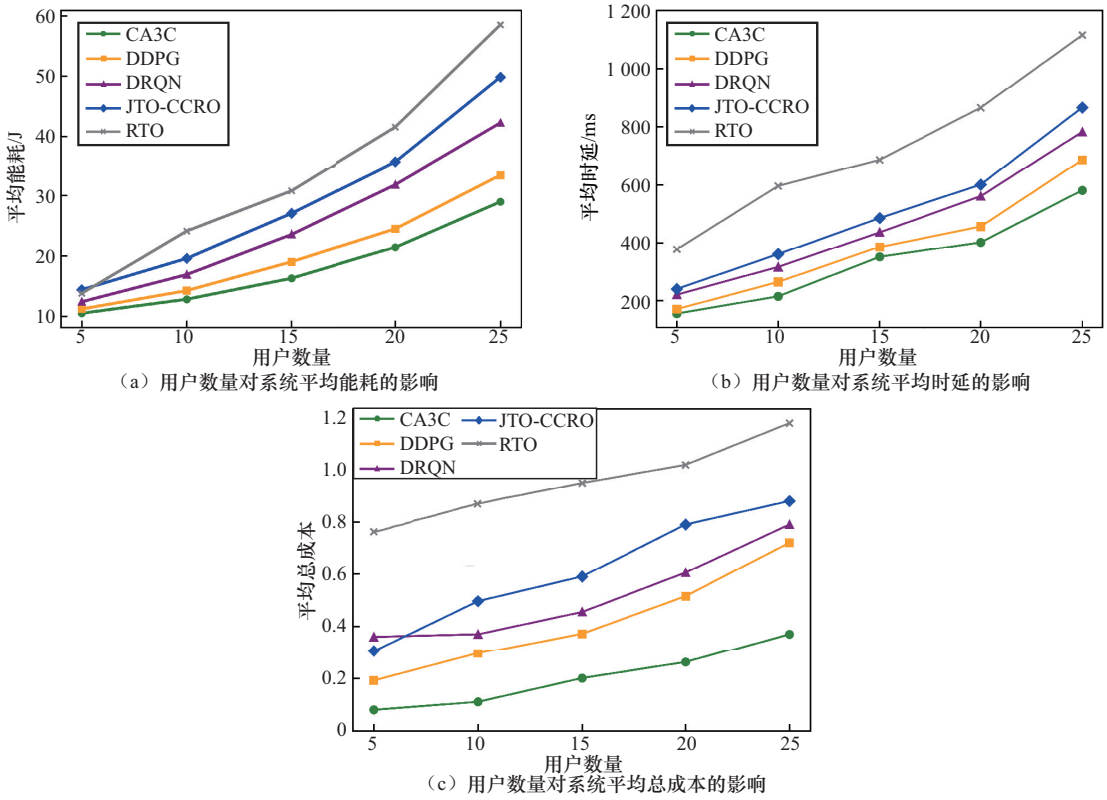
表 4　A3C-CNN 和 A3C-DNN 的 reward 分析

算法	奖励均值	奖励方差
A3C-DNN	−8.73	10.62
A3C-CNN	−6.12	7.17

3.2.2　算法性能比较

实验中将 CA3C 方法与基准方法进行了对比。随机任务卸载（random task offloading，RTO）方法：RTO 方法为任务随机选择卸载节点。基于博弈论的方法（JTO-CCRO）[29]：采用博弈论的方法通过迭代运算获得任务的卸载决策。DRQN[30]：一种基于 DQN 的任务卸载方法，并使用递归神经网络改进 DQN 的网络结构。DDPG[13]：采用深度确定性策略梯度算法为任务制定卸载决策。为确保对比实验的公平性与可靠性，以上基准方法均在本文所提出的系统模型下进行了重新实现与训练。

