控制与决策
Control and Decision

Actor-Critic框架下一种基于改进DDPG的多智能体强化学习算法

陈亮， 梁宸， 张景异， 刘韵婷

引用本文:
陈亮, 梁宸, 张景异, 等. Actor-Critic框架下一种基于改进DDPG的多智能体强化学习算法[J]. 控制与决策, 2021, 36(1): 75-82.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2019.0787

Actor-Critic框架下一种基于改进DDPG的多智能体强化学习算法

陈 亮, 梁 宸, 张景异, 刘韵婷†

(沈阳理工大学 自动化与电气工程学院，沈阳 110159)

摘 要: 现实世界的人工智能应用通常需要多个agent协同工作,人工agent之间有效的沟通和协调是迈向通用人工智能不可或缺的一步. 以自主开发的警员训练虚拟环境为测试场景,设定任务需要多个不同兵种agent小队互相协作或对抗完成. 为保证沟通方式有效且可扩展,提出一种混合DDPG(Mi-DDPG)算法. 首先,在Actor网络加入双向循环神经网络(BRNN)作为同兵种agent信息交流层;然后,在Critic网络加入其他兵种agent信息来学习多agent协同策略. 另外,为了缓解训练压力,采用集中训练,分散执行的框架,同时对Critic网络里的Q函数进行模块化处理. 实验中,在不同的场景下用Mi-DDPG算法与其他算法进行对比,Mi-DDPG在收敛速度和任务完成度方面有明显提高,具有在现实世界应用的潜在价值.

关键词: 强化学习；深度学习；多智能体；RNN；DDPG；Actor-Critic

中图分类号: TP181 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2019.0787 **开放科学(资源服务)标识码(OSID):**

引用格式: 陈亮,梁宸,张景异,等. Actor-Critic框架下一种基于改进DDPG的多智能体强化学习算法[J]. 控制与决策, 2021, 36(1): 75-82.

A multi-agent reinforcement learning algorithm based on improved DDPG in Actor-Critic framework

CHEN Liang, LIANG Chen, ZHANG Jing-yi, LIU Yun-ting†

(College of Automation and Electrical Engineering，Shenyang Ligong University，Shenyang 110159，China)

Abstract: Real-world artificial intelligence (AI) applications often require multiple agents to work together, and effective communication and coordination between artificial agents is an indispensable step toward universal artificial intelligence. This paper takes the self-developed virtual environment for police training as a test scenario. Setting tasks requires multiple different service agent teams to cooperate or fight against each other. In order to ensure that the communication method is effective and scalable, this paper proposes the mixed deep deterministic policy gradient (Mi-DDPG) algorithm. Firstly, the bidirectional recurrent neural networks (BRNN) is added to the Actor network as the information exchange layer of the same type of agent, and then the other agent information is added to the Critic network to learn the multi-agent cooperation strategy. In addition, in order to alleviate the training pressure, the centralized training and distributed execution framework are adopted, and the Q function in the Critic network is modularized. In the experiment, the Mi-DDPG algorithm is compared with other algorithms in different scenarios, which shows its most advanced performance and potential value in real-world.

Keywords: reinforcement learning；deep learning；multi-agent；RNN；DDPG；Actor-Critic

0 引 言

强化学习(reinforcement learning, RL)是一种通过奖赏和惩罚引导学习的机器学习(machine learning, ML)方法,是机器学习的一个重要分支,如图1所示,其要研究的问题是智能体(agent)如何在环境中学到一定的策略(policy),使得长期的奖励(reward)最大[1].

近10年来,人们在人工智能领域取得了巨大的进步,单个人工智能单元通过使用强化学习方法在许多策略性问题上都取得了突破性进展,包括在游戏中

收稿日期: 2019-06-04；**修回日期:** 2019-08-07.

基金项目: 国家重点研发计划项目(2017YFC0821004, 2017YFC0821001)；辽宁省自然科学基金项目(20170540788)；辽宁省教育厅基本科研项目(LG201707).

责任编委: 张维海.

†通讯作者. E-mail: 71019976@qq.com.

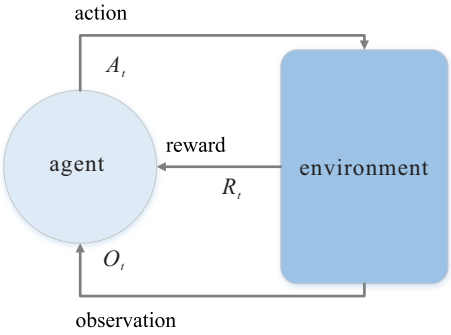


图1 强化学习示意

学习人类操作策略[2]或是控制机器人进行机器博弈[3].然而,真正的人类智能包含了群体的智慧,很多重要的应用场景牵涉到多个agent之间的交互,在这种交互过程中,会演化出新的联合行为,问题也会变得更加复杂[4].比较典型的多agent场景包括多机器人协同控制[5]、多玩家游戏[6]、社会金融困境分析等,利用强化学习来实现多agent协作目前已经成为多agent系统与机器学习领域的研究热点.

