汪恒, 卞俊伟, 孙希霞. 基于海洋捕食者算法和深度强化学习的策略搜索算法[J]. 智能计算机与应用,2025,15(11):75-81.
DOI:10.20169/j.issn.2095-2163.251112

基于海洋捕食者算法和深度强化学习的策略搜索算法

汪 恒, 卞俊伟, 孙希霞
(南京邮电大学 物联网学院, 南京 210003)

摘 要: 深度强化学习(Deep Reinforcement Learning, DRL)是一种常见的策略搜索算法,具有良好的样本采样率和学习效率,在解决高维问题方面表现优秀,但对环境的探索性不足,容易陷入局部最优。进化算法(Evolutionary Algorithm, EA)具有全局搜索能力强等优点,因此将进化算法与 DRL 结合用于提升 DRL 性能的方法成为了当前研究的热点。现有进化深度强化学习(Evolutionary DRL, EvoDRL)算法的 EA 部分大多采用粒子群算法、遗传算法等传统进化算法,存在收敛速度慢等不足。针对该问题,本文融合海洋捕食者算法(Marine Predators Algorithm, MPA)、交叉熵方法(Cross Entropy Method, CEM)和 DRL 算法,提出了一种 MPA-CEM-RL 算法,利用 MPA 种群的多样性经验训练种群中表现较差的个体,并将训练后的个体重新插入种群,促进 MPA 与 DRL 之间的信息交互;通过引入 CEM 的全局优化能力,进一步提升了 MPA-CEM-RL 算法的整体搜索性能。在 Mujoco 物理引擎中的仿真实验结果表明,和 ERL、CEM-RL、Surrogate-assisted-ERL 以及 MPA-RL 算法相比,本文提出的 MPA-CEM-RL 算法具有更高的性能和更好的稳定性。

关键词: 海洋捕食者算法; 深度强化学习; 进化算法; 交叉熵方法
中图分类号: TP181 **文献标志码**: A **文章编号**: 2095-2163(2025)11-0075-07

A strategy search method based on marine predator algorithm and deep reinforcement learning

WANG Heng, BIAN Junwei, SUN Xixia
(School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

Abstract: Deep Reinforcement Learning (DRL) is a common strategy search algorithm with good sampling rate and learning efficiency which performs well in solving high-dimensional problems. However, DRL cannot sufficiently explore the environment, which makes it prone to falling into local optima. Evolutionary Algorithms (EAs) have the advantages of strong global search ability, and therefore, the hybridization of EAs with DRL to improve DRL performance has become a hot research topic. The EA parts of existing evolutionary DRL (EvoDRL) algorithms typically use traditional EAs, such as particle swarm optimization and genetic algorithm. These traditional EAs exhibit shortcomings such as slow convergence speed. To solve this problem, this paper integrates the Marine Predators Algorithm (MPA), the Cross Entropy Method (CEM), and DRL, proposing a novel MPA-CEM-RL algorithm. This algorithm leverages the diversity experience of the MPA population to train underperforming individuals within the population and subsequently reintegrates the trained individuals back into the population. This mechanism facilitates enhanced information exchange between MPA and DRL. Furthermore, by incorporating the global optimization capability of CEM, the overall search performance of the MPA-CEM-RL algorithm is further improved. The simulation results in the Mujoco physics engine show that compared with ERL, CEM-RL, surrogate-assisted-ERL, and MPA-RL, the MPA-CEM-RL algorithm has higher performance and better stability.

Key words: Marine Predators Algorithm; Deep Reinforcement Learning; Evolutionary Algorithm; Cross Entropy Method

0 引 言

深度强化学习(Deep Reinforcement Learning, DRL)是一种结合了深度学习(Deep Learning, DL)和强化学习(Reinforcement Learning, RL)的机器学习算法,使智能体能够通过与环境交互来学习最优行为策略[1]。其基本原理是使用神经网络或者值函数来近似 RL 的策略,从而解决高维问题[2]。

作者简介: 汪 恒(2000—),男,硕士,主要研究方向:进化强化学习; 卞俊伟(2000—),男,硕士,主要研究方向:进化学习与优化算法。
通信作者: 孙希霞(1988—),女,博士,讲师,主要研究方向:进化算法与机器学习。Email:sxx1017@njupt.edu.cn。

收稿日期: 2024-02-26

DRL 因具有高样本效率和良好的学习能力,受到了研究者的广泛关注,并在游戏、机器人和自动驾驶等许多领域得到了广泛应用[3]。

尽管 DRL 算法在解决高维和大规模问题方面表现良好,但在复杂环境中的鲁棒性较低,且容易被欺骗性奖励误导,从而陷入局部最优。而进化算法(Evolutionary Algorithm,EA)则具有良好的全局搜索能力、鲁棒性、稳定性和并行性,但其采样效率较低,不能全面利用来自环境的反馈信号和历史数据[4]。进化强化学习算法(Evolutionary Reinforcement Learning, EvoDRL)将 DRL 和 EA 相结合,可以同时利用 DRL 的样本效率和 EA 的多样性和稳健性在学术界和工业界都受到了广泛的关注[5]。然而,现有 EvoDRL 算法的 EA 部分大多采用较为简单的进化算法,如粒子群(Particle Swarm Optimization, PSO)算法、遗传算法(Genetic Algorithm, GA)以及交叉熵方法(Cross Entropy Method, CEM)等,导致现有的 EvoDRL 算法存在收敛精度较低、不稳定等不足[6]。

海洋捕食者算法(Marine Predators Algorithm, MPA)作为一种新型的 EA,具有寻优能力强等特点[7]。现有研究表明,与 PSO、GA 相比,MPA 具有更强的优化能力。针对现有 EvoDRL 算法存在的 EA 部分性能不足的问题,本文将 MPA、CEM 和 DRL 算法结合,提出了一种混合 MPA-CEM-RL 算法。首先,融合 MPA 与 DRL,在每轮迭代中先通过 MPA 的进化机制更新种群;其次,使用 DRL 对种群中适应度值较低的一半个体进行更新,若更新后个体的适应度值优于原个体,则予以替换;为进一步提升算法的全局搜索能力,引入 CEM 算法替代 MPA 算法,提升全局搜索能力的鱼类聚集装置(Fish Aggregating Devices, FADs)效应。最后,在 Mujoco 的 4 个典型仿真环境中进行了仿真实验,实验结果表明:与 MPA-RL、ERL、CEM-RL 以及 Surrogate-assisted-ERL 算法相比,MPA-CEM-RL 算法具有更高的平均性能和更好的稳定性。

1 相关工作

DRL 算法主要分为两类:基于值的 DRL 算法和基于策略的 DRL 算法。基于值的算法包括 DQN(Deep Q Network, DQN)、双 DQN、决斗 DQN 和彩虹 DQN 等[8];基于策略的 DRL 算法包括确定性策略梯度(Deterministic Policy Gradient, DPG)算法、近端策略(Proximal Policy Optimization, PPO)算法和演员-评论家(Actor-Critic, AC)算法等[9]。基于策略的 DRL 算法具有处理连续动作空间问题的能力,并能够自适应地学习最优策略。因此,本文重点对基于策略的 DRL 算法进行研究,并将其与 EA 结合。

2015 年,Li 等[10]将演员-评论家算法与 DQN 网络相结合,提出了一种适用于连续控制场景的 DDPG(Deep Deterministic Policy Gradient, DDPG)算法。与大多数 DRL 算法类似,DDPG 算法也存在高估问题。为了解决这个问题,Fujimoto 等[11]提出了双延迟 DDPG(Twin Delayed Deep Deterministic Policy Gradient, TD3)算法,该算法结合了削波双 Q 学习、延迟策略更新和目标策略平滑策略,以减少高估偏差,提高算法稳定性。

基于策略的 DRL 仍存在对环境的探索性不足,容易陷入局部最优等问题。进化算法因其较强的全局搜索能力,可有效探索策略空间;在此基础上,结合 DRL 的梯度优化机制对策略进行精调,能实现更优的策略搜索与优化。EA 和 DRL 算法相互协作,从而取得更好的效果。Bai 等[12]对进化深度强化学习算法进行了全面综述,并将其分为 6 种类型。其中,一类算法利用进化算法引导 DRL,通过维持策略种群的多样性以增强探索能力,并借助种群进化机制生成新策略。因此,该类算法在处理复杂的连续控制任务时表现出显著优势。作为该算法的典型代表,Khadka 等[13]开创性地提出了进化强化学习(Evolutionary Reinforcement Learning, ERL)算法,利用 EA 种群产生的多样性经验训练 RL 代理,并定期将该代理重新注入 EA 种群,以融入梯度优化信息。ERL 借助 DRL 的梯度优化能力,显著提升了学习速度并降低了样本复杂度。此外,Pourchot 等[14]将 CEM 和 DRL 算法相结合,提出了 CEM-RL 混合算法。CEM-RL 和 ERL 的区别在于其使用不同类型的 EA 以及 EA 和 RL 之间的不同通信方法。在 ERL 中,RL 使用 EA 的经验来计算梯度和更新策略,并定期向 EA 发送梯度信息;CEM-RL 则直接对 EA 中的一半个体执行梯度更新。然而,上述两种算法在部分测试环境中性能表现不佳。本文引入一种更优的启发式算法,以提升 EvoDRL 的性能及其对不同环境的适应性与泛化能力。

2 MPA-CEM-RL 算法

Afshin[7]在 2020 年提出 MPA 算法,是一种新颖的启发式算法,由其灵感来源于海洋适者生存理

论，算法流程如图 1 所示。MPA 和传统的进化算法相比存在明显优势，因此基于 MPA 的 EvoDRL 算法理论上具有更好的性能。

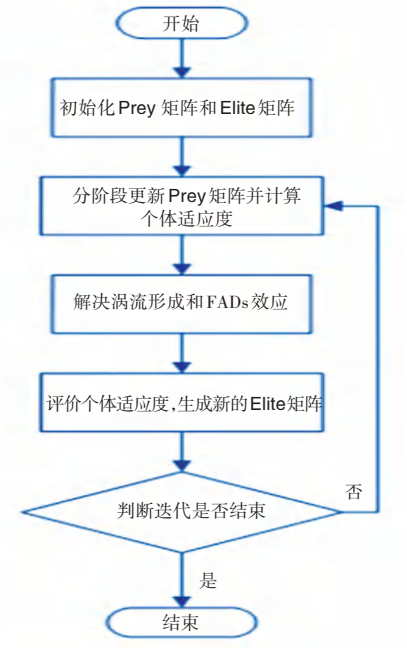

图 1　MPA 算法流程图

Fig. 1　Flowchart of MPA

首先，将 MPA 和 DRL 进行结合，RL 部分可以使用任意一种 AC 算法，本文使用 TD3 算法。在初始化阶段，首先初始化 MPA 的猎物（**Prey**）矩阵以及 TD3 的评价网络、目标评价网络。**Prey** 是一个 m 行 n 列的矩阵，其中 m 表示种群规模，初始化方法如下：

$$X_{ij} = X_{\min} + \text{rand} \times (X_{\max} - X_{\min}) \tag{1}$$

$$\mathbf{Prey} = \begin{bmatrix} X_{11} & \cdots & X_{1n} \\ \vdots & \ddots & \vdots \\ X_{m1} & \cdots & X_{mn} \end{bmatrix}_{m \times n} \tag{2}$$

其中，$X_{\min}$ 是网络参数的最小取值；$X_{\max}$ 是网络参数的最大取值；rand 为[0,1]内的随机数。

其次，将 **Prey** 矩阵的行向量 $\mathbf{Prey}_i$ 分别作为策略网络 π 的参数，根据下式计算每个个体 $\mathbf{Prey}_i$ 的适应度（fitness）。适应度定义为智能体在环境中执行单次完整交互所获得的累积奖励。

$$\text{fitness} = \sum_{i=1}^{\text{steps}} r_i \tag{3}$$

其中，r_i 是策略网络 π 在与环境的交互过程中第 i 步的奖励，steps 表示策略在环境中运行一轮的步数。

用 **Prey** 矩阵中适应度最大的个体 $\mathbf{Prey}_{\text{best}}$ 初始化 **Elite** 矩阵，如下式所示：

$$\mathbf{Elite} = \begin{bmatrix} \mathbf{Prey}_{\text{best}} \\ \vdots \\ \mathbf{Prey}_{\text{best}} \end{bmatrix}_m \tag{4}$$

在每轮迭代中，首先根据当前代 t 所处阶段更新 **Prey** 矩阵。

当 $t < T/3$ 时，**Prey** 矩阵的更新如下式所示：

$$\begin{cases} \text{stepsize}_i = \boldsymbol{R}_B \otimes (\mathbf{Elite}_i - \boldsymbol{R}_B \otimes \mathbf{Prey}_i) \\ \mathbf{Prey}_i = \mathbf{Prey}_i + P \times \boldsymbol{R} \otimes \text{stepsize}_i \end{cases}, i = 1, \cdots, m \tag{5}$$

当 $T/3 \leqslant t < 2 \times T/3$ 时，**Prey** 矩阵的更新如公式如下：

$$\begin{cases} \text{stepsize}_i = \boldsymbol{R}_L \otimes (\mathbf{Elite}_i - \boldsymbol{R}_L \otimes \mathbf{Prey}_i) \\ \mathbf{Prey}_i = \mathbf{Prey}_i + P \times \boldsymbol{R} \otimes \text{stepsize}_i \end{cases}, i = 1, \cdots, m/2 \tag{6}$$

$$\begin{cases} \text{stepsize}_i = \boldsymbol{R}_B \otimes (\boldsymbol{R}_B \otimes \mathbf{Elite}_i - \mathbf{Prey}_i) \\ \mathbf{Prey}_i = \mathbf{Elite}_i + P \times \text{CF} \otimes \text{stepsize}_i \end{cases}, i = m/2, \cdots, m \tag{7}$$

当 $2 \times T/3 \leqslant t$ 时，**Prey** 矩阵的更新如下式：

$$\begin{cases} \text{stepsize}_i = \boldsymbol{R}_L \otimes (\boldsymbol{R}_L \otimes \mathbf{Elite}_i - \mathbf{Prey}_i) \\ \mathbf{Prey}_i = \mathbf{Elite}_i + P \times \text{CF} \otimes \text{stepsize}_i \end{cases}, i = 1, \cdots, m \tag{8}$$

其中，$\boldsymbol{R}_B$ 是采用布朗随机游走产生的随机数组成的 n 维向量；$\boldsymbol{R}_L$ 是一个基于 Levy 分布的 n 维向量；stepsize_i 表示移动的步长；P 是一个值为 0.5 的常数；$\boldsymbol{R}$ 是一个由[0,1]中的均匀分布的随机数组成的 n 维向量；t 是当前迭代次数；T 是最大迭代次数；CF 是一个自适应参数，计算方如法下：

$$\text{CF} = \left(1 - \frac{t}{T}\right)^{\left(2\frac{t}{T}\right)} \tag{9}$$

Prey 矩阵更新后，计算新的 **Prey** 矩阵中各个体的适应度并将个体与环境交互过程中产生的所有经验 (s, a, r, s') 存入经验缓存区 $\boldsymbol{R}$ 中，s 和 s' 分别代表当前状态和下一时刻的状态，a 和 r 分别代表当前动作和从环境中获得的奖励。选择 **Prey** 矩阵