不同用户数量对系统的影响如图 8 所示，是不同用户数量情况下各算法的时延、能耗和总成



（a）用户数量对系统平均能耗的影响

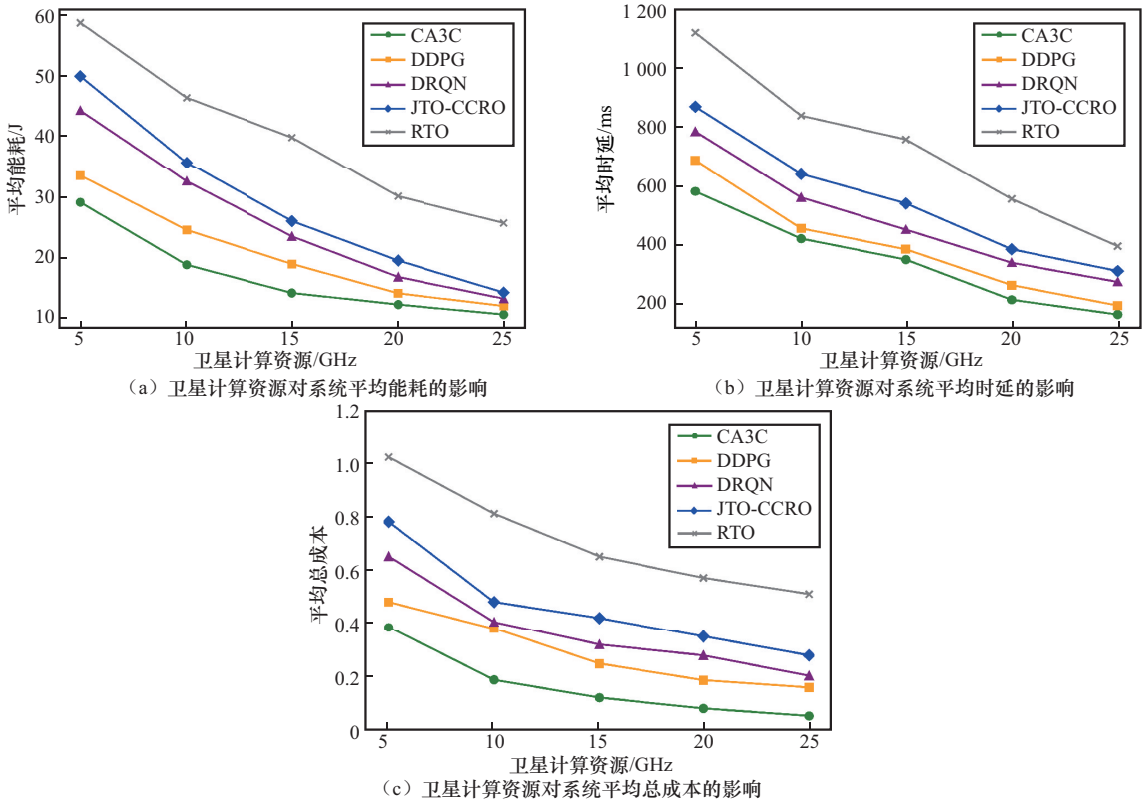
（b）用户数量对系统平均时延的影响

（c）用户数量对系统平均总成本的影响

图8　不同用户数量对系统的影响

本对比，随着用户数量的逐步增加，所有算法的平均时延、能耗以及总成本也随之增加，系统需要处理的任务规模也相应扩大，而低轨卫星的计算资源相对有限，导致卸载至卫星的任务可分配的计算资源减少。另外，随着任务数量的增加，待处理任务在队列中累积，超出了LEO卫星的处理能力，更多任务被卸载至地面云，增加了任务传输时延与能耗。与DDPG、DRQN、JTO-CCRO和RTO相比，CA3C的平均能耗分别降低了11.30%、28.9%、36.1%、43.25%，平均时延分别降低了12.8%、27.0%、33.8%、54.6%。CA3C的性能始终优于其他方法，因为随着任务数量增多，动作空间也随之扩大，RTO算法由于具有随机性，不能为任务制定合理的卸载决策导致其性能最差；JTO-CCR算法不能适应动态变化的环境，每次制定决策时需要经过大量迭代，在面对时变的卫星网络环境时性能较差；DRQN算法在处理连续动作空间时采用了离散化操作，导致其在处理大规模连续动作空间时难以达到最优性能；DDPG算法能够处理离散和连续的动作空间，但是优化系统成本时并未考虑异构任务的不同需求，导致其制定的卸载决策和资源分配策略并不是最合适的；相比之下，CA3C采用异步更新机制，具备处理大规模状态与动作空间的能力，同时兼顾异构任务的差异化需求，其制定的策略更优。

不同卫星计算资源对系统的影响如图9所示，是不同卫星计算资源配置下各算法的时延、能耗及总成本对比，随着卫星计算资源的增加，各算法的系统平均时延、能耗和总成本均呈下降趋势，与DDPG、DRQN、JTO-CCRO及RTO相比，CA3C的平均能耗分别降低了17.1%、32.2%、39.1%和58.1%，平均时延则分别降低了13.2%、29.9%、38.6%和54.1%，CA3C在时延、能耗和



（a）卫星计算资源对系统平均能耗的影响

（b）卫星计算资源对系统平均时延的影响

（c）卫星计算资源对系统平均总成本的影响

图9 不同卫星计算资源对系统的影响图

总成本方面表现更好。随着卫星计算资源的增加，边缘服务器能够为任务分配更多的计算资源，能够减少任务的处理时延。CA3C在降低时延和能耗方面表现更优，这表明CA3C在制定任务卸载和资源分配策略方面具有极高的效率；相比之下，RTO、JTO-CCRO、DRQN及DDPG方法由于前述原因，性能都要低于CA3C。综上所述，在不同卫星计算资源条件下，CA3C在降低系统时延和能耗方面的表现均显著优于对比算法。