很早之前,强化学习作为一种重要手段应用于多agent系统,以获得学习环境中agent交互和协作的最佳策略[7].本文利用自主开发的警员训练虚拟环境来探索多个agent之间如何学习最优联合行为,在这个环境中,不同兵种在不同地形条件下完成合作目标或者在对抗中摧毁对方作战队伍.传统的强化学习方法,如基于值的方法Q-learning[8]、基于概率的方法Policy Gradient[9]等在面对这种大规模多agent系统的学习面临着维数灾难,即当环境较为复杂或者任务较为困难时,agent的状态空间过大,会导致需要学习的参数以及所需的存储空间急速增长,强化学习难以取得理想的效果.

多agent强化学习中的关键问题之一是研究agent之间的通信协议.近几年,随着深度强化学习的发展,一些嵌套多agent通讯的模型被应用到多agent强化学习算法研究中.Sukhbaatar等[10]提出的CommNet适用于完全可观察环境.在CommNet中,用一个单独的网络将信息直接传递给层之间的agent模块.然而,通信网络是完全对称的并且嵌入到原有网络中,因此缺乏处理异构agent(heterogeneous agent)之间的通讯能力.Foerster等[11]提出的可区分的智能体间学习(differentiable inter-agent learning, DIAL)则专为部分可观察环境中的联合行动学习者设计.与CommNet不同,DIAL中每个agent都包含一个循环神经网络,输出个体agent的Q值和每个时间步传送的消息,然后将生成的消息转移到其他agent,作为下一个时间步中其他agent的输入.这种通讯方法可以让梯度在agent之间传递,以缓解agent相互作用时常见的不稳定问题.但是,这种方法无法良好地适用于复杂环境中.

为了解决上述问题,阿里巴巴团队提出的在谷歌的pysc2多agent环境[12]下的新算法BiCNet[6],在以上两种方法的基础上通过用双向循环神经网络(bidirectional recurrent neural networks, BRNN)[13]来连接每个同质agent(homogeneous agent),通信发生在潜在空间中,以便高层信息可以在agent之间传递,同时异构agent可以使用不同的参数和输出动作集创建.在pysc2这样的复杂环境下,BiCNet显示了先进的性能.但是BiCNet将学习任务制定为零和博弈,考虑的重点在于多agent竞争环境中的微观管理任务,在纯合作无竞争环境中的表现尚不明确.openAI团队提出的MADDPG[14]在一些简单的案例中采用了集中学习和分散执行的框架,在训练时每个agent都会获取到额外信息,可以在各种混合环境下发挥作用.

基于上述分析,本文提出一种混合环境下的多agent协同作战策略算法Mi-DDPG(mixed deep deterministic policy gradient, Mi-DDPG),该算法假设同质agent之间固有关联性,让同质agent在Actor层进行通信;而对于异构agent则采取一种整体的集中训练、分散执行的框架,使Critic层增加其他agent的policy额外信息,agent执行时只能接触到本地信息(包含同兵种通信的信息),最后用模块化Q函数的方法缓解训练压力,加快收敛.经实验验证,Mi-DDPG表现出其优越的性能,可以自动学习协调各agent的最佳策略,尤其是在agent数量增长后表现出优于其他算法的稳定性和收敛速度,并且显示了其在大规模多agent任务中实际应用的潜在价值.

1 理论基础

1.1 马尔可夫博弈

马尔可夫性[15]是指系统的下一状态s_{t+1}仅与当前状态s_t有关,与之前的状态无关.简单来说,当前状态s_t蕴含了所有相关的历史信息$s_1,\ldots,s_t$,若当前状态已知,历史信息将会无用.强化学习的马尔可夫决策过程是将状态作为输入并生成输出动作,其中所有的状态都满足马尔可夫性.马尔可夫博弈则是马尔可夫决策过程在多agent条件下的扩展(MDPs)[7].在多agent环境中N个agent的马尔可夫博弈由以下几个元素构成:

1) 状态S用来描述所有agent的所有可能配置;

2) 每个智能体的动作$A_1,\ldots,A_N$;

3) 每个智能体的观测值$O_1,\dots,O_N$.

在马尔科夫博弈过程中,初始状态S由一个随机分布确定,策略π是从状态到动作的映射概率. 每个agent会使用随机策略π_{θ_i}来选取动作,然后根据状态转移函数得到下一个状态,最后根据当前状态及动作获得各自奖励以及观测值O_i. 每个agent的目的都是为了最大化自己的最终预期奖励,即

$$R_i=\sum_{t=0}^{T}\gamma^t r_i^t, \tag{1}$$

其中γ为折扣因子,表明agent越快得到最佳策略,其得到的奖励也会越高.

1.2 Q-Learning和DQN

Q-learning和DQN[16]是典型的基于值的强化学习算法,在与环境的交互过程中利用值函数学习最优策略. Q-learning的动作值函数(Q函数)为(基于策略π)

$$Q^{\pi}(s,a)=\mathrm{E}_{s'}[r(s,a)+\gamma\mathrm{E}_{a'\pi}[Q^{\pi}(s',a')]]. \tag{2}$$

其中:$r(s,a)$是状态s采取动作a后获得的立即奖励;γ为衰减率,用来计算从状态s'一直到回合结束的累计奖励.

DQN在Q-learning基础上加入神经网络,采用双网络结构,通过最小化损失来更新参数,即

$$\begin{cases} y=r+\gamma\max_{a'}\bar{Q}^{*}(s',a'),\\ L(\theta)=\mathrm{E}_{s,a,r,s'}[(Q^{*}(s,a|\theta)-y)^2]. \end{cases} \tag{3}$$

其中$\bar{Q}$是Q目标网络(target-net),而θ是Q估计网络(eval-net)的参数,每隔一定的时间步,Q目标网络会用最新的θ来更新自身参数,这种“冻结”网络的目的是切断数据相关性,用以稳定训练过程.

