文章编号：2095-2163(2020)05-0291-07 中图分类号：TP389.1 文献标志码：A

基于深度强化学习的群体对抗策略研究

刘 强，姜 峰
（哈尔滨工业大学 计算机科学与技术学院，哈尔滨 150000）

摘 要：多智能体强化学习方法之前一直在博弈论和控制论的基础上进行研究，但是实验结果表明，这一类多智能体强化学习方法无法处理现实生活中的复杂问题。直到最近几年深度强化学习技术的成熟，给群体智能的研究带来了新的解决方案。通过深度神经网络来拟合策略函数，使得智能体有更强的处理复杂问题的能力。本文主要研究多智能体强化学习方法在对抗与协作环境下的应用，以及算法稳定性的提升和智能体规模的扩大，使得智能体能够像人类一样在复杂环境下互相协作地与其他智能体进行对抗。

关键词：深度强化学习；多智能体强化学习；深度神经网络

Research on group confrontation strategies based on deep reinforcement learning

LIU Qiang, JIANG Feng
(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150000, China)

【Abstract】 The Multi-Agent Reinforcement learning method has been studied on the basis of game theory and cybernetics before, but the experimental results show that this kind of Multi-Agent Reinforcement learning method can't deal with the complex problems in real life. Until the maturity of deep reinforcement learning technology in recent years, new solutions have been brought to the study of swarm intelligence. The deep neural network is used to fit the strategy function, which makes the agent have stronger ability to deal with complex problems. This topic mainly studies the application of multi-agent reinforcement learning method in the environment of confrontation and cooperation, the improvement of algorithm stability and the expansion of the scale of agents, so that agents can cooperate with other agents in the complex environment just like human beings.

【Key words】 Deep reinforcement learning; Multi-Agent reinforcement learning; Deep neural network

0 引 言

早在 2000 年，美国国防预先研究计划局（DARPA）就曾借鉴蚂蚁信息素交互行为，开展过无人机集群的空战仿真研究。2015 年 9 月美国国防高级研究计划局（DARPA）发布了“小精灵”（Gremlins）项目公告，项目设想通过发射大量微小无人机对敌防御系统进行饱和攻击，如通过 C-130 运输机在防区外发射携带侦察与电子战装备的无人机蜂群执行离岸电子攻击与侦察等任务，在执行完任务后，对幸存的无人机进行回收。2017 年 1 月，美国海军 3 架 F/A -18F “超级大黄蜂”战斗机以 0.6马赫的速度投放了 103 架 Perdix 无人机，创下美军军用无人机蜂群最大规模飞行纪录。试验中，“山鹑”蜂群未预先编写飞行程序，而是在地面站指挥下自主实现协同，展现了集体决策、自修正和自适应编队飞行能力。鉴于为对抗无人机集群的攻击，目前最可能有效的方法是利用无人机集群对入侵的无人机集群进行拦截，而进行群对抗的关键就是多智能体强化学习方法。本文提出结合已有的深度学习和强化学习方法，以 gym 为研究平台，在虚拟对战平台中实现无人机群的群对抗策略。

1 多智能体深度强化学习理论

1.1 单智能体强化学习

强化学习（Reinforcement Learning）是关于决策优化的科学，是一种环境状态映射到动作的学习，目标是使 Agent 在与环境的交互过程中获得最大的累积奖励。马尔科夫决策过程（Markov Decision Process，MDP）可以用来对 RL 问题建模。通常将 MDP 定义为一个四元组 (S,A,ρ,f) [1]，其中：

（1）S 为所有环境状态的集合，$s_t \in S$ 表示 Agent 在 t 时刻所处的状态。

（2）A 为 Agent 可执行动作的集合，$a_t \in A$ 表示 Agent 在 t 时刻所采取的动作。

（3）$\rho: S \times A \rightarrow R$ 为奖赏函数，$r_t \sim \rho(s_t, a_t)$ 表

基金项目：国家重点研发计划项目资助（SQ2018YFC0806802，2018YFC0832105）。

作者简介：刘 强（1997-），男，硕士研究生，主要研究方向：深度学习、强化学习；姜 峰（1978-），男，博士，教授，博士生导师，主要研究方向：计算机视觉、机器学习、视频编解码。

