计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2023,32(4):293−299 [doi: 10.15888/j.cnki.csa.009015]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 BN-DDPG 轻量级强化学习算法的智能兵棋推演①

李卓远, 张德平

(南京航空航天大学 计算机科学与技术学院, 南京 211106)
通信作者: 李卓远, E-mail: 408163489@qq.com

摘　要: 兵棋推演与智能算法融合成为当前军事应用领域的研究热点, 利用深度强化学习技术实现仿真推演中决策过程的智能化, 可显著减少人为经验对决策过程的影响, 提高推演效率和灵活性. 现有基于 DRL 算法的决策模型, 其训练时间过长, 算力开销过大, 无法满足作战任务的实时性需求. 本文提出一种基于轻量级深度确定性策略梯度 (BN-DDPG) 算法的智能推演方法, 根据推演规则, 采用马尔可夫决策过程描述推演过程中的决策行为, 以 actor-critic 体系为基础, 构建智能体训练网络, 其中 actor 网络使用自定义混合二进制神经网络, 减少计算量; 同时根据经验样本的状态和回报值建立双缓冲池结构, 采用环境相似度优先提取的方法对样本进行采样, 提高训练效率; 最后基于自主研制的仿真推演平台进行实例验证. 结果表明, BN-DDPG 算法可简化模型训练过程, 加快模型收敛速度, 显著提高推演决策的准确性.

关键词: 智能推演; 深度强化学习; 二值神经网络; 自主决策

引用格式: 李卓远,张德平.基于 BN-DDPG 轻量级强化学习算法的智能兵棋推演.计算机系统应用,2023,32(4):293–299. http://www.c-s-a.org.cn/1003-3254/9015.html

Intelligent Wargame Deduction Based on BN-DDPG Lightweight Reinforcement Learning Algorithm

LI Zhuo-Yuan, ZHANG De-Ping

(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

Abstract: The integration of wargaming and an intelligent algorithm has become a research hotspot in the field of military application. Using deep reinforcement learning (DRL) to realize the intellectualized decision-making process in simulation deduction can significantly reduce the impact of human experience on the decision-making process and improve deduction efficiency and flexibility. Limited by its long training time and high computational cost, the existing decision-making model based on the DRL algorithm cannot meet the requirement of combat tasks for real-time performance. This study introduces an intelligent deduction method based on the lightweight binary neural network-deep deterministic policy gradient (BN-DDPG) algorithm. According to deduction rules, the Markov decision process is used to describe the decision behavior during deduction. Relying on the actor-critic system, an agent training network is constructed, in which the actor network uses a custom hybrid binary neural network to reduce the amount of calculation. At the same time, a double-buffer-pool structure is built according to the status and return value of empirical samples, and sampling is performed by the method of priority extraction of environmental similarity for higher training efficiency. Finally, an example is verified on a self-developed simulation deduction platform. The results show that the BN-DDPG algorithm can simplify the model training process, accelerate the convergence of the model, and significantly improve the

① 基金项目: 国防基础科研基金 (JCKY2020605C003)
收稿时间: 2022-08-25; 修改时间: 2022-09-27; 采用时间: 2022-10-08; csa 在线出版时间: 2023-03-17
CNKI 网络首发时间: 2023-03-19

accuracy of deduction and decision-making.

Key words: intelligent deduction; deep reinforcement learning; binary neural network; autonomous decision-making

兵棋推演技术在各个作战领域都有非常广泛的应用场景. 通过兵棋推演系统, 可以推测在一些应用场景下, 采取不同的行动而产生的后续作战结果, 根据结果的好坏, 对所作决策进行适当调整[1]. 这种调整过程异常繁琐, 当场景环境复杂多变时, 所考虑的决策数量巨大, 导致推演需要消耗大量的人力物力, 最终决策结果一定程度上也无法满足作战要求.

深度强化学习算法作为一种常见的智能探索算法, 已经开始应用于仿真推演领域. 深度强化学习算法应用于仿真推演智能化具有两大优势.

(1) 利用深度学习算法 (DL) 对战场感知数据进行充分的分析处理, 可以帮助作战人员快速判断当前战况.

(2) 在强化学习算法 (RL) 的帮助下, 作战人员能够更有效的进行决策, 对战争结果产生重要的影响.