如果将这种基于值的强化学习方法直接用在多agent环境中,则每个agent都会随着训练的进展独立更新自身的策略.因此,以任意智能体的角度来看,环境都会变得不稳定(其他agent也作为环境的一部分),这使得该方法收敛所需的马尔科夫假设不再成立.

1.3 Policy Gradient和Actor-Critic

Policy Gradient是一种基于策略的算法,通过调整策略的参数θ沿目标函数梯度方向前进来最大化目标. 目标函数定义为奖励的期望值,即

$$J(\theta)=\mathrm{E}_{sp^{\pi},a\pi_{\theta}}[R]. \tag{4}$$

利用之前提到的Q函数,可以用梯度下降法更新策略参数,如下所示:

$$\nabla J(\theta)=\mathrm{E}_{sp^{\pi},a\pi_{\theta}}[\nabla_{\theta}\log\pi_{\theta}(a|s)Q^{\pi}(s,a)]. \tag{5}$$

其中:$\pi_{\theta}(a|s)$为在状态s下采取动作a的发生概率;p^{μ}为状态分布;$Q^{\pi}(s,a)$为策略π下状态s采取动作a一直到回合结束的总回报,$Q^{\pi}(s,a)$越大,梯度下降的越快,选择该动作的机率就越大. 简单来说,当策略$\pi_{\theta}(a|s)$沿梯度方向更新参数时,Q函数会作为一个“监督者”来把控更新的幅度.

基于策略的算法由于是沿梯度方向更新,往往会偏离预期奖励,产生很高的方差,可以通过加入基线的方法缓解,即

$$\nabla J(\theta)=\mathrm{E}_{sp^{\pi},a\pi\theta}[\nabla_{\theta}(Q^{\pi}(s,a)-b(s))\log\pi_{\theta}(a|s)]. \tag{6}$$

其中$b(s)$为基线函数,当$b(s)=\mathrm{E}_a(Q^{\pi}(s,a))=V(s)$(状态值函数)时,方差最小.

如果将基于值的算法(如DQN)和Policy Gradient算法相结合,采用时间差分(temporal difference, TD)[1]的方法来更新,则算法的参数便由回合更新变成了单步更新,此时被称为“Critic”,并演化出各种Actor-Critic算法[1].

1.4 DDPG算法

DDPG(deep deterministic policy gradient)[17]算法是Actor-Critic框架与DQN算法的融合.

DDPG中的第1个D(Deep)指的是使用DQN算法中的经验重放方法[16]和双网络结构来切断相关性以提高神经网络的学习效率. 第2个D(Deterministic)指的是Actor网络采用确定性策略输出具体的动作,而不是动作的概率,这样使学习可以在连续的动作空间中进行.

在Actor和Critic中都有目标网络(target-net)和估计网络(eval-net),在训练过程中只需估计网络的参数,而目标网络的参数由估计网络每隔一定时间步直接复制. Critic根据下式所示的损失函数进行网络学习:

$$\begin{cases} y=r+\gamma\max_{a'}\bar{Q}^{*}(s',a'),\\ L(\theta)=\mathrm{E}_{s,a,r,s'}[(Q^{*}(s,a|\theta)-y)^2]. \end{cases} \tag{7}$$

虽然式(7)看上去与1.2节提到的DQN相同,但含义却并不一样,其中$Q^{*}(s,a|\theta)$是根据Critic估计网络得到的,a是Actor估计网络传过来的动作. 而y是目标网络Q值,因为采用确定性策略,计算目标Q值时,也不再使用贪心算法来选择动作a',而是采用Actor目标网络传过来的a'. 总体而言,Critic估计网络的训练还是基于目标Q值和估计Q值的平方损失,估计Q值根据当前的状态s和Actor估计网络输出的动作a输入Critic估计网络得到,而目标Q值根据奖励r,以及将下一时刻的状态s'和Actor目标网络得到

的动作a',输入到Critic目标网络而得到的Q值乘以折扣因子加和得到.

Actor网络基于确定性策略$\mu_\theta : S \to A$,并根据下式进行参数更新:

$$\nabla J(\theta) = \mathrm{E}_{sD}[\nabla_\theta \mu_\theta(a|s) \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)}]. \quad (8)$$

由于这个定理依赖于$\nabla_a Q^\mu(s,a)$,它需要动作空间A连续. 简单来说,假如对同一个状态,输出了两个不同的动作a_1和a_2,根据状态估计网络得到了两个反馈的Q_1和Q_2值,若$Q_1 > Q_2$,则执行动作a_1可以获得更多的reward. 策略梯度的思想是,增加a_1的概率,降低a_2的概率,以获得更大的Q值.

DDPG在近年来是一个非常受欢迎的非策略性的Actor-Critic方法,它是确定性策略梯度算法的深层变体,然而确定性Actor网络和Q函数之间的相互作用通常使得DDPG难以达到稳定,超参数的设置也变得困难,因此很难直接将DDPG扩展到复杂的高维多智能体环境.

