基于互信息的智能博弈对抗分层强化学习研究

魏竞毅,赖 俊,陈希亮
(陆军工程大学 指挥控制工程学院,江苏 南京 210007)

摘 要:智能博弈在当前人工智能的发展中是较为热点的一个问题,同时随着人工智能的不断发展,在作战指挥领域也逐渐得到了广泛的应用,尤其以美国 DAPPA 为首,利用人工智能来为指挥员的战场决策提供全方位的策略支持,如何利用人工智能模拟战场环境下进行战场对抗也是研究的一方面。当前智能体虽然能够通过获得奖励不断进行优化,在策略上通常是根据即时奖励选择当时收益最大的策略,现实战场环境中有些决策当时虽不会有即时收益,但之后是会对整体的战场形势有更好的推动作用,能够取得更有利的战果。针对此问题,利用分层强化学习进行智能体的智能博弈训练,并应用于简单战场环境下来模拟虚拟指挥员,提出了一种基于互信息的智能博弈对抗的分层强化学习算法 MI-A3C。MI-A3C 算法在模拟的战场环境中能够取得 86.7% 的胜率,并能够完成主要任务,同时在实验中可以发现一些有利于长远收益的决策。

关键词:智能博弈;强化学习;互信息;分层;A3C 算法;分队指挥
中图分类号:TP181 **文献标识码**:A **文章编号**:1673-629X(2022)09-0142-06
doi:10.3969/j.issn.1673-629X.2022.09.022

Research on Hierarchical Reinforcement Learning of Intelligent Game Confrontation Based on Mutual Information

WEI Jing-yi, LAI Jun, CHEN Xi-liang
(School of Command Information System, Army Engineering University, Nanjing 210007, China)

Abstract: Intelligent game is a hot issue in the current development of artificial intelligence. At the same time, with the continuous development of artificial intelligence, it has gradually been widely used in the field of battle command. Especially, led by American DAPPA, artificial intelligence is used to provide all-round strategic support for commanders' battlefield decisions. How to use artificial intelligence to simulate battlefield confrontation in battlefield environment is also one of its research aspects. At present, although agents can continuously optimize by obtaining rewards, they are usually real-time strategies in strategy. Although some decisions in battlefield environment will not have immediate benefits at that time, but then it will play a better role in promoting the overall battlefield situation and achieve more favorable results. To solve this problem, hierarchical reinforcement learning is used for intelligent game training of agents and applied to simulate virtual commanders in a simple battlefield environment. A hierarchical reinforcement learning algorithm MI-A3C algorithm based on intelligent game confrontation based on mutual information is proposed. MI-A3C algorithm can achieve 86.7% victory rate in the simulated battlefield environment, and can complete the main tasks. At the same time, some decisions conducive to long-term benefits can be found in the experiment.

Key words: intelligent game; reinforcement learning; mutual information; hierarchical; A3C algorithm; unit commander

0 引 言

近年来随着深度强化学习[1-2]开始不断发力,在各项自主学习任务中,不断取得进展,但由于环境复杂度的提高,发现学习的参数会随着状态变量的增加而成指数增加,引发维度灾难,为减少学习训练的复杂度,分层深度强化学习应运而生。当前大多数强化学习完成任务是根据智能体动作获取的奖励来进行优化,实现最大化动作奖励来实现优化策略,例如经典的 DQN[3]、DDPG[4]、PPO[5]等。为了能学习到更多不同的策略,算法 DIAYN[6]、DADS[7]通过引入互信息的理念,依据对智能体和环境状态变化来对策略进行分类,使得智能体能够不仅仅依靠环境即时收益来学习策略,同时根据智能体和环境的状态变化来进行策略学习,最终在导航和机械臂的实验中取得良好效果。在

收稿日期:2021-08-07 **修回日期**:2021-12-09
作者简介:魏竞毅(1993-),男,硕士研究生,通讯作者,研究方向为分层强化学习;赖 俊,副教授,研究方向为人工智能、计算机仿真;陈希亮,副教授,研究方向为深度强化学习、指挥信息系统工程。