收稿日期：2020-03-10

示 Agent 在状态 s_t 下执行动作 a_t 获得的立即奖赏值。

(4) $f:S\times A\times S\rightarrow[0,1]$ 为状态转移概率分布函数，$s_{t+1}\sim f(s_t,a_t)$ 表示 Agent 在状态 s_t 执行动作 a_t 转移到下一个状态 s_{t+1} 的概率。

Agent 的目标是最大化累积奖励，式(1)：

$$G_t=R_t+R_{t+1}+R_{t+2}+\cdots+R_T. \tag{1}$$

其中，G_t 代表目标，T 代表结束的时间节点。

为了使目标最大，需要一个策略 π 计算每个状态 s 映射到 a 的概率，Agent 要学习的就是何如找到一个最优的策略 π 使得 G_t 最大。在实际问题中，考虑累积奖励时会在每一项上加上衰减因子 γ，用来调节不同时间的奖励对目标的影响。具体来讲就是距离当前状态越近，获得的回报对现在的影响越大。所以，式(1)可以修改为式(2)：

$$G_t=\sum_{t'=t}^{\mathrm{T}}\gamma^{t'-t}R_{t'}. \tag{2}$$

其中，$\gamma\in[0,1]$。根据 G_t 的定义，在状态 s 下，它的期望回报(价值函数 Value Function)定义如下：

$$V(s)=E[G_t\mid S_t=s]. \tag{3}$$

将(3)式展开：

$$\begin{aligned}V(s)&=E[G_t\mid S_t=s]=\\&E[R_t+\gamma R_{t+1}+\gamma^2R_{t+2}+\cdots+\gamma^{\mathrm{T}-t}R_T\mid S_t=s]=\\&E[R_t+\gamma(R_{t+1}+\gamma R_{t+2}+\cdots+\gamma^{\mathrm{T}-t-1}R_T)\mid S_t=s]=\\&E[R_t+\gamma G_{t+1}\mid S_t=s]=\\&E[R_t+\gamma V(S_{t+1})\mid S_t=s].\end{aligned}$$

因此，可以得到 $V(s)$ 的迭代公式(4)：

$$V(s)=E[R_t+\gamma V(S_{t+1})\mid S_t=s]. \tag{4}$$

这就是贝尔曼方程的基本形态[2]。知道了每个状态的价值是不能帮助 Agent 直接做出决策的。如果 Agent 知道了当前状态下每个动作的价值，那么直接选择一个价值最大的动作就可以了。于是有了强化学习中另一个重要的函数，叫做动作-价值函数 $Q^{\pi}(s,a)$。动作价值函数指的是在当前状态 s 下执行动作 a，并一直遵循策略 π 到结束为止，这一过程中 Agent 所获得的累积回报表示为式(5)：

$$\begin{aligned}Q^{\pi}(s,a)&=E[r_t+\gamma r_{t+1}+\gamma^2r_{t+2}+\cdots+\gamma^{\mathrm{T}-t}r_T\mid s,a]=\\&E_{s'}[r_t+\gamma Q^{\pi}(s',a')\mid s,a].\end{aligned} \tag{5}$$

因为要找的是最优策略，所以最优策略定义为式(6)：

$$Q^{*}(s,a)=\max_{\pi}Q^{\pi}(s,a). \tag{6}$$

结合(4)式，可以得到动作价值函数的迭代公式(7)：

$$Q^{*}(s,a)=E_{s'}[r_t+\gamma\max_{a'}Q^{*}(s',a')\mid s,a]. \tag{7}$$

状态值函数(Value Function)和动作价值函数(Q Function)是强化学习中最重要的两个函数，状态值函数负责评估当前状态的平均价值，动作价值函数负责得出做出某一个具体动作的价值是多少。