2 Mi-DDPG

2.1 模型结构

本文所研究的算法基于一个允许任意数量agent的灵活框架,认为所有agent共享当前环境的相同状态空间S,并且每个兵种具有相同的动作空间A,因此Actor网络在同兵种内部加了双向循环神经网络BRNN通信层. 当不同兵种agent交互时,一定程度上代表了自己兵种内的所有agent,而同兵种内部不进行这种整体的交互,所以Critic网络的输入默认忽略了同兵种信息. 算法的模型结构如图2所示.

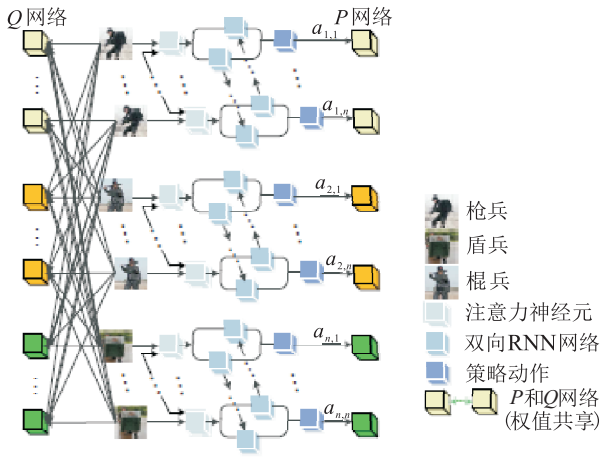


图2 Mi-DDPG模型结构

图2是模型的整体结构,该结构要解决的问题是不同兵种多智能体如何进行通信交流,在实验中并不局限于图中这3类兵种,会根据任务要求增加消防兵、排爆兵、歹徒等所需兵种.

图2中Actor网络增加了带注意力神经元的BRNN网络用于同兵种agent通讯,在实验过程中,由于实验环境的高度可变性,agent在采取行为前交换的信息并不具备任何具体含义,Mi-DDPG的BRNN通讯层只关注双向通讯,与通讯无关的其他含义被忽略. 可以这样认为,在每一回合中,agent获取其他agent观察到的输入的总和,并根据总和与他们预测(动作输出)之间的差值获得奖励.

而Critic网络采取如图3所示的集中训练、分散执行的框架,拥有两种输入形式,在训练时每个agent的Q网络输入全局信息进行训练(包含其他agent信息),而在执行时P网络通过与Q网络共享权值,只接收本地信息来更新策略. P网络和Q网络均是一个每层64个神经元的多层感知机(multilayer perceptron, MLP).

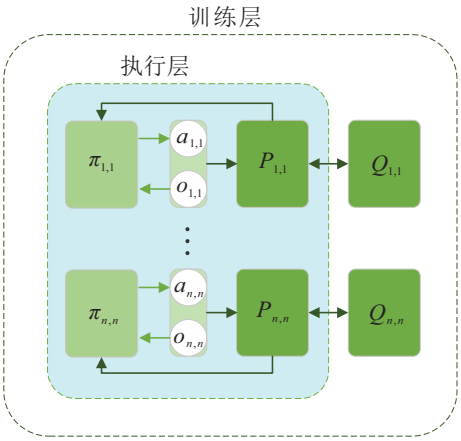


图3 分散执行,集中训练框架

2.2 带注意力神经元的双向BRNN网络

本文使用BRNN网络来实现同质agent内部通讯,同时,为了使agent对当前信息的重要性进行区分,引入注意力机制[18],将注意力集中在所给出信息的一部分子集上. 网络结构如图4所示.

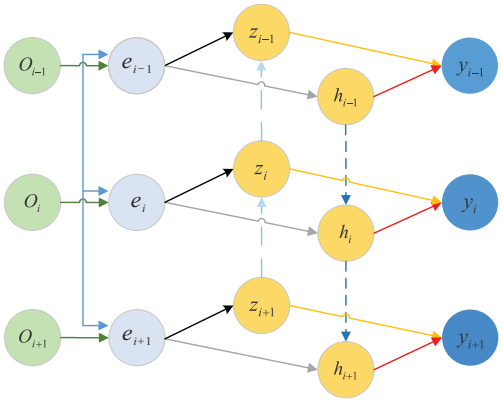


图4 双向BRNN网络结构

图4网络中加入的注意力单元对输入向量的不同元素赋予不同的注意力权重,并且所有agent的注

意力单元相连以便一起进行参数更新. 具体如下:

$$\bar{O}_i = e_i O_i, \tag{9}$$

$$h_i = \sigma(W_{h\bar{O}}\bar{O}_i + W_{hh}h_{i-1} + b_h), \tag{10}$$

$$z_i = \sigma(W_{z\bar{O}}\bar{O}_i + W_{zz}z_{i+1} + b_z), \tag{11}$$

$$y_i = \text{soft}\max(W_{yh}h_i + W_{yz}z_i + b_y). \tag{12}$$

其中: O_i、e_i、z_i、h_i、y_i 分别代表输入、注意力单元、后向RNN隐状态(输出)、前向RNN隐状态(输出)、输出; W 为各类输入循环权重; b 为偏置项; σ 为sigmoid函数,用于控制单元流经的权重,范围在 0、1之间. 可以看出, $\bar{O}_i$ 代替原始输入 O_i 参与到BRNN的循环运算中,并且通过这种双向的结构, agent_i 可以获取前后其他同质agent的信息并进行通讯.

2.3　损失函数

在这项工作中,本文提出一种既能用于合作环境,也能用于竞争环境的通用多智能体学习算法,并且该算法属于无模型(model free)算法,不需要为环境建立动力学模型.