战场环境中,有些策略在当时并不会产生即时的奖励,但对于任务的完成会起到助推作用,同时在战场环境中这种前期埋下伏笔的策略对整体战场形势会在后期产生很大的作用,当智能体仅依靠即时奖励时就会忽视这种策略。该文为训练智能体不受限于即时收益,可以学习具有长远收益有利于任务完成的策略,通过引入互信息[8](Mutual Information)的概念,依据环境和智能体的不同状态来进一步对策略进行分类,使得智能体能够发现更多的策略,不仅仅发现能获得奖励的策略,同时对于这些有利于任务完成的策略也进行保留,使得任务的完成率得到提升,并对智能体动作采用了 option 分层[9]的思想,对动作进行了划分。为了加快智能体的训练效率,采用分布式的训练方式,与 A3C[10](Asynchronous Advantage Actor-Critic)算法进行结合,称这一算法为 MI-A3C。

1　基于互信息的智能博弈对抗算法描述

基于互信息的智能博弈算法是在 A3C 算法的基础上,通过引入互信息的概念来进一步增强智能体的探索能力,将环境和智能体的状态与互信息理论相结合,来促使智能体学到的策略,可以使环境状态有较大变化同时保证不同策略的环境状态变化也不同,以此来增强策略的有效性和多样性。最终学习到的算法不仅仅可以依靠环境的即时收益来进行参数更新,同时策略的互信息也会影响算法的参数更新,下面针对算法的一些基础理论和整体结构进行介绍。

1.1　互信息的基本理论及应用

信息熵是一个随机变量不确定性的度量值,对于一个离散型随机变量 $X \sim p(x)$,公式表示为:

$$H(X) = -\sum_{x \in X} p(x) \log p(x) \tag{1}$$

一个随机变量的信息熵越大,也就意味着不确定性就越大,包含的信息量也就越大,必然事件的信息熵为 0,随机概率均等事件的信息熵为 1。

互信息(Mutual Information)[11]是衡量随机变量之间相互依赖程度的度量,是信息论中的一个重要概念,表示为 $I(X;Y)$,公式表示为:

$$I(X;Y) = H(X) - H(X \mid Y) \tag{2}$$

其中,$H(X)$ 表示随机变量 X 的信息熵,$H(X \mid Y)$ 表示在条件 Y 的情况下,随机变量 X 的信息熵,互信息 $I(X;Y)$ 表示当知道已经发生 Y 时,随机变量 X 的信息量减少多少,具体表示为图 1。

将互信息这一概念应用在强化学习中,通常将 X 表示为状态,Y 表示为动作,当使得互信息最大化为目标函数时,也就表示当采取这一动作后,环境的信息熵会增大,也就使得动作策略具有更多探索性,这也就增大了对环境的探索能力,使得环境状态具备更多的可能性,会产生更多的策略。

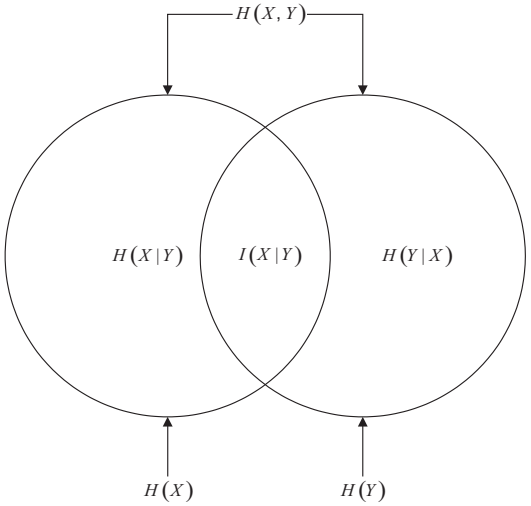

图 1　互信息与信息熵的文氏图

1.2　A3C 算法

A3C 算法是对基于策略梯度(Policy Gradient)的算法 AC 的优化,该算法采用的是异步的训练方式。如图 2 所示,在 A3C 算法中有一个全局网络(Global Network),这个网络是一个公共的神经网络模型,包含公共 Actor 网络和 Critic 网络,同时下面有 n 个线程,每个线程都会独立地与环境进行交互并得到经验,这些线程之间独立互不干扰,收集完经验后,独立完成训练并得到参数更新量,之后异步更新到全局网络的模型参数中。下次训练时,各线程的模型参数与全局网络进行同步,再次进行训练。