4 结束语

为解决星地边缘计算中因异构任务卸载决策不合理而导致的卸载时延过高与能耗过大的问题，本文在SDN架构下提出了基于CA3C的卫星-地面边缘计算异构任务卸载方法，详细分析了异构任务的卸载成本，根据任务的属性特征动态调整时延和能耗权重，利用卷积神经网络改进了A3C网络结构。实验表明CA3C算法在降低异构任务卸载成本方面具有更好的性能，并且算法的收敛性更好，收敛速度更快，适用于动态时变的卫星网络环境。然而，如同大多数基于深度强化学习的解决方案，本文提出的方法有效性在一定程度上依赖于训练阶段的系统参数分布，当前模型的泛化能力有待进一步验证。当系统参数发生显著或结构性变化时，理论上可以通过增量学习或重新训练模型以改进模型性能。提升算法的泛化与自适应能力，使模型能够快速适应未经验证的系统参数与动态环境将是下一步工作的研究方向。

参考文献：

[1] Ranaweera P, Jurcut A D, Liyanage M. Survey on multi-access edge computing security and privacy[J]. IEEE Communications Surveys & Tutorials, 2021, 23(2): 1078-1124.

[2] Liu Y Q, Peng M G, Shou G C, et al. Toward edge intelligence: multiaccess edge computing for 5G and Internet of Things[J]. IEEE Internet of Things Journal, 2020, 7(8): 6722-6747.

[3] Yang J Y, Zhang Y J, Xiao Z Y, et al. Joint access selection and computation offloading in LEO ubiquitous edge computing networks: an alternating DRL-based approach[J]. IEEE Transactions on Cognitive Communications and Networking, 2025, 11 (3): 1870-1886.

[4] Ji Z, Wu S, Jiang C X. Cooperative multi-agent deep reinforcement learning for computation offloading in digital twin satellite edge networks[J]. IEEE Journal on Selected Areas in Communications, 2023, 41(11): 3414-3429.

[5] Zhang C Z, Yang J. An energy-efficient collaborative offloading scheme with heterogeneous tasks for satellite edge computing[J]. IEEE Transactions on Network Science and Engineering, 2024, 11(6): 6396-6407.

[6] Lai Z Q, Li H W, Wu Q, et al. Futuristic 6G pervasive on-demand services: integrating space edge computing with terrestrial networks[J]. IEEE Vehicular Technology Magazine, 2023, 18(1): 80-90.

[7] Xie R C, Tang Q Q, Wang Q N, et al. Satellite-terrestrial integrated edge computing networks: architecture, challenges, and open issues[J]. IEEE Network, 2020, 34(3): 224-231.

[8] Tang Q Q, Fei Z S, Li B, et al. Computation offloading in LEO satellite networks with hybrid cloud and edge computing[J]. IEEE Internet of Things Journal, 2021, 8(11): 9164-9176.

[9] Zhang S H, Cui G F, Long Y T, et al. Joint computing and communication resource allocation for satellite communication networks with edge computing[J]. China Communications, 2021, 18(7): 236-252.

[10] Song Z Y, Hao Y Y, Liu Y W, et al. Energy-efficient multiaccess edge computing for terrestrial-satellite Internet of Things [J]. IEEE Internet of Things Journal, 2021, 8(18): 14202-14218.

[11] Gao X Q, Liu R K, Kaushik A. Virtual network function placement in satellite edge computing with a potential game approach[J]. IEEE Transactions on Network and Service Management, 2022, 19(2): 1243-1259.

[12] 方海, 高媛, 赵扬, 等. 卫星边缘计算中任务卸载与资源分配联合优化算法[J]. 小型微型计算机系统, 2023, 44(6): 1214-1219.
Fang H, Gao Y, Zhao Y, et al. Joint optimization of task offloading and resource allocation in satellite edge computing[J]. Journal of Chinese Computer Systems, 2023, 44(6): 1214-1219.

[13] Zhang H Y, Liu R K, Kaushik A, et al. Satellite edge computing with collaborative computation offloading: an intelligent deep deterministic policy gradient approach[J]. IEEE Internet of Things Journal, 2023, 10(10): 9092-9107.

[14] Huang C, Chen G J, Xiao P, et al. Joint offloading and resource allocation for hybrid cloud and edge computing in SAGINs: a decision assisted hybrid action space deep reinforcement learning approach[J]. IEEE Journal on Selected Areas in Communications, 2024, 42(5): 1029-1043.