首先,定义policy是以 $\theta = \{\theta_1, \ldots, \theta_N\}$ 为参数的 N 个智能体之间的博弈,将所有智能体策略的集合设为 $\pi = \{\pi_1, \ldots, \pi_N\}$,则期望收益的梯度公式为

$$\nabla_{\theta_i} J(\theta_i) =$$

$$\mathrm{E}_{sp^{\mu}, a_i \pi_i}[\nabla_{\theta_i} \log \pi_i(a_i|o_i) Q_i^{\pi}(s, a_1, \ldots, a_N)]. \tag{13}$$

其中 $Q_i^{\pi}(S, a_1, \ldots, a_N)$ 为一个集中的动作值函数,将所有agent的动作 $a_1, \ldots, a_N$ 加上一些状态信息 S 作为输入,然后输出 agent_i 的 Q 值. 简单来说, S 可以包含所有的观测值, $S = \{O_1, \ldots, O_N\}$,同时也可以定义附加的状态信息. 由于每个 Q_i^{π} 是分开学习的,智能体可以有任意的奖励方式,包括在对抗环境中相互竞争的奖励,本文利用模块化 Q 函数的方法对 Q 函数的输入做简化处理.

将上述想法扩展到确定性策略,令 N 个策略表示为 μ_{θ_i}(参数为 θ_i,缩写为 μ_i),则

$$\nabla_{\theta_i} J(\mu_i) =$$

$$\mathrm{E}_{s,aD}[\nabla_{\theta_i} \mu_i(a_i|o_i) \nabla_{a_i} Q_i^{\mu}(s, a_1, \ldots, a_N)|_{a_i = \mu_i(o_i)}]. \tag{14}$$

经验重放缓冲区 D 包含元组 $(S, S', a_1, \ldots, a_N, r_1, \ldots, r_N)$,记录了所有智能体的经验. 集中的动作值函数 Q_i^{μ}(参数为 ξ_i,与Actor网络区分)按下式更新:

$$\begin{cases} y = r_i(s,a) + \lambda \max\limits_{\theta} Q_i^{u'}(s', a_1', \ldots, a_N')|_{a_j' = u_j'(o_j)}, \\ L(\xi_i) = \mathrm{E}_{s,a,r,s'}[(Q_i^{u}(s, a_1, \ldots, a_N) - y)^2]. \end{cases} \tag{15}$$

其中 $\mu' = \{\mu'_{\xi_1}, \ldots, \mu'_{\xi_N}\}$ 是具有延迟参数 ξ_i' 的目标策略集合. 实验中发现具有确定性策略的集合Aritic运行状况良好.

2.4　模块化 Q 函数

在很多问题中,智能体之间只在部分特定的状态下才进行交互,例如两名排爆警员在进入同一片雷区时才考虑其联合动作,此时避免碰撞和通力合作是提高效率的必须条件. Mi-DDPG算法另一个思想就是对于距离较远且关联性不高的agent不考虑其联合动作,这样可以避免 Q 函数的输入空间随智能体个数 N 的增加呈线性增长. 算法一开始拟采用聚类方法对agent进行聚类并以此决定 Q 函数的输入,然而在模型训练时聚类算法在每一个回合都消耗了大量的时间和计算资源,因此最后改用基于欧氏距离的邻域内agent交互来实现模块化 Q 函数,如图5所示.

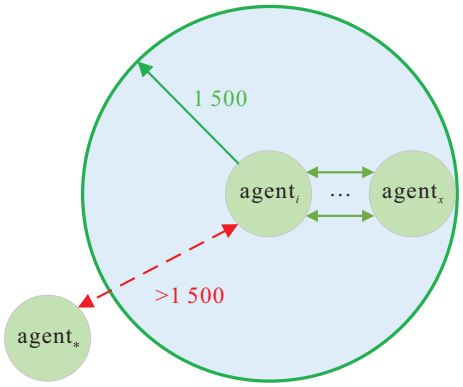


图 5　双向BRNN网络结构

图 5 中,圈内的 agent_i-agent_x 在 agent_i 邻域内,1 500是距离值. 应用模块化 Q 函数方法后,损失函数的定义由式(14)、(15)改为