AC 算法的优势函数定义为:

$$A(s,a) = Q(s,a) - V(s) = r + \gamma V(s') - V(s) \tag{3}$$

A3C 算法为了使训练能够在早期更快收敛,采取了 n 步回报估计法,优势函数变为:

$$A(s,a) = R_t + \gamma R_{t+1} + \cdots + \gamma^{n-1} R_{t+n-1} + \gamma^n V(S') - V(S) \tag{4}$$

同时为了增加模型的探索性,A3C 算法在模型的目标函数中加入策略的熵,熵越大也意味着随机策略拥有越多的信息,也使得模型可以拥有更强的多样性。策略梯度的计算公式为:

$$\nabla_\theta J(\theta) = \frac{1}{T} \sum_{T} \nabla_\theta \log \pi(a_t \mid s_t;\theta) \left[\sum_{i=1}^{n} \gamma^{i-1} r_{t+n+1} + \gamma^n V(s') - V(s) \right] + \beta \nabla_\theta H(\pi(s_t;\theta)) \tag{5}$$

2　基于互信息的智能博弈对抗算法流程

采用互信息来实现智能体的策略发现主要有三点要求:(1)使智能体具有更强的探索性,可以对环境进

行更多的探索,使智能体与环境交互产生更多样的状态 s ;(2)通过状态 s 来对策略进行区分,当一系列动作没有对环境产生影响时,认为这些动作是无用,而是将一系列对环境产生影响的动作划分为策略;(3)有用的策略可以控制智能体到达不同的状态,所以策略是具有可分辨性的可划分的。目前采用互信息的算法有 DIAYN[6]、DADS[7],在导航和机械臂的实验中取得了良好的效果。

针对这三点要求,着重对目标函数进行了构建,在已知状态 s 的条件下,下一个状态 s' 和当前策略 z 的互信息,公式表达为:

$$I(s';z \mid s) = H(z \mid s) - H(z \mid s',s) \tag{6}$$

$$I(s';z \mid s) = H(s' \mid s) - H(s' \mid s,z) \tag{7}$$

公式(6)中第一项表示在当前状态下选择不同策略的熵,值越大说明策略越多样,第二项表示当前状态 s 转移到下一状态 s' 的策略 z 的熵,这个值越大则说明很多策略都可以实现两个状态的转变,策略区分度很小。当两个值都很大时则说明策略很多,但策略的相似度很大,两个值都很小时则说明策略很少,策略区分度很大。所以前大后小是所需要的理想情况,能够实现既有较多的策略,同时状态的转换也不同。