1.2 深度强化学习

强化学习的概念早在上世纪就已经被提出了，由于计算能力等问题一直没有取得突破性的进展。直到深度神经网络的出现，Mnih 等人将深度神经网络和传统的 Q 学习算法相结合，提出了 Deep Q-Network(DQN)算法[3]。这是深度强化学习领域开创性的进展，网络模型如图 1 所示。

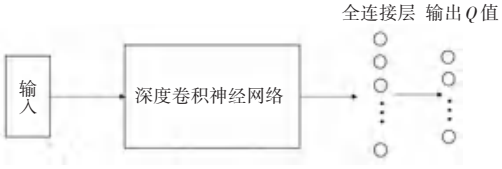


图 1 深度 Q-Network 模型

Fig. 1 Deep Q-Network model

DQN 算法是一种基于价值 Value 的算法，是一种间接的方法。因为在决策的时候还要根据 $1-\varepsilon$ 策略选择动作。$1-\varepsilon$ 策略是以 $1-\varepsilon$ 为概率，选择状态价值最大的动作，以 ε 为概率随机选择动作。既然可以利用深度神经网络去近似表达动作价值函数或值函数，那么也可以利用深度神经网络直接近似表达策略 π[4]。Policy Network 也就是策略网络，输入是当前的状态，输出直接就是动作。一般情况下，策略网络的输出一般是动作的概率，如图 2 所示。

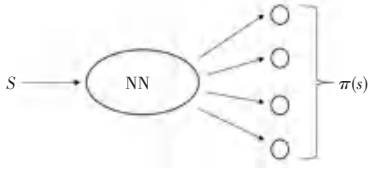


图 2 策略网络图

Fig. 2 Policy network

Q-Learning 是一种基于价值的方法，即通过计算每个状态动作的价值，选择价值最大的动作执行，这是一种间接的做法，更直接的做法是直接更新策略网络(指网络直接根据状态输出动作或者动作的概率)。对于策略网络的训练思路和价值网络稍有不同，因为它要表达的是在状态 s 下每个可选动作的可能性大小，所以损失函数不容易构造。一个最简单的想法就是根据执行完动作后的奖励大小来调整动作出现的概率，即获得的奖励越大，动作出现的概率越大。可以构造一个评价网络来评判当前决策

的好坏，来指导策略网络的训练，这就是 Actor-Critic[5]模型的基本思想，结构如图 3 所示。

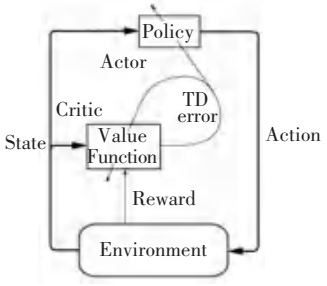


图 3　Actor-Critic 模型结构图

Fig. 3　Actor-Critic model structure

1.3　多智能体强化学习

多智能体强化学习也要服从马尔可夫决策过程[6]，马尔可夫决策过程的多智能体扩展(MDPs)，称为部分可观察马尔可夫博弈。N 个智能体的马尔可夫博弈定义为描述所有智能体的可能配置的一组状态 S，动作 $A_1,\dots,A_N$ 和每个智能体的观测值 $O_1,\dots,O_N$，具体如图 4 所示。