$$\nabla_{\theta_i} J(\mu_i) =$$

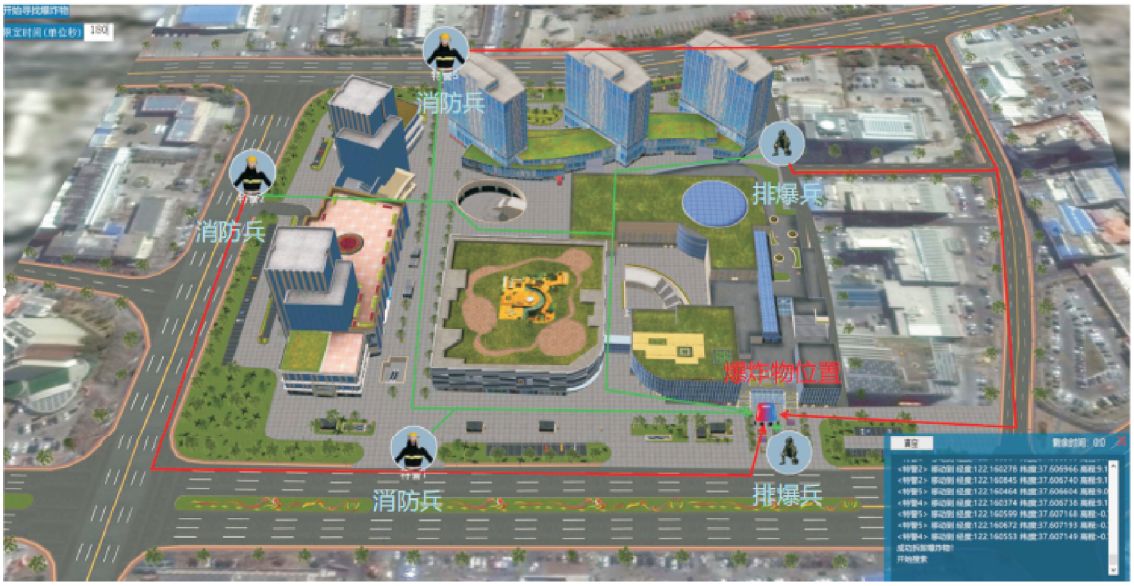
$$\mathrm{E}_{s,aD}[\nabla_{\theta_i} \mu_i(a_i|o_i) \nabla_{a_i} Q_i^{\mu}(s, a_i, \ldots, a_x)|_{a_i = \mu_i(o_i)}]; \tag{16}$$

$$\begin{cases} y = r_i(s,a) + \lambda \max\limits_{\theta} Q_i^{u'}(s', a_i', \ldots, a_x')|_{a_j' = u_j'(o_j)}, \\ L(\xi_i) = \mathrm{E}_{s,a,r,s'}[(Q_i^{u}(s, a_i, \ldots, a_x) - y)^2]. \end{cases} \tag{17}$$

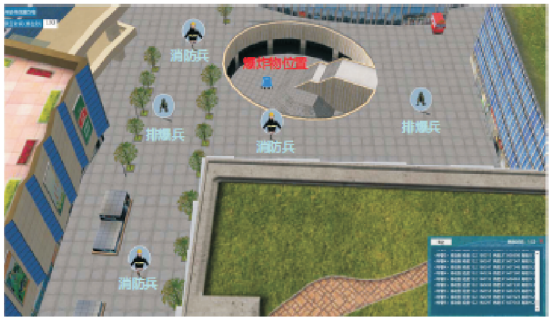
3　实验及结果分析

3.1　实验环境

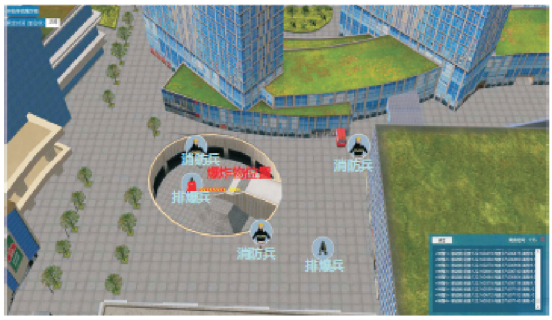
本文采用一个自主研发的警员训练系统作为训练平台,该平台仿照openAI GYM[14]的强化学习环境进行了深化改进,重新定义了三维环境和一些通用API,并且大部分经典算法都可以在该环境中进行测试. 与其他实验环境相比,使用自主研发的实验平台可以更好地适用于课题所关注的多智能体决策问题



(a) 各agent得到一个随机的初始位置



(b) 异构agent开始交流



(c) 异构agent合作排爆

图6 实验环境-排爆任务(合作)

(在遇到突发事件时指挥层如何制定最好的策略让不同警组协同配合处理事件),并且在奖励函数定义以及API定义等方面也更加自由.实验环境由N个agent和L个地标组成,每个agent都定义了一个兵种属性,不同的兵种有不同的动作空间,而同兵种的动作空间相同.如排爆兵负责排爆,消防兵负责协助排爆和疏散人群,哨兵负责传递消息,枪、盾、棍兵和歹徒属于战斗兵种模拟实战对抗,这些agent在具有连续空间和离散时间的三维空间中执行不同任务以获得最大奖励.本文为不同环境定义了不同的任务类型,如合作环境下的排爆任务,竞争环境下的对抗任务,混合环境下的营救任务等.以排爆任务为例,将训练过程可视化得到图6所示效果.

由图6可以看出,实验定义的排爆任务可以简单认为是一种路径规划问题,旨在最短时间内找到爆破物并排除,回合开始时,图6(a)中每个agent获得一个随机的初始位置并根据当前模型参数决定行走路线寻找爆破物,当图6(b)中这些agent互相靠近后,异构agent的Critic网络开始交流,最后在图6(c)中,可以看到两个异构agen协同合作完成任务.

3.2 模块化Q方法阈值的选择

实验过程中,首先确定模块化Q方法邻域的阈值,以对抗任务为例,在训练过程中,适应性动量估计算法(adaptive moment estimation, Adam)被设置成优化器,学习率设为0.002,其他参数由默认值设置.每个回合(episode)的最大步数设置为800步.在该模型参数条件下,选择不同的邻域阈值(以当前agent位置为圆心),通过观察回合平均奖励和收敛所需时间,确定最佳阈值.实验结果如图7所示.