利用深度强化学习算法探究作战最优策略逐渐成为兵棋推演智能化的一个重要研究热点.

邓克波等[2] 在强化学习的基础上提出了面向作战方案分析的计算机兵棋推演系统. 殷宇维等[3] 改进了 DDPG 算法并将其应用到空战决策中, 使决策过程更加平稳可控. 王兴众等[4] 提出一种基于 SAC 算法的仿真推演决策技术, 提高了决策结果的获胜率.

现有基于 DRL 算法的众多决策模型, 其训练时间过长, 算力开销过大, 无法满足作战任务的实时性需求.

在深度强化学习中, 神经网络的复杂性是影响训练速度的重要因素之一, 网络中众多的参数会增加模型的训练负担, 耗费大量的训练时间. 针对此问题, Li 等[5] 提出了将 BNN 应用在 DQN 算法中, 并取得了较好的加速效果, 虽然 BNN 减少模型训练的计算量, 但是对特征的提取能力降低, 导致在一些训练中, 模型的最终效果不理想. 在 off-policy 强化学习算法中, 经验回访缓冲池用来保存经验样本数据, 然后随机采样历史数据更新深度神经网络的参数, 采用合适的采样策略对缓冲池内的数据集进行采样, 可以加速训练的过程, Schaul 等[6] 提出了优先级经验回放的思想, 将经验数据的 TD-error 作为采样的优先级对数据进行优先级采样, 加快了 DQN 模型的收敛速度. Hou 等[7] 又将这种思想从离散领域的 DQN 算法扩展到了连续领域的 DDPG 算法, 并认为这种方法能够明显缩短模型的训练时间, 增强模型训练时的可靠性. 张建行等[8] 提出了一种基于情节的双缓冲池结构, 提高经验采样的有效性.

尽管这些研究在一定程度上提高了模型训练效率, 但是训练过程中的操作会使训练中间信息丢失, 导致最终模型精度下降, 训练结果无法满足实际作战需求. 针对此问题, 本文提出一种基于轻量级深度确定性策略梯度 (BN-DDPG) 算法的智能推演方法, 使用混合 BNN 网络作为 DDPG 的 actor 网络作为训练加速的方法, 在减少计算量的基础上保证网络的特征提取能力; 同时采用基于环境相似度的优先提取方法, 对缓冲池中的历史经验数据进行采样, 加快训练时的收敛速度. 最后, 使用自主研制的仿真推演平台进行实例验证, 以轰炸机轰炸目标机场为任务想定, 验证提出算法的有效性.

1 相关技术

1.1 DDPG 经验回放缓冲机制

DDPG 算法[9] 是 DQN 算法的扩展版本, 在连续控制领域有很好的表现[10], 采用目标网络和经验回放的技巧, 基于 actor-critic 体系结构, actor 网络和 critic 网络相互独立.

经验回放缓冲池[9] 是 DDPG 中的信息库可以将模型探索环境得到的经验数据储存起来, 然后通过采样这些数据对网络模型进行训练, 最终过去的 experience 和目前 experience 混合, 从而减少了信息相关性.

DDPG 算法作为一个 off-policy 离线学习法, 能同时对当前和过去的经验数据进行学习, 在学习过程中可以随机增加之前的经验数据, 这会使神经网络更加高效. 经验池的出现, 成功解决了相关性及非静态分布问题, 每个 time step 下 agent 与环境交互得到的转移经验数据 (s_t, a_t, r_t, s_{t+1}), 然后保存到回放的记忆网络上, 可以按照一定的方式提取这些数据用于训练, 打乱数据之间的相关关系[11].

1.2 二值神经网络

二值神经网络 (BNN)[12] 把权值 W 和隐藏层激活

值二值化为 1 或者−1. 通过这种操作, 使模型的参数占用更小的存储空间, 存储器能耗在理论方面上减少为原来的 1/32, 从 float32 到 1 bit; 同时利用位操作来取代了乘加运算, 从而大大降低了计算成本.

关于怎么把浮点型的神经网络进行二值化, 一般给出两种方法.