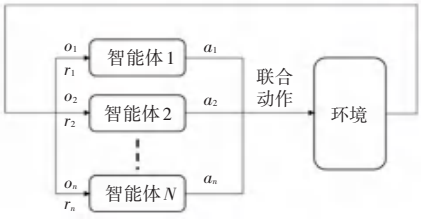


图 4　多智能体与环境交互图

Fig. 4　Multi-agent and environment interaction diagram

动作 $a_1,\dots,a_N$ 成为多智能体的联合动作，如果每个智能体的观测值 $O_1,\dots,O_N$ 都是可观察的，叫做完全可观察。但是在现实生活中，每个智能体是不可能知道其他智能体的全部信息的，对于单个智能体来说环境对它来讲就是非稳定的，不符合最基本的马尔可夫假设，这也是多智能体强化学习中的一个难点。当选择动作时，每个智能体使用随机策略 $\pi_{\theta_i}:O_i\times A_i\to[0,1]$，根据状态转移函数产生下一个状态 $T:S\times A_1\times\cdots\times A_N\to S^2$。每个智能体都是根据在当前状态下执行的动作获得相应的奖励 r_i：$S\times A_i\to R$，并收到与各自状态相关的观测值：$o_i:Sa\mapsto O_i$。每个智能体旨在最大化自己的总预期回报 $R_i=\sum_{t=0}^{\mathrm{T}}\gamma^t r_i^t$，其中 γ 是折扣因子，T 是时间范围。与单智能体强化学习不同，多智能体中对每个智能体动作的奖励和状态转移函数不仅取决于智能体自身的动作和观测值，也取决于其他智能体的动作和观测值。多智能体的目的是最大化累积奖励，如式(8)所示：

$$J_i(\pi_i)=E_{a_1\sim\pi_1,\cdots,a_N\sim\pi_N,o\sim T}\left[\sum_{t=0}^{¥}\gamma^t r_{it}(o_t,a_{1t},\cdots a_{nt})\right]. \tag{8}$$

2　MADDPG 方法

MADDPG(Multi Agent Deep Deterministic Policy Gradient)算法采用的是去中心化策略，用一个中心化的 Critic 和去中心化的 Policies[7]。它的意思是每个智能体执行的动作仅依据于它们自己的动作观察历史上。但是这又可能损失一些交互信息，单个智能体无法找到对于全局来说最优的动作。MADDPG 算法是基于 DDPG 算法，所以依然是 Actor-Critic 架构。在训练时，每个智能体的 Actor 网络部分仍然是以自己的局部观察作为输入，不同于 DDPG 算法的是，Critic 网络部分是以全局观察作为输入。假定目前的智能体数目是两个，对应的结构图如图 5 所示。

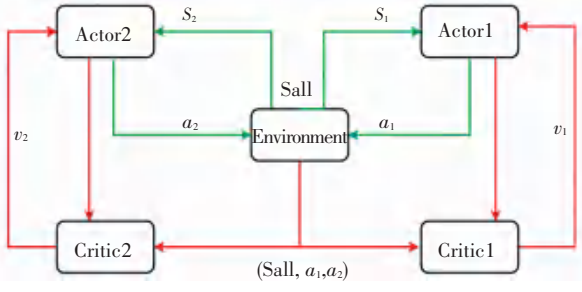


图 5　MADDPG 算结构简图

Fig. 5　MADDPG algorithm structure

具体地说，考虑策略由 $\theta=\{\theta_1,\dots,\theta_N\}$ 参数化的具有 N 个智能体的博弈，令所有智能体策略的集合为 $\pi=\{\pi_1,\dots,\pi_N\}$，那么可以表示出智能体 i 的期望收益的梯度 $J(\theta_i)=E[R_i]$ 如式(9)所示：

$$\tilde{\mathrm{N}}_{\theta_j}J(\theta_i)=E_{s\sim p^{\mu},a_i\sim\pi_i}[\tilde{\mathrm{N}}\theta_i\log_{\pi_i}(a_i\mid o_i)\,Q_i^{\pi}(x,a_1,\cdots,a_N)]. \tag{9}$$