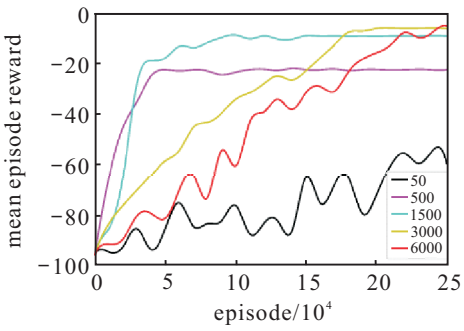


图7 在不同阈值下agent获得的平均奖励

收敛一般指的是损失函数趋于稳定,在本文中,当回合平均奖励趋于稳定时,可以看作环境中的

agent 已经学到了基于当前算法的最佳策略,无法获得更高奖励,而在实验中也发现当回合平均奖励趋于稳定时,损失函数收敛. 由图 7 可以看出,当阈值设置过小时,异构 agent 之间交流变得困难,算法难以收敛,当阈值设置过大时,算法收敛时间延长. 根据图 7 的结果,设置 1 500(单位:坐标点)为模块化 Q 的阈值. 本文在测试其他任务时使用学习的最优值来修正这个参数.

3.3　在不同难度下的性能测试

为测试算法性能,本文在不同设置的训练环境下对 Mi-DDPG 算法进行实验. 如表 1 所示,任务实现难度通过改变兵种类型数量和每类兵种数量来调整. 考虑到训练资源和实际应用价值,其中最高难度设定为难度 5(环境中存在 8 种不同兵种且每个兵种包含 20 个 agent).

表 1　不同难度下模型收敛速度

难度	兵种类型	数量	是否收敛	达到收敛速度 / 步	收敛后的任务完成度 / %
1	2	5	是	29 K	98.3
2	2	10	是	47 K	93.6
3	4	10	是	121 K	89.2
4	4	20	是	163 K	85.7
5	8	20	是	276 K	83.8

表 1 中任务完成度表示所有 agent 的回合总得分占预设的回合最高总得分的百分比. 由表 1 可以看出,随着 agent 类型和数量的增多,算法的收敛时间延长,但是即便是在最高难度下,算法依然得到了收敛并取得了较高的任务完成度. 这表示在大规模混合环境中,算法通过 agent 之间的交互学习到了最佳策略,最终趋于稳定.

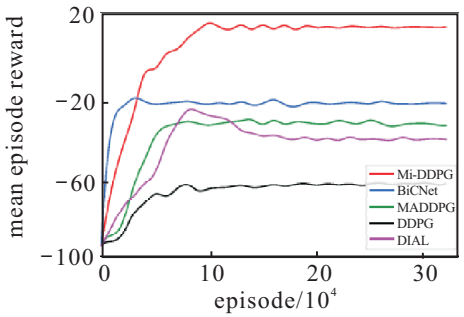
3.4　对比实验

为进一步验证 Mi-DDPG 在不同环境中学到的策略的质量,本文将 Mi-DDPG 算法与其他算法进行对比,训练各个算法的模型直到收敛. 实验结果如图 8 所示. 图 8 在不同的难度和环境下通过迭代训练 350 000 次对各种算法的性能进行对比.

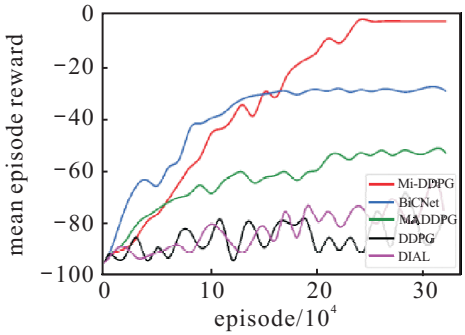
当难度相同环境不同时,分别对比图 8(a) 和图 8(c)、图 8(b) 和图 8(d),可以看出相比纯对抗环境,混合环境下 BiCNet 和 DDPG 变得难以收敛,MADDPG 和 DIAL 收敛速度变得缓慢,而 Mi-DDPG 的收敛速度几乎没有受到影响.

当环境相同难度不同时,分别对比图 8(a) 和图 8(b)、图 8(c) 和图 8(d),可以看出当难度增加后每种算法的收敛都变得困难,同时获得的回合平均奖励也开始降低,其中 DDPG 和 DIAL 受到的影响尤为明显,而 Mi-DDPG 回合平均奖励降低的幅度最小.

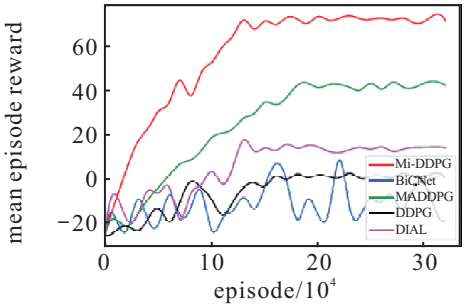
综上所述,在小规模多智能体对抗环境中 Mi-DDPG 和 BiCNet 的收敛速度接近并优于其他算法,随着 agent 的增加以及纯合作环境的加入,除了 Mi-DDPG 外最终只有 MADDPG 能够勉强达到收敛,并且在每种度量的评估中,Mi-DDPG 最终都获得了最高的回合平均奖励,结合 3.3 节,这表明 Mi-DDPG 也取得了最高的任务完成度,学习到了优于其他算法的最佳策略.