[15] 杨桂松, 陶挺, 何杏宇, 等. 卫星物联网中联合资源分配的边缘计算卸载策略[J]. 小型微型计算机系统, 2024, 45(10): 2544-2550.
Yang G S, Tao T, He X Y, et al. Strategy of joint resource allocation and computing edge offloading in satellite Internet of Things[J]. Journal of Chinese Computer Systems, 2024, 45 (10): 2544-2550.

[16] Wang Q T, Chen S Y, Yang C L, et al. Energy-efficient task split and resource allocation in LEO-satellite-assisted IoT network[J]. IEEE Internet of Things Journal, 2024, 11(21): 34519-34527.

[17] Zhou J, Liang J W, Zhao L, et al. Latency-energy efficient task offloading in the satellite network-assisted edge computing *via* deep reinforcement learning[J]. IEEE Transactions on Mobile Computing, 2025, 24(4): 2644-2659.

[18] Zhang H Y, Zhao H B, Liu R K, et al. Dynamic user association and computation offloading in satellite edge computing networks *via* deep reinforcement learning[J]. IEEE Transactions on Green Communications and Networking, 2024, 8(4): 1888-1901

[19] Zhou J, Zhao Y C, Zhao L, et al. Adaptive task offloading with spatiotemporal load awareness in satellite edge computing[J]. IEEE Transactions on Network Science and Engineering, 2024, 11(6): 5311-5322.

[20] Zhang H Y, Zhao H B, Liu R K, et al. Collaborative task offloading optimization for satellite mobile edge computing using multi-agent deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2024, 73(10): 15483-15498.

[21] Qin Z Y, Yao H P, Mai T L, et al. Multi-agent reinforcement learning aided computation offloading in aerial computing for the Internet-of-things[J]. IEEE Transactions on Services Computing, 2023, 16(3): 1976-1986.

[22] Deng X H, Sun Z H, Li D, et al. User-centric computation offloading for edge computing[J]. IEEE Internet of Things Journal, 2021, 8(16): 12559-12568.

[23] Xiao T T, Chen C, Dong M X, et al. Multi-agent reinforcement learning-based trading decision-making in platooning-assisted vehicular networks[J]. IEEE/ACM Transactions on Networking, 2024, 32(3): 2143-2158.

[24] Li Q, Wang S G, Ma X, et al. Battery-aware energy optimiza-

tion for satellite edge computing[J]. IEEE Transactions on Services Computing, 2024, 17(2): 437-451.

[25] Lv W K, Yang P F, Ding Y Q, et al. Energy-efficient and QoS-aware computation offloading in GEO/LEO hybrid satellite networks[J]. Remote Sensing, 2023, 15(13): 3299.

[26] Dou J M, Wang X J, Liu Z, et al. MAPIRL: a hyperbolic tangent-enforced physical-informed RL for multi-IESs optimal dispatch[J]. IEEE Transactions on Industry Applications, 2025, 61(2): 2549-2564.

[27] Zou J F, Hao T B, Yu C, et al. A3C-DO: a regional resource scheduling framework based on deep reinforcement learning in edge scenario[J]. IEEE Transactions on Computers, 2021, 70 (2): 228-239.

[28] 刘治国, 董效奇, 汪林, 等. 面向ET-DQN的卫星网络任务部署算法研究[J]. 小型微型计算机系统, 2024, 45(2): 418-424.
Liu Z G, Dong X Q, Wang L, et al. Research on satellite network task deployment algorithm based on ET-DQN[J]. Journal of Chinese Computer Systems, 2024, 45(2): 418-424.

[29] Jia M, Zhang L, Wu J, et al. Joint computing and communication resource allocation for edge computing towards Huge LEO networks[J]. China Communications, 2022, 19(8): 73-84.

[30] Zhao Z L, Feng M J, Ke C X, et al. Federated deep recurrent Q-learning for task partitioning and resource allocation in satellite mobile-edge-computing-assisted industrial IoT[J]. IEEE Internet of Things Journal, 2024, 11(15): 26444-26458.

[作者简介]

刘治国（1974− ），男，博士，大连大学信息工程学院教授、硕士生导师，主要研究方向为网络体系结构、协议技术、网络管理。

金晓勇（1997− ），男，大连大学信息工程学院硕士生，主要研究方向为卫星网络、边缘计算。；

汪林（1979− ），女，博士，大连大学环境与化学工程学院副教授、硕士生导师，主要研究方向为环境分析。；

潘成胜（1962− ），男，博士，南京信息工程大学电子与信息工程学院教授、博士生导师，主要研究方向为卫星网络通信研究。