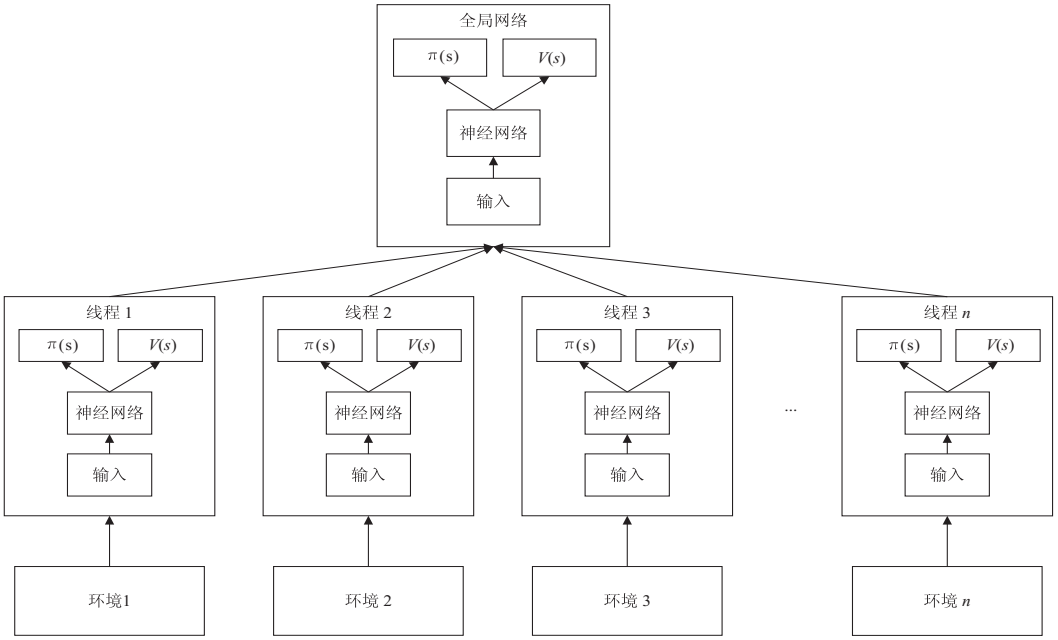

图 2 A3C 算法结构框图

公式(7)与公式(6)采用不同的计算方式,第一项表示状态转移的多样性,值越大则说明可转移的状态越多,第二项表示在当前状态 s 的情况下采取策略 z 所能到达下一状态的可能性,值越小则说明策略 z 对转移状态 s' 的确定性就越高,即策略的区分度也就越大。在保证前大后小的情况下,依旧可以实现互信息的最大化,使得策略的多样性和区分度有良好的保证。

MI-A3C 算法通过改进 A3C 算法来增强算法的探索与发现能力[6],不仅通过原本的收益函数来进行梯度的更新,同时通过加入互信息内容重新构造目标函数,联合价值函数来一同对 Actor 神经网络参数实现更新,如公式(8)所示:

$$\mathrm{d}\theta \leftarrow \mathrm{d}\theta + \nabla_{\theta}\log\pi_{\theta}(s_i,a_i)(Q(s,i) - V(S_i,w')) + c\nabla_{\theta}I(s';z \mid s) \tag{8}$$

神经网络的输入主要为当前环境状态 s_c 以及智能体 i 状态值 s_i ,随后神经网络通过训练的参数来输出对智能体采取的动作 a_t ,随后智能体与环境再次进行交互,同时在此基础上采用 option 分层[12-14]的思想,将动作进行抽象。将策略 z 集合定义为一个三元组 (A,s,s') , A 可以理解为抽象的动作序列是一段时间内多个动作集合, s,s' 是采取动作的先后的状态,根据动作在与环境的交互中,状态变化的程度来转换为策略,当前后状态变化的互信息较大时,就说明策略执行后产生的影响也较大,在对互信息进行优化时也就是对采取的策略进行不断调整,再加上奖励的变化,从而对整体网络的参数进行优化调整。

算法:ML-A3C 算法。

初始化公共部分神经网络参数 θ , ω ,全局共享迭代轮数 T ,最大迭代次数 $T_{\max}$,单线程内最大迭代次数 $t_{\max}$,随机策略

1. 更新时间序列 $t=1$

2. 重置 Actor 和 Critic 的梯度更新量: $\mathrm{d}\theta \leftarrow 0$, $\mathrm{d}\omega \leftarrow 0$

3. 将公共部分神经网路参数同步到本线程: $\theta' = \theta$, $\omega' = \omega$

4. $t_{\text{start}} = t$,初始化状态 s_t

5. 基于策略 $\pi(A \mid s,z)$ 选取动作

6. 执行新动作 A ,获得奖励 r_t 和新状态 s_{t+1}

7. 计算 $t = t+1$, $T = T+1$

8. 如果 s_t 是终止状态,或 $t - t_{\text{start}} = t_{\max}$,则进入步骤 9,否则

返回步骤 5

9. 计算最后一个时间序列位置 s_t 的 $Q(s,t)$

$$Q(s,t)=\begin{cases}0 & s_t \text{ 是最终状态}\\ V(s_t,\omega') & s_t \text{ 不是最终状态}\end{cases}$$