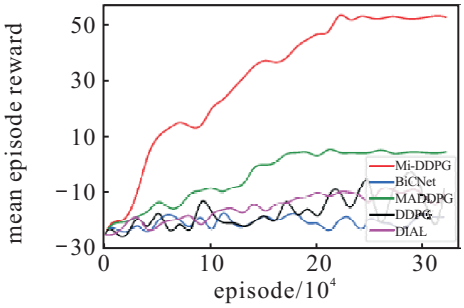
(a)　难度 3 对抗环境下 agent 的平均奖励比较



(b)　难度 5 对抗环境下 agent 的平均奖励比较



(c)　难度 3 混合环境下 agent 的平均奖励比较



(d)　难度 5 混合环境下 agent 的平均奖励比较

图 8　Mi-DDPG 和其他算法在各种环境下的性能对比

4 结 论

本文以DDPG算法为基础,分别对算法的Actor和Critic网络进行改进,通过在Actor网络加入同质agent的BRNN通讯层,在Critic网络加入分散执行、集中训练框架和模块化Q函数方法,提出一种新算法Mi-DDPG,并在本文提到的自主开发环境中将该算法与其他典型算法进行了对比实验. 理论上该算法可以适用于其他类似的环境和问题(涉及多智能体合作和竞争),但是在解决实际问题时需要将实验环境和算法的接口统一. 实验结果表明,与其他典型算法相比,Mi-DDPG算法在各种环境中回合平均奖励最高,收敛速度最快;在各个环境中均能满足任务要求. 但还存在应用模块化Q函数方法可能会丢失一些重要信息的问题,下一步将拓展研究范围,继续进行以多agent之间的通信为基础的多agent算法研究.

参考文献(References)

[1] Sutton R S, Barto G A. Reinforcement learning: An introduction[M]. Cambridge: MIT Press, 1998.

[2] Silver D. Mastering the game of go with deep neural networks and tree search[J]. Nature, 2016, 529(7587): 484-489.

[3] Levine S, Finn C, Darrell T, et al. End-to-end training of deep visuomotor policies[J]. The Journal of Machine Learning Research, 2016, 17(1): 1334-1373.

[4] 吴军, 徐昕, 王健, 等. 面向多机器人系统的增强学习研究进展综述[J]. 控制与决策, 2011, 26(11): 1601-1610.
(Wu J, Xu W, Wang J, et al. A review of research progress on enhanced learning for multi-robot systems[J]. Control and Decision, 2011, 26(11): 1601-1610.)

[5] Matignon L, Jeanpierre L, Mouaddib A I. Coordinated multi-robot exploration under communication constraints using decentralized Markov decision processes[C]. Proceedings of the 26th AAAI Conference on Artificial Intelligence. Palo Alto: AAAI Press, 2012: 2017-2023.

[6] Peng P, Wen Y, Yang Y, et al. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games[J/OL]. (2017-09-14). https://arxiv.org/abs/1703.10069.

[7] Littman M L. Markov games as a framework for multi-agent reinforcement learning[C]. Proceedings of the Eleventh International Conference on Machine Learning (ML-94). San Francisco: Morgan Kaufmann, 1994: 157-163.

[8] Watkins C J C H, Dayan P. Q-learning[J]. Machine Learning, 1992, 8(3/4): 279-292.

[9] Ronald J, Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning[J]. Machine Learning, 1992, 8(3/4): 229-256.

[10] Sukhbaatar S, Szlam A, Fergus R. Learning multiagent communication with backpropagation[C]. Proceedings of the 30th International Conference on Neural Information Processing Systems. Red Hook: Curran Associates Inc., 2016: 2252-2260.

[11] Foerster J N, Assael Y M, de Freitas N, et al. Learning to communicate with deep multi-agent reinforcement learning[C]. Proceedings of the 30th International Conference on Neural Information Processing Systems. Red Hook: Curran Associates Inc., 2016: 2145-2153.

[12] Vinyals O, Ewalds T, Bartunov S, et al. Starcraft ii: A new challenge for reinforcement learning[J/OL]. (2017-08-16). https://arxiv.org/abs/1708.04782.

[13] Schuster M, Paliwal K K. Bidirectional recurrent neural networks[J]. IEEE Transactions on Signal Processing, 1997, 45(11): 2673-2681.

[14] Lowe R, Wu Y, Tamar A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments[C]. Proceedings of the 31st International Conference on Neural Information Processing Systems. Red Hook: Curran Associates Inc., 2017: 6382-6393.

[15] Kai Arulkumaran. Deep reinforcement learning: A brief survey[J]. IEEE Signal Processing Magazine, 2017, 34(6): 26-38.

[16] Volodymyr Mnih. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529-533.

[17] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning[J/OL]. (2019-07-05). https://arxiv.org/abs/1509.02971.

[18] Zhang P, Xue J, Lan C, et al. Adding attentiveness to the neurons in recurrent neural networks[C]. European Conference on Computer Vision. Cham: Springer, 2018: 136-152.

作者简介

陈亮(1979−), 男, 副教授, 博士, 从事人工智能技术、大数据分析等研究, E-mail: kongkuchen@126.com;

梁宸(1994−), 男, 硕士生, 从事机器学习、决策支持的研究, E-mail: 18555006386@163.com;

张景异(1965−), 男, 教授, 硕士, 从事电工与电子技术、自动控制等研究, E-mail: zjy9668@139.com;

刘韵婷(1983−), 女, 副教授, 博士, 从事人工智能、无线传感器网络等研究, E-mail: 71019976@qq.com.

(责任编辑：孙艺红)