这里 $Q_i^{\pi}(x,a_1,\cdots,a_N)$ 是一个集中的动作值函数，函数的输入是所有的智能体的动作 $a_1,\cdots,a_N$，以及它们的状态信息 x，输出则是智能体 i 的 Q 值。如果环境比较简单的话，x 可以是所有智能体的观测值也就是 $x=(o_1,\cdots,o_N)$；如果环境比较复杂，为了便于训练，可能还会加入通讯信息等。由于每个 Q_i^{π} 是分开学习的，可以随意的给智能体设置对应的奖赏方式，比如达到某一目的而受到奖励，或者是碰到某一障碍而受到惩罚。将以上的思路应用于确定性策略中。考虑 N 个策略 μ_{θ_i}，其对应参数是 θ_i(缩写为 μ_i)，则梯度可以写成式(10)的形式：

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

$$E_{x,a\sim D}[\nabla_{\theta_i}\mu_i(a_i \mid o_i)\,\nabla_{a_i} Q_i^{\mu}(x,a_1,\cdots,a_N)\mid_{a_i=\mu_i(o_i)}]. \quad (10)$$

MADDPG 算法为了提高样本的利用率,采用 off-policy的方式训练,将元组 $(x,x',a_1,\ldots,a_N,r_1,\ldots,r_N)$ 放入 replay buffer D 中,记录下所有智能体的经验。集中的动作值函数 Q_i^{μ} 更新方式如式(11)、(12)所示:

$$L(\theta_i) = E_{x,a,r,x'}[(Q_i^{\mu}(x,a_1,\cdots,a_N) - y)^2], \quad (11)$$

$$y = r_i + \gamma\, Q_i^{\mu'}(x',a_1',\ldots,a_N')\mid_{a_j'=\mu_j'(o_j)}. \quad (12)$$

其中,$\mu' = \{\mu_{\theta_1'},\ldots,\mu_{\theta_N'}\}$ 是具有延迟参数 θ_i' 的目标策略集合。

3 带注意力机制的群体对抗策略

3.1 TRPG(Trust Region Policy Gradient)算法思想

DDPG 方法的 Actor 网络也是基于 Policy Gradient 方法的,Policy Gradient 的缺点是更新的幅度和训练的步长难以确定。如果训练步长太大,那么训练出来的策略会很躁动,难以收敛;而与之相反,若训练步长过小,那么训练的速度将会大打折扣,难以付出那么多的时间与代价去等待它完成训练,效率会很低。Policy Gradient 的另一个缺点是参数更新比较慢,每更新一次参数都需要进行重新采样,这其实就是 on-policy 策略,即想要训练的智能体和与环境交互的智能体是同一个智能体;与之对应的就是 off-policy 策略,即想要训练的智能体和与环境交互的智能体不是同一个智能体[8]。也就说拿别人的经验来训练自己。为了提升训练速度,让采样的数据可以重复使用,将 on-policy 方式转换为 off-policy,具体方法就是 replay buffer(经验回放)。DDPG 方法采用的就是 off-policy 策略,但是这种方法涉及到了置信域问题[8],也就是说与环境交互的智能体的策略 π_{old} 和要训练的智能体的策略 π_{new} 的分布不能够差太大,否则要进行多次的采样,才能得到近似的结果。这里的 π_{old} 和 π_{new} 的分布不能差太大的意思是输入同样的 state,网络得到的动作的概率分布不能差太远,那么只要限制参数更新的幅度。Actor 的新的参数更新方式如式(13):

$$L(\theta) =$$

$$E_t\left[Min\left(\frac{\pi_\theta}{\pi_{\theta_{old}}}A_t, clip\left(\frac{\pi_\theta}{\pi_{\theta_{old}}}(\theta),1-\varepsilon,1+\varepsilon\right)A_t\right)\right]. \quad (13)$$