(1) 基于符号变量 *sign* 的定义方法, 大于 0 就为 +1, 小于 0 则为−1, 如式 (1):

$$x^b = sign(x) = \begin{cases} +1, & x \geqslant 0 \\ -1, & x < 0 \end{cases} \tag{1}$$

(2) 第 2 种是随机二值化方法:

$$x^b = \begin{cases} +1, & \text{with probability } \rho = \sigma(x) \\ -1, & \text{with probability } 1-\rho \end{cases} \tag{2}$$

BNN 计算梯度的方式, 是通过使用二值化的权值和激活值, 这里的梯度必须是高精度的实际值, SGD 在计算梯度的时候, 量级一般比较小, 再加之计算的中间过程会有服从正态分布的噪声, 所以如果算子没有较高的精度会影响模型最后的实际效果. 另外, 为帮助避免过拟合现象的发生[13], 可以在计算梯度的时候, 给二值化后的权值和激活值添加噪声.

2 BN-DDPG 模型

BN-DDPG 模型将 DDPG 算法作为主体框架, 将 get-actor 当前网络设计为混合 BNN 网络, 缓冲池结构设计为基于环境相似度的双缓冲池结构.

2.1 BNN 模块

传统的 DRL 算法中, 不管是在单智能体系统或是多智能体系统, 都是在强化学习算法的基础上, 通过神经网络拟合 Q 函数, 利用深度学习网络的特征提取能力, 将 Q 值函数尽可能优化, 大部分使用的是 DNN、CNN 或者 LSTM 等网络, 虽然这些网络虽然具有强大的学习能力, 但也要更根据研究目的或应用场景合理设计网络结构, 例如卷积核大小和网络深度等, 才能发挥出更有效的作用. 对于 CNN[14] 来讲, 单个卷积层的时间复杂度为:

$$\text{Time}: \ \mathrm{O}\left(M^2 \times K^2 \times C_{\text{in}} \times C_{\text{out}}\right)$$

其中, M 表示卷积核输出的特征图的边长, K 则表示各个卷积核的边长, C_{in} 表示各个卷积核的通道数. C_{out} 表示在当前卷积层卷积核的总个数.

卷积神经网络整体的时间复杂度为:

$$\text{Time}: \ \mathrm{O}\left(\sum_{l=1}^{D} M_l^2 \times K_l^2 \times C_{l-1} \times C_l\right)$$

其中, D 表示在神经网络中的卷积层数, 也就是网络的深度. CNN 网络的整体复杂度为各个卷积层的复杂度的累加之和. 如此的复杂度通常是不适用于嵌入式电子设备等小型设备, 这限制了深度强化学习在一些急迫任务中应用真实世界.

直观来讲, 减少复杂度最容易想到的方式是减少的网络的深度和宽度. 不过这种操作会带来一个很大的缺点: 容易出现欠拟合. 当我们直接对网络的基础参数进行消减后, 网络的性能一定会在不同程度上受到影响, 从导致网络不能很好拟合函数的情况发生. Li 等[5] 将 BNN 使用到了 DDPG 中并完成了对星际争霸游戏的训练加速, 其使用的神经网络层全部是二值神经网络层如图 1 所示, 这在提取特征时虽然会大幅度减少计算量, 但特征提取能力下降的过快, 我们重新设计使用轻量级的神经网络 BNN 来拟合 DDPG 中的 actor 网络, 卷积层使用二值网络和全精度网络交替提取特征, 在尽可能保留了神经网络强大的特征提取能力的同时, 将其复杂度极大简化.

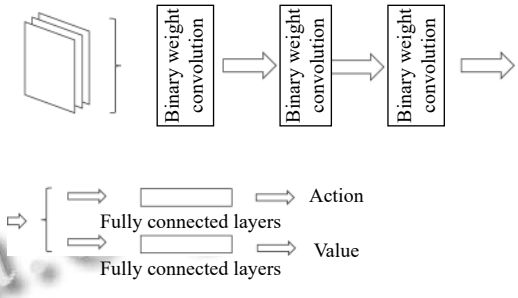

图 1 卷积层全部二值化的 BNN 网络

对于混合 BNN 网络的设计, 采用 3 层神经隐藏层的结构, 为了保证网络的特征提取质量, 第 1 层使用全精度网络层, 第 2、3 层采用二值网络层, 降低计算量以及复杂度, 混合 BNN 具体网络结构见图 2.