10. for $i \in (t-1,\cdots,t_{start})$

11. 计算每一时刻的 $Q(s,i)$: $Q(s,i)=r_i+\gamma Q(s,i+1)$

12. 根据公式(7)计算每一个状态的互信息

13. 累积 Actor 的梯度更新:

14. $d\theta \leftarrow d\theta + \nabla_\theta \log\pi_\theta(s_i,a_i)(Q(s,i)-V(S_i,w')) + c\nabla_\theta I(s';z \mid s)$

15. 累积 Critic 的梯度更新:

16. $d\omega \leftarrow d\omega + \frac{\partial[Q(s,i)-V(S_i,\omega')]^2}{\partial\omega'}$

17. End

18. 更新全局神经网络参数:

19. $\theta=\theta+\alpha d\theta$, $\omega=\omega+\beta d\omega$

20. 如果 $T>T_{max}$ 则算法结束,输出公共部分的 A3C 神经网络参数 θ , ω ,否则进入步骤 3

21. 结束

3　实验设计及结果分析

3.1　实验参数设置

实验硬件设置为 Intel SSD PEKKW 256G + NVIDIA Quadro RTX 5000 +32G 内存,软件环境为 windows10+TensorFlow2.0,超参数设置如表 1 所示。

表 1　超参数设置

超参数	MI-A3C	A3C
学习率 l	0.01	0.01
折扣系数 γ	0.95	0.95
线程数	10	10
神经网络层数	4	4
隐藏层节点数	500,300	500,300
更新系数 α	0.000 5	0.000 5
更新系数 β	0.001	0.001
权重系数 c	0.01	

3.2　实验仿真环境设计

实验仿真环境采用的是自主研发的深度强化学习仿真训练平台 DRL - STP,其程序是基于 TensorFlow2.0,采用了统一的算法框架,支持 Unity3D ml-agents 和 Gym,能够完成单智能体和多智能体的训练与测试。

采用的环境是基于 Unity3D 平台安装 ml-agents 插件构建的,用于多智能体智能对抗战场作战环境[15],具备坦克、装甲车、侦察车、火炮和武装直升机等多个智能体作战单元,作战元素图像如图 3 所示,作战单元属性如表 2 所示。作战环境地形包括平原、道路、桥梁、房屋、森林、灌木、丘陵、山脉等。战场地图是采用六边形地图(如图 4 所示),采用六边形地图约束每个作战单元每次可以移动的方向有 6 个,相对减少其移动动作的复杂度,地图大小为 64 * 64,其包含多个地形场景,作战单元双方可根据实际情况进行灵活设置,每个智能体除了采取移动的动作外,依旧具备攻击和原地待命的两个动作。

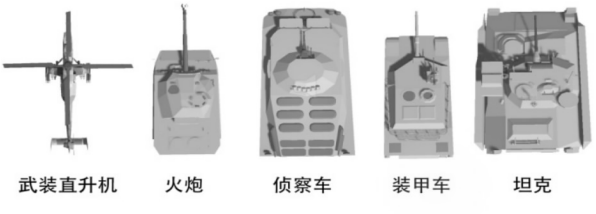

图 3　环境包含的作战元素

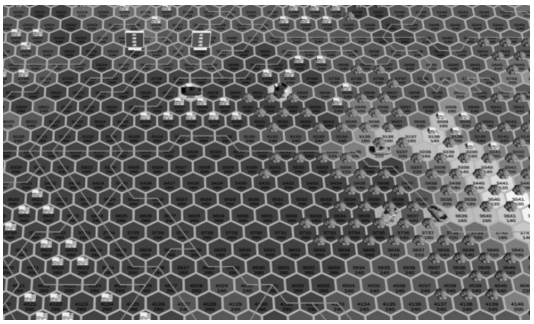

图 4　战场环境地图

智能体对周围环境的感知,其可以获取周围六个格子的通行状态和消耗的时间,对于敌方单位可以获取可视范围内能观察到的敌人数量,敌方智能体的距离所处地形,武器类型、生命值和是否可以攻击等状态,对于己方单位可以获取所有友方的数量、位置、武器类型和生命值,这一系列内容作为其神经网络的输入,来进行学习,智能体的移动方向、攻击状态、攻击目标作为输出值。

在对战双方进行交火时,也人工定义一些交战规则。对战双方出现在射程之内才可以采取攻击动作,在智能体移动结束后发现敌人并在攻击的有效范围即可以开火,同时根据攻击目标所处位置,攻击方击中被攻击者的概率会有所不同,空旷地域击中概率为 70%,在其他有遮蔽属性的地形中会减少击中概率,在表 2 和表 3 中射程与击中概率都有描述。同时针对智能体的攻击行为,也进行特定情况设定,如果炮管与目标夹角在 60°以内,消耗 2 个 T 完成射击,60° ~120°之间,消耗 3 个 T 完成射击,大于 120°,需要消耗 4 个 T 完成射击。火炮间瞄射击,需要 10 个 T 完成射击,所有 agent 在攻击期间不接受其他指令,这一系列规则是为了使环境能够更贴合现实战争中的场景。

算法与环境通过运用 ML-Agents 工具包中的 Python Low - Level API 和通信交互(External Communicator)这两个组件来实现与 python 训练算法的对接。Python Low-Level API 主要包含用于交互和

操纵学习环境的低级 Python 界面。与学习环境不同，Python API 不是 Unity 的一部分，而是在外面通过通信器与 Unity 沟通，此 API 包含在专用 Python 包 mlagents_envs 中，用于在训练期间的通信和控制。通信交互模块是将学习环境与 Python Low-Level API 连接起来，Python 首先通过环境参数模块获取仿真环境初始值，并获知了智能体的序号，将此序号发送给环境。Python 在接下来的正式训练之前，环境首先依据刚才的序号发送环境状态数据以及当前各个智能体可用的动作列表，Python 据此选择一个动作进入环境执行，环境执行并获取新的状态和奖励返回 Python，同时还有额外的敌我双方剩余兵力等信息，随后 Python 中的智能算法通过不断的信息交互，不断提升智能体策略规划能力。

表 2 作战单元属性情况

名称	最小射程	最大射程	观察距离	炮弹威力	生命值	限制
坦克	2	10	20	3	10	不能攻击空中目标
装甲车	1	8	20	2	10	无
侦察车	1	8	20	2	10	无
火炮	10	50	10	5	10	不能攻击空中目标
武装直升机	3	8	20	2	10	无

主要将两种作战任务作为目标，第一是摧毁所有敌方作战元素，在仿真环境中，击中敌人+1 分，消灭一个+5 分，被摧毁-5 分，全部歼灭敌人+15 分，全部被歼灭则-15 分；第二是占领夺控点，夺控点分为主要夺控点和次要夺控点，占领主要夺控点得 30 分，次要夺控点得 20 分，环境奖励设置如表 3 所示。

表 3 环境奖励设置

作战态势	具体情况	获得奖励
双方交火	击中敌方 1 个单位	+1
	消灭敌方 1 个单位	+5
	被摧毁 1 个己方单位	-5
	歼灭全部敌方单位	+15
	己方单位全被歼灭	-15
占领夺控点	占领主要夺控点	+30
	占领次要夺控点	+20