其中,A_t 叫优势函数(advantage function),在规定时间只向后算一个时间步长的情况的 TD 误差就是优势函数。在传统的 Actor-Critic 框架中,只有一个 Critic 网络用来计算优势函数,基于 DDPG 方法的双 Critic 的优势,可以将 A_t 改写如式(14):

$$A_t = r_t + \gamma * V_{target}(s') - V_{online}(s). \quad (14)$$

3.2 带注意力机制的多智能体 TRPG 算法

本文针对 DDPG 算法的不足之处提出了 TRPG 算法,MATRPG 算法的核心思想是将 DDPG 算法的改进算法 TRPG 算法应用到多智能体的环境当中。将式(13)中的策略梯度应用到多智能体强化学习中,得到 MATRPG 算法中每个智能体 Actor 网络的策略梯度如式(15)所示:

$$\nabla_{\theta_i} J(\pi_i) = E_{x,a\sim D}\left[\nabla_{\theta_i} Min\left(\frac{\pi_{\theta_i}}{\pi_{\theta_i'}}, clip\left(\frac{\pi_{\theta_i}}{\pi_{\theta_i'}},1-\varepsilon,1+\varepsilon\right)\right)\nabla_{a_i} A_i^{\pi}(x,a_1,\cdots,a_N)\mid_{a_i=\pi_i(o_i)}\right]. \quad (15)$$

MADDPG 算法虽然解决了整体状态评估的问题,但是没有解决在智能体的规模逐渐增加时纬度爆炸问题。即使采用了去中心化的思想,每个智能体的 Actor 网络只需要以自己的局部信息作为输入,但是中心的 Critic 网络还是要以所有智能体的状态信息作为输入,当智能体的规模不断增加的时候,会因为状态空间过大产生难以收敛的问题。本文为解决智能体集群规模较大时群体强化学习算法难以收敛的问题,引入 Attention 机制。说针对某一个智能体,它不必对于每一个伙伴都予以相同的关注程度,只需要关注自己周围的某些智能体。例如,在一个足球场上,己方的门将根本无需关注敌方门将和后卫所作的动作。通过引入多头注意力机制,使得智能体学会不必关注每一个智能体的状态和动作,这样在智能体的规模变大时,可以只关注自身周围的局部信息,减少 Critic 网络的计算复杂度。注意力机制构图如图 6 所示。

从图 6 可以看出,在计算每个智能体的 Q-function 时,并不是简单的将智能体自身的观察和其他智能体的观察在一起作为 Q-function 的输入,其计算方式如式(16)所示:

$$Q_i(o,a) = f_i(g_i(o_i,a_i),x_i). \quad (16)$$

式中,f_i 是一个两层的多层感知机(MLP)网络,g_i 是一个单层的 MLP 网络,x_i 代表其他智能体的贡献,是其他智能体的价值的加权求和。计算方式如式(17)所示:

$$x_i = \sum_{j\neq i}\alpha_j v_j = \sum_{j\neq i}\alpha_j h(V g_j(o_j,a_j)). \quad (17)$$

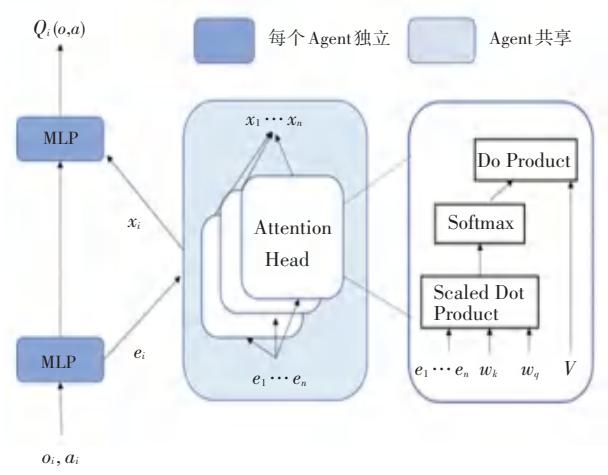


图 6　Multi-Head Attention 结构图

Fig. 6　Multi-Head Attentionstructure

式中，h 是一个非线性激活函数 Relu，v_j 是经过将编码器g_j和一个共享的权值矩阵V计算而来，α_j 是注意力权重，代表 *agen* t_i 和 *agen* t_j 的相似性权重，计算如式(18)所示：

$$\alpha_j = \exp\left(e_j^{\mathrm{T}} W_k^{\mathrm{T}} W_q e_i\right). \tag{18}$$