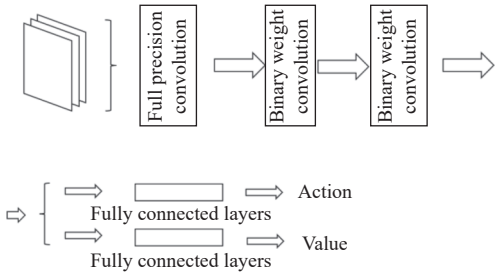

图 2 卷积层二值化和全精度的混合 BNN 网络

2.2 双缓冲池模块设计

传统 DDPG 算法[9] 中, 经验回放采用单缓冲池结构, 模型会将采集得到的数据不加以区分全部放入单个缓冲池当中, 这样做操作简单, 但是忽略了不同的历史数据对模型的训练所起到的效果并不相同, 针对这一问题, Schaul 等[6] 提出了缓冲池优先级经验回放机制, 并应用在 DQN 中, 这种方法可以使重要的经验被回放的概率变大, 从而使学习更有效率, 并在一些实验中取得了较好的结果. 具体的该机制使用 TD-error 来表示优先级的大小, 当前 Q 值与目标 Q 值两者之间差值的大小, 计算方式如式 (3), TD-error 的值越高, 就反映了预测效果不好, 还需要进一步提升, 此样本数据就越值得被学习, 也就是该样本数据的优先级 p 越高.

$$\delta = R_t + \gamma_t \max Q(S_t, a) - Q(S_{t-1} + A_{t-1}) \tag{3}$$

其中, R_t 是前 t 步所获得的累计回报, γ 是折扣率. 当 reward 有噪声时, TD-error 来估计优先级不稳定, Schaul 等[6] 又补充了 greedy TD-error prioritization 算法, 在经验池中存储了每轮交互最后的 TD-error, 用这种方式, 将会以 TD-error 的绝对值最大的进行回放. 如果当一个新的 transition 到来时, 不知道它的 TD-error, 那么就把这个 transition 的 TD-error 值设置为最大, 这样可以保证所有的经验都会被至少回放一次.

为了提高经验回放机制在 DDPG 中的效率, 张建行等[8] 提出一种基于情节经验回放的深度确定性策略梯度方法 (EEP-DDPG), 将样本数据以情节为单元进行保存, 再按照情节回报的情况通过两个缓冲池分类存放. 最后, 网络训练阶段着重选择累积回报较大的数据进行采样, 以提高训练效率.

以上对缓冲池的改进主要思路都集中在使用较多的回报值较大的经验数据来提升经验回放的效率. 本文从每条经验数据的环境状态出发, 设计优先提取机制的双缓冲池结构.

在挑选经验数据时, 与当前训练的环境状态相似的经验数据有更大的可选取性和参考价值, 根据环境状态的相似程度, 将不同的经验数据存放在不同的数据缓冲池中. 具体的, 新增超参数 distance, 作为经验数据状态相似程度的区分, 在开始收集数据时, 如果两条经验数据状态差的绝对值小于 distance, 则认为两条经验数据环境相似, 放入相同的经验池中, 否则放入不同的经验池中. 两个经验池分别维护一个变量 x_i, 记录第 i 个经验池内存放数据状态的平均值, 计算方式如式 (4):

$$x_i = \frac{s_{i1} + s_{i2} + s_{i3} + \cdots + s_{in}}{n} \tag{4}$$

其中, i 表示第 i 个经验回放池, n 为该经验池目前保存的经验数据的个数. 在经验回放阶段, 使用此时的环境状态与每个经验池的平均状态作比较, 在差值更小的经验池内, 取出所需数据总数的 70%, 在另一个经验池内取出所需数据总数的 30%, 并把这一步所获得的经验数据加入到差值更小的经验池内, 并更新该回放池的 x, 更新方式如式 (5):

$$x_i^{t+1} = \frac{n \times x + s^{t+1}}{n+1} \tag{5}$$

其中, x 是第 i 个经验回放缓冲池未更新前, 经验回放缓冲池环境状态的平均值, $s^{(t+1)}$ 为 t+1 时刻智能体的环境状态. 利用相似环境下数据的高参考性, 提升经验回放的效率. 具体操作如图 3.

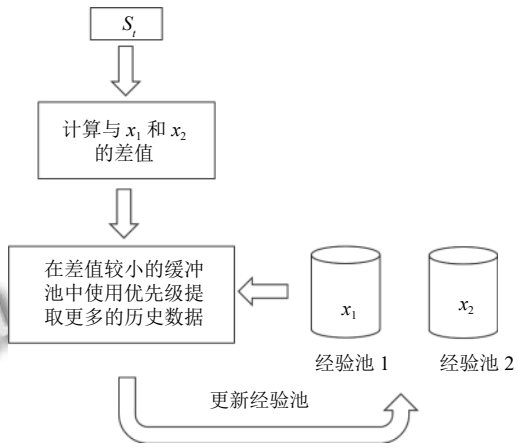

图 3 基于环境状态的双经验池结构

2.3 BN-DDPG 算法

BN-DDPG 算法整体基于 DDPG 算法, 因此两者流程类似. 算法输入为: actor 当前网络, actor-target 网络, critic 当前网络, critic-target 网络, 参数分别为 θ, θ', ω, ω', 折扣因子 γ, 软更新系数 τ, 批量梯度下降的样本数 m, target-Q 网络参数的更新频率 C, 训练的最大迭代次数 T, 环境状态差异值 distance; 算法的输出为: 训练后的 actor 网络参数 θ 和 critic 网络参数 ω.

算法的具体流程见算法 1.

算法 1. BN-DDPG 算法

1) 随机初始化θ, ω, $\omega'=\omega$, $\theta'=\theta$. 清空两个经验回放缓冲池 D1, D2;
2) for i from 1 to T, 进行迭代;
a) 初始化S, 将其作为状态序列的初始状态, 并得到其特征向量$\varphi(S)$;
b) 在 actor 当前网络基于状态S得到动作 $A=\pi_\theta(\varphi(S))$;
c) 执行步骤 b) 中得到的动作 A, 在与环境交互后获得新状态S', 及其相应的奖励 R, 并判断是否终止状态;
d) 如果 D1、D2 均没有数据, 那么将四元组$\{\varphi(S), A, R, \varphi(S')\}$随机放入一个缓冲池, 如果缓冲池有数据, 则分别计算目前状态S与两个缓冲池环境均值之差的绝对值, 将四元组放入数值较小的缓冲池中, 并更新该缓冲池的环境均值;
e) $S=S'$;
f) 根据相似原则, 从相似度高的缓冲池使用优先级策略选取 $N1$ 个数据集, 从另一个缓冲池选取 $N2$ 个数据集, $N1>N2$;
g) 计算当前目标 Q 值;
h) 更新 critic 当前网络的所有参数w;
i) 更新 actor 当前网络的所有参数θ;
j) 更新 critic-target 和 actor-target 参数;
k) 对 S'进行判断, 如果 S'满足中止条件, 本轮迭代完毕, 如果 S'不满足终止条件, 则跳转到步骤 b);

3 仿真推演实例分析

3.1 推演任务想定设计

搭载空对地导弹轰炸机对目标进行战略轰炸推演想定的场景如下: 蓝方部队在某地有威胁红方的军用机场, 蓝方在机场周围部署有拦截导弹车对空中威胁进行打击, 并有侦察机监视一定区域内的敌方威胁, 一旦侦察机发现威胁, 导弹车会发射导弹对目标进行拦截. 红方派遣轰炸机前往搜寻蓝方的机场, 并对其进行轰炸. 红方轰炸机要做的就是尽可能避开蓝方侦察机的侦察, 并以最快的速度完成轰炸任务. 红蓝双方的设定的兵力编成如表 1 和表 2 所示.

表 1 红方军队兵力编成

单元类型及名称	速度 (km/h)	数量	单元主要武器
MH-60 “海鹰”轰炸机-001	200	1	MK-54轻型鱼雷×10
MH-60 “海鹰”轰炸机-002	200	1	MK-54轻型鱼雷×10

表 2 蓝方军队兵力编成

单元类型及名称	速度 (km/h)	数量	单元主要武器
战略机场-001	0	1	无
HHK-75 “风神”导弹车	0	1	SS-N-15 “海星”火箭弹×10
955A北侦察机	210	1	无

3.2 红方轰炸机建模

在第 3.1 节的想定任务下, 对红方轰炸机进行强化学习训练, 通过与环境的不断交互, 不断优化轰炸机的决策策略, 最终完成想定任务. MDP 过程以四元组 (s, a, s', r) 的形式表示, 这里 s 为红方轰炸机状态空间, a 为红方轰炸机的动作决策空间, s'为采取动作之后轰炸机转移到的状态空间, r 是此次状态转移所获得的回报.

1) 状态空间设置

红方轰炸机返回的态势数据维度较多, 其中执行任务时的最关键因素有 5 个, 分别是经纬度、海拔、航向以及存货状态, 所以状态空间确定为 S=[经度, 纬度, 海拔, 航向, 存活状态]. 如图 4 所示, 经纬度和海拔可以锁定当前轰炸机的所在位置, 航向 α 表示红方轰炸机与蓝方目标机场的偏差角度, 存活状态 status 表示目前飞机是否被击落.

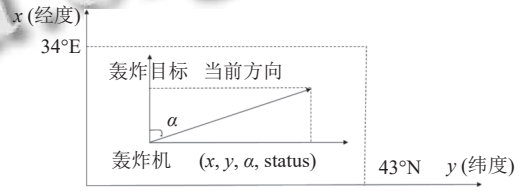

图 4 红方轰炸机状态分析

2) 动作空间设置

红方轰炸机可执行的动作也相对较多, 动作空间选择最能影响任务成败的动作. 在该任务中, 轰炸机的方向即航向角, 决定了轰炸机是否能顺利绕过侦察机, 抵达机场附近执行轰炸任务, 因此选择轰炸机航向角作为动作空间, 即 A=[航向].

3) 奖励函数设置

回报函数的设置如下: 红方轰炸机朝向目标机场移动, 此次移动如果使得轰炸机更接近机场, 获得正回报, 反之获得负回报. 若轰炸机与机场方向的偏向角在 0 到 180°之间, 获得正回报, 若轰炸机与机场夹角在 180° (包含 180°) 到 360°之间, 获得负回报. 若机场进入轰炸机攻击范围, 获得一个较大的正回报. 若轰炸机成功摧毁机场, 获得一个更大的正回报. 若飞机被摧毁 (status 状态为 0), 轰炸机获得一个负回报.

将网络模型推算的动作列表作为推演系统的输入, 并在推演系统中完成执行该动作后各单位状态的改变, 以获得的新状态, 判断红方轰炸机在此动作下所应该获得的回报, 具体计算公式如式 (6) 所示:

$$R_t=\begin{cases}+100, & \text{轰炸机摧毁机场}\\ -80, & \text{轰炸机被击落}\\ -50, & \text{轰炸机飞出指定区域}\\ \rho(\alpha,dis,t), & \text{其他}\end{cases} \tag{6}$$

其中, dis 表示轰炸机与机场在推演开始的原始距离, dis_t 表示 t 时刻机场距离轰炸机的位置, dis_{t-1} 表示在 t–1 时刻, 机场距离轰炸机的位置, α表示当前轰炸机与机场的偏向角如图 4.

函数$\rho(\alpha,dis,t)$为:

$$\rho(\alpha,dis,t)=\frac{[(20\times\sin\alpha)+(dis_t+dis_{t-1})]}{dis}$$

4 仿真推演实验及分析

在第 3.1 节介绍的仿真任务下, 在实验环境中开展仿真推演, 实验的第 1 部分通过与连续 DQN 算法进行对比实验 BN-DDPG 算法的优越性; 实验的第 2 部分通过消融实验, 验证改进部分对于模型整体的贡献.

4.1 推演结果分析

本文的实验环境基于团队开发的智能推演系统. 在轰炸机执行任务的开始阶段, 需要不断试错寻找机场位置并避开侦察机的侦察, 因此开始阶段回报相对较低. 经过不断的迭代, 回报值最终可以趋于平稳. 在不断地学习下, 轰炸机能够不断减少发现机场所用的时间, 最终成功执行任务. BN-DDPG 训练时的超参数设置如表 3 所示.

表 3 超参数设置

超参数	数值
折扣率γ	0.96
经验池大小	10 000
actor网络学习率	0.000 1
critic网络学习率	0.000 1
actor更新间隔	5
批大小	56
最大训练的episode (个)	5 000
初始状态差异distance	50

将 BN-DDPG 算法与连续 DQN 进行实验对比, 对比结果如图 5 所示.

训练初期, BN-DDPG 和连续 DQN 的平均回报均上升的较快, 利用连续 DQN 算法训练智能体, 在经过 1 000 轮训练后开始收敛, 平均回报值也趋于稳定; 而使用 BN-DDPG 算法收敛的速度相对更快, 从实验结果中可以看出, BN-DDPG 算法在迭代 850 轮左右就已经趋于稳定, 且 BN-DDPG 的收敛后的平均回报略大于连续 DQN 算法. 两种算法模型训练完毕后, 分别对其进行 100 次的仿真推演实验, 两种算法的获胜概率的比较分析结果如表 4 所示, 采用连续 DQN 算法, 红方轰炸机的平均获胜概率是 50.5%; 采用 BN-DDPG 算法, 红方轰炸机获胜概率为 58.4%, 提高了 8 个百分点左右. 图 6 为 DQN 算法与 BN-DDPG 算法收敛轮数对比的箱线图.

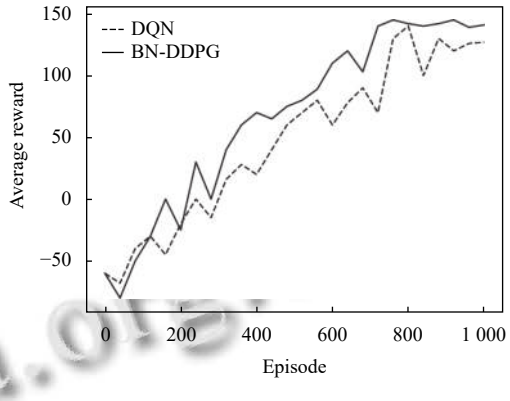

图 5 BN-DDPG 算法与连续 DQN 算法平均回报对比

表 4 红方轰炸机获胜概率对比 (%)

算法	稳定胜率
连续DQN	50.5
BN-DDPG	58.4

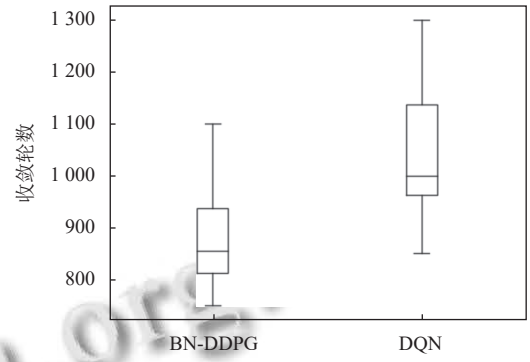

图 6 BN-DDPG 与连续 DQN 收敛轮数对比

4.2 消融实验

为检验 BN-DDPG 中提出的改进方法的有效性, 同时为分别研究算法中的混合二进制网络和基于环境状态相似度的双缓冲池结构这些改进机制对算法实验效果的影响, 在 BN-DDPG 算法的基础上分别减去这些机制进行消融对比试验. 消融实验的设置如表 5 所示, 消融实验结果如图 7. 红方轰炸机获胜概率对比如表 6, 平均收敛轮数对比如图 8 所示.

表 5 消融对比实验设置

实验算法	混合二进制网络	基于环境状态的双缓冲池结构
DDPG	×	×
BN-DDPG-A	√	×
BN-DDPG-B	×	√
BN-DDPG	√	√

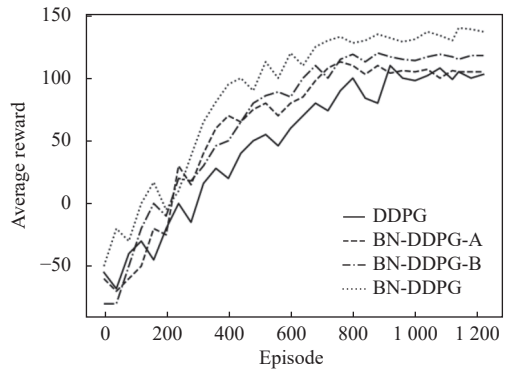

图 7 消融实验平均回报对比

表 6 红方轰炸机获胜概率对比 (%)

算法	稳定胜率
DDPG	51.5
BN-DDPG-A	55.4
BN-DDPG-B	54.0
BN-DDPG	60.4

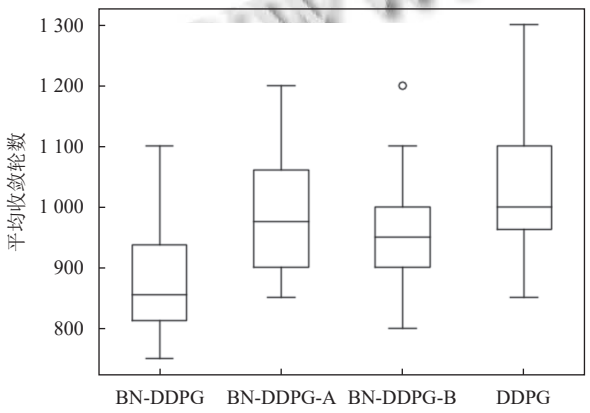

图 8 消融实验平均收敛轮数对比

实验结果表明, 传统的 DDPG 算法在该任务中, 训练 1 000 轮之后基本可以收敛; 分别采用混合二进制神经网络和基于环境状态的双缓冲池结构对 DDPG 进行优化, 可以在 800 轮左右稳定收敛; 采用 BN-DDPG 算法训练该任务时, 在训练到 700 轮左右, 即可稳定收敛, 并且收敛后的平均回报值明显大于传统 DDPG 算法, 这也说明了我们所提供的优化方案, 相比于常规 DDPG 算法不管从有效性还是收敛效率方面, 都有了一定的提高.

5 总结与展望

本文在 DDPG 算法的基础上, 使用混合二进制神经网络和基于环境状体的双缓冲池结构对算法进行优化, 并将其应用在自定义的推演作战任务中. 从实验结果可以看出, 新提出的模型在收敛速度和平均回报上均优于原始模型, 通过消融实验证了模型优化部分的有效性. 在作战任务中, 大部分情况下是作战单元的协同作战, 下一步工作将着重探索将该优化方式应用于多智能体算法, 对推演的协同作战任务策略制定进行优化.

参考文献

1 戴勇, 黄杏花. 人工智能在计算机兵棋推演领域的应用. 集成电路应用, 2020, 37(5): 67–69.
2 邓克波, 朱晶, 韩素颖, 等. 面向作战方案分析的计算机兵棋推演系统. 指挥信息系统与技术, 2016, 7(5): 73–77.
3 殷宇维, 王凡, 吴奎, 等. 基于改进 DDPG 的空战行为决策方法. 指挥控制与仿真, 2022, 44(1): 97–102. [doi: 10.3969/j.issn.1673-3819.2022.01.014]
4 王兴众, 王敏, 罗威. 基于 SAC 算法的作战仿真推演智能决策技术. 中国舰船研究, 2021, 16(6): 99–108.
5 Li YF, Fang YC, Akhtar Z. Accelerating deep reinforcement learning model for game strategy. Neurocomputing, 2020, 408: 157–168. [doi: 10.1016/j.neucom.2019.06.110]
6 Schaul T, Quan J, Antonoglou I, *et al*. Prioritized experience replay. arXiv:1511.05952, 2015.
7 Hou YN, Liu LF, Wei Q, *et al*. A novel DDPG method with prioritized experience replay. Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics. Banff: IEEE, 2017. 316–321.
8 张建行, 刘全. 基于情节经验回放的深度确定性策略梯度方法. 计算机科学, 2021, 48(10): 37–43. [doi: 10.11896/jsjkx.200900208]
9 Silver D, Lever G, Heess N, *et al*. Deterministic policy gradient algorithms. Proceedings of the 31st International Conference on Machine Learning. Beijing: ACM, 2014. I–387–I–395.
10 Lillicrap TP, Hunt JJ, Pritzel A, *et al*. Continuous control with deep reinforcement learning. arXiv:1509.02971, 2015.
11 Adam S, Busoniu L, Babuska R. Experience replay for real-time reinforcement learning control. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2012, 42(2): 201–212.
12 Courbariaux M, Bengio Y, David JP. BinaryConnect: Training deep neural networks with binary weights during propagations. Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal: ACM, 2015. 3123–3131.
13 Li ZF, Ni BB, Zhang WJ, *et al*. Performance guaranteed network acceleration via high-order residual quantization. Proceedings of the IEEE International Conference on Computer Vision. Venice: IEEE, 2017. 2603–2611.
14 LeCun Y, Bottou L, Bengio Y, *et al*. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998, 86(11): 2278–2324. [doi: 10.1109/5.726791]

(校对责编: 孙君艳)