同时为了使训练的每一个动作都是有效的，不会出现智能体为了避免惩罚而原地不动的情况，通过公式(8)设置奖励会随时间变化而逐渐减少，每一局进行的时间越长奖励会成比重的进行相应的扣除：

$$R = \sum R_i - \sqrt[3]{T} \tag{8}$$

在实验中智能体分为蓝方和红方，蓝方智能体的战术规则，主要通过人工定义。蓝方规则的设定对于红方的训练效果起着重要的作用。当蓝方规则智能体的规则较为完善时，具有的战斗能力就越强，这样可以通过训练使红方智能体具有更优秀的策略规划能力，但是训练的难度也会变大，训练时间也会更长；另一种情况当蓝方规则智能体的规则较为简单时，蓝方具有的战斗力会变弱，相对的红方训练的难度会变小，训练时间会更短，但是训练出的战术策略没有什么特别的价值。所以针对这种情况，在对规则的构建时，首先保证了蓝方智能体具有较强的对抗强度，但在开始训练的一段时间里，减少蓝方智能体的进攻性，给红方智能体探索学习的时间，在红方智能体能够开始获得有效收益后，加强进攻性以及对抗强度，增强红方智能体所学习到策略的有效性。

3.3 实验结果与对比分析

分别使用 MI-A3C 算法、A3C 算法以及经典 DQN 算法在相同场景下进行对比实验。实验主要采用平均回报(Average Reward)来评价算法效能，平均回报越高则算法效能越好。

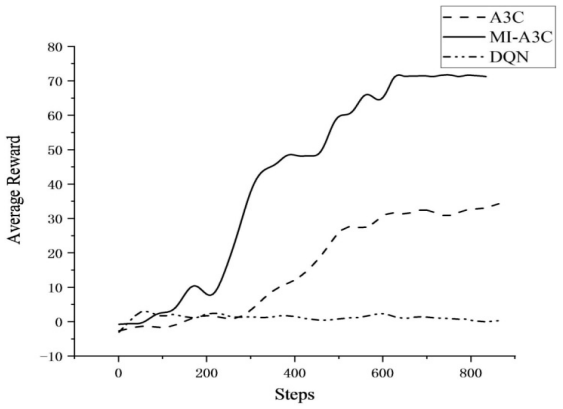

图 5 算法收益曲线对比

图 5 展示了不同算法的平均收益。为了更清楚地表现智能体收益曲线的增长，对数据图像进行平滑处理，展示了采用不同算法的智能体在对抗任务中收益曲线对比。根据图像我们可以明显看出，在 200 轮的训练后，MI-A3C 算法的平均回报提升速度相比 A3C 算法有了显著的提升，这也就表明红方智能体策略逐渐学到了能够有效获取奖励的策略，能够逐渐有效摧毁敌方单位并开始前往占领夺控点，并可以发现其随后开始了一个收益快速增长的阶段，而 A3C 算法也在逐渐增长但增长速度相对较为缓慢，通过对实验环境

的观察发现,这一阶段的增长是智能体学习到了迅速到达某一夺控点的策略,其能够迅速获得占领夺控点的收益,而 A3C 算法由于其探索效率的问题,相对学到这一策略的速度更慢。DQN 算法收益曲线相对较低,通过观察其训练场景发现,其在最初阶段能够逐渐学习到前往夺控点,但无法学习到摧毁敌方元素的有效策略,甚至始终无法找到第二个夺控点,而敌方单位是规则操控对夺控点进行占领以及攻击,面对此情况采用 DQN 算法的智能体经常损失惨重,始终没有学习到有效应对的方式,传统的深度学习方法在此种对抗场景下效果较差。

从最终的收益可以看出 MI-A3C 算法相较 DQN 和 A3C 算法都有较大的优势,其最终收益能稳定到 65 左右,能够占领两个夺控点,并歼灭所有敌方元素,自身损失 1 到 2 个单位,可以较好地完成所有任务。因此可以得出结论,ML-A3C 算法在智能指挥控制上,能够通过增加智能体的探索性,获得更多的技能来提升智能对抗的效能。

通过观察到 MI-A3C 算法收益曲线发现,智能体在某一时刻会产生收益下降的情况,通过分析实验情况发现,当出现收益下降的时刻是智能体学到了一种策略,但此种策略在此刻并未立即获得收益,例如集群行动,随后的动作因为策略是新的,所以之后智能体的动作会与之前的不同,也就会出现无法更好完成任务的情况,于是出现收益下降。在经过长时间的学习后,智能体学会如何更好地采取这一策略后,任务完成情况就会顺利,收益也随之上升,这样也证明了学到的策略是有效的。

表 4　稳定后不同算法的实验数据

算法	平均红方得分	红方胜率/%	平均蓝方得分
MI-A3C	64.5	86.7	34.3
A3C	33.5	34.9	40.2
DQN	1.2	0	62.4

4　结束语

针对智能对抗环境下的智能决策问题,提出了一种基于互信息策略发现的 A3C 改进算法 ML-A3C。该算法通过互信息使得智能体能够通过状态的变化来发现更多具有长远收益的策略,同时采用 option 分层的方式,将策略 z 动作进行划分,并将其互信息与神经网络参数进行互联。最终通过实验证明,对智能体稳定后的数据进行了统计(见表 4),通过训练的智能体在对抗实验中能够实现更有利的策略,来保存己方兵力,摧毁敌方单元。MI-A3C 算法可以为复杂环境多兵种联合对抗进行仿真模拟,为智能指挥决策提供理论依据,对现实战场中的应用具有一定的参考意义。

但该算法也存在一定的局限性,算法的仿真试验平台所开发的地图环境能够较好地贴近现实地图环境,但实验的元素依旧较少,属于小分队形式的战场对抗,MI-A3C 算法在小规模的战场中能够取得良好的效果,但并未进行大规模的战场模拟试验,所以对大规模的对战能否取得良好的效果仍需进一步进行证明,同时算法增强了对环境探索的能力,当环境复杂度较高时,学习到有效策略的时长会增加。

参考文献:

[1] SUTTON R S, BARTO A G. Reinforcement learning: an introduction[M]. Cambridge MA: MIT Press, 1998.

[2] 刘　全, 翟建伟, 章宗长. 深度强化学习综述[J]. 计算机学报, 2018, 41(1): 1-27.

[3] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement learning[J]. arXiv: 1312.5602, 2013.

[4] LILLICRAP T P, HUNT J, PRITZEL A, et al. Continuous control with deep reinforcement learning[J]. arXiv: 1509.02971, 2015.

[5] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms[J]. arXiv: 1707.06347, 2017.

[6] EYSENBACH B, GUPTA A, IBARZ J, et al. Diversity is all you need: Learning skills without a reward function[J]. arXiv: 1802.06070, 2018.

[7] SHARMA A, GU S, LEVINE S, et al. Dynamics-aware unsupervised discovery of skills[J]. arXiv: 1907.01657, 2019.

[8] HOUTHOOFT R, XI C, YAN D, et al. VIME: variational information maximizing exploration[J]. arXiv: 1605.09674, 2016.

[9] 沈　晶, 顾国昌, 刘海波. 分层强化学习研究综述[J]. 模式识别与人工智能, 2005, 18(5): 574-581.

[10] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning[J]. arXiv: 1602.01783, 2016.

[11] 范雪莉, 冯海泓, 原　猛. 基于互信息的主成分分析特征选择算法[J]. 控制与决策, 2013, 28(6): 915-919.

[12] 沈　晶, 顾国昌, 刘海波. 分层强化学习中的 Option 自动生成算法[J]. 计算机工程与应用, 2005, 41(34): 4-6.

[13] SUTTON R S, PRECUP D, SINGH S. Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning[J]. Artificial Intelligence, 1999, 112(1-2): 181-211.

[14] 祖丽楠, 田彦涛, 梅　昊. 基于分层强化学习的多移动机器人避障算法[J]. 吉林大学学报: 工学版, 2006(S2): 108-112.

[15] 李　航, 刘代金, 刘　禹. 军事智能博弈对抗系统设计框架研究[J]. 火力与指挥控制, 2020, 45(9): 116-121.