式中，W_q 将 e_i 转换成“query”，W_k 将 e_j 转换成“key”，根据这两个矩阵的维数进行匹配，以防止梯度消失。多头注意力机制的每个“头”使用单独的参数 (W_q,W_k,V)，这就使得每个智能体对于其他智能体的关注度不相同。

4　实　验

4.1　单智能体算法对比

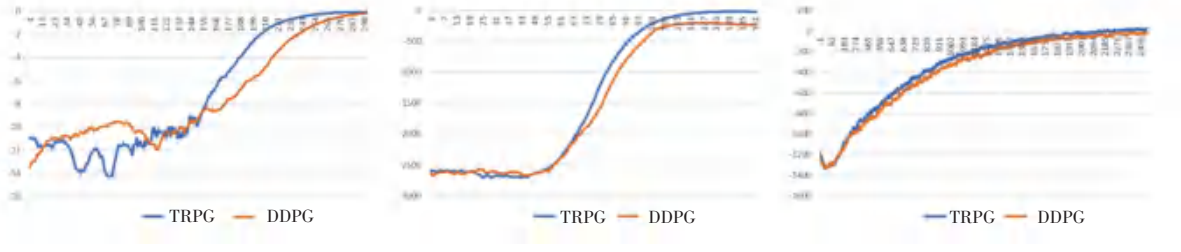
由于 TRDPG 算法是基于单智能体算法的改进，所以本文在单智能体的环境下验证算法的有效性。实验在 gym 平台上进行，主要的实验环境有 MountainCarContinuous-v0、Pendulum-v0 和 CartPole-V0。场景如图 7 所示。

(a) MountainCarContinuous-v0　　(b) Pendulum-v0　　(c) CartPole-V0

图 7　实验环境图

Fig. 7　Experimental environment

本文在以上三种实验环境中对 TRPG 算法和 DDPG 算法的效果进行了对比分析。横坐标是训练次数，纵坐标是评价指标，mean_episode_rewards 是指该智能体在当前 episode 获得的平均奖励。mean_episode_rewards 由碰撞次数、能量等加权构成。实验结果如图 8 所示。



(a) MountainCarContinuous-v0　　(b) Pendulum-v0　　(c) CartPole-V0

图 8　TRPG 与 DDPG 算法对比图

Fig. 8　Comparison of TRPG and DDPG algorithms

4.2　多智能体算法对比

本文在 gym 模拟平台上对比了 MATRPG 算法和基础的 MADDPG 算法。为了适合应用场景，设计了一个新的评价指标：平均碰撞次数，即追捕智能体在 100 个回合中平均能碰撞到被追捕智能体的次数。两种算法在 3V2 的 simple_tag 场景(如图 9 所示)下的平均碰撞次数，如图 10 所示。绿色智能体需要躲避红色智能体的追捕，红色智能体的目标是协作围捕绿色的智能体，图中黑色部分是障碍物，追捕者和被追捕者都无法穿过。

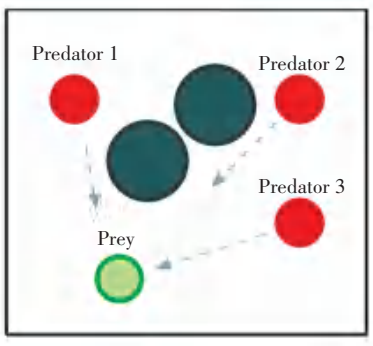


图 9 simple_tag 场景图

Fig. 9 Simple_tag scene

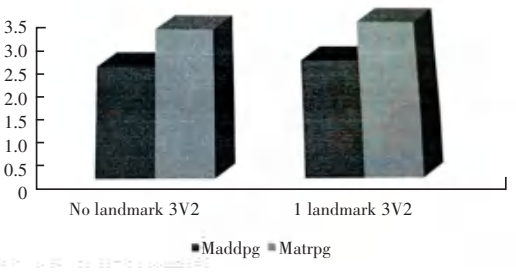


图 10 平均碰撞次数对比图

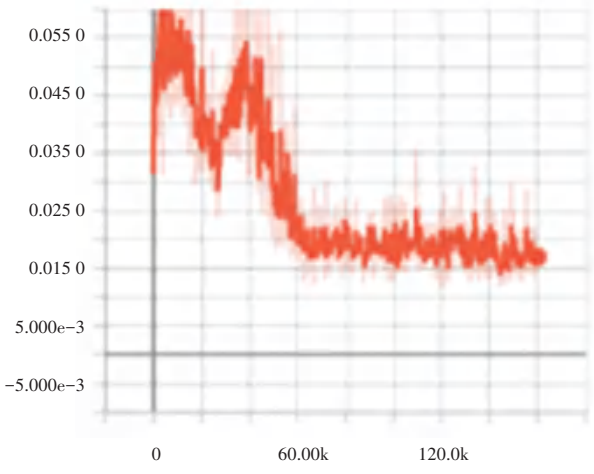
Fig. 10 Comparison of average collision times

追捕智能体分别采用 MADDPG 算法和 MATRPG 算法,被追捕智能体采用单智能体 DDPG 算法,对比了在 3V2 模式下的两种场景,分别是有 landmark(障碍物)场景和 no_landmark(无障碍物)场景,可以看出采用加速方案后的平均碰撞次数有所提升。MADDPG 方法和 MATRPG 方法训练过程中的损失函数如图 11 所示。通过对比发现,在相同的迭代次数的情况下,改进过后的 MAPTRPG 算法有更好的探索率,损失函数的下降更快。

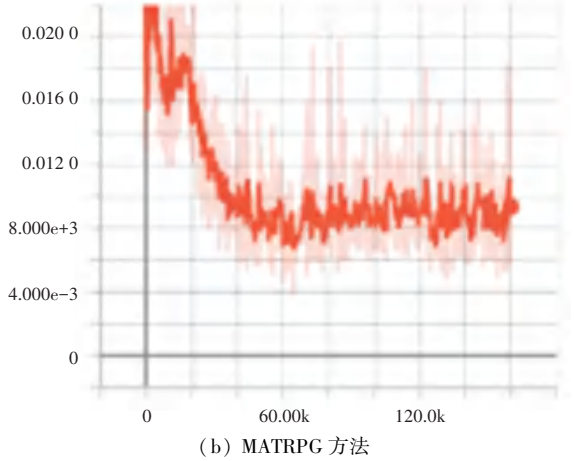
为了验证加入注意力机制后的 MAATRPG(Multi Agent Attention TRPG)方法的有效性,本文在多个实验环境下分别对 MADDPG、MATRPG 和 MAATRPG 进行对比。实验环境分为协作环境 Rover-Tower 场景和混合环境(竞争与协作)下的 simple_tag 场景。Rover-Tower 场景如图 12 所示,指挥塔需要指挥与它配对的漫游者到与它颜色相匹配的目的地。

本文首先对比了各算法在 simple_tag 3V2 和 5V3 场景下的平均碰撞次数,如图 13 所示。

图 13 对比了在 3V2 模式下的两种场景,分别是有 landmark(障碍物)和 no_landmark(无障碍物),可以看出采用加速方案后的平均碰撞次数虽然有提升但是提升效果不大,在加入注意力机制后有比较明显的提升。在 3V2 场景上获得比较好的实验效果后,本文还对比了各算法在 5V3 规模下的平均碰撞次数,如图 14 所示。这里没有统计 MADDPG 算法是由于 MADDPG 算法无法处理这种规模升级的情况。从图中可以看出加入注意力机制后,智能体的表现更有优势。



(a) MADDPG 方法



(b) MATRPG 方法

图 11 损失函数比较

Fig. 11 Comparison of algorithms's loss function

图 12 Rover-Tower 场景图

Fig. 12 Rover-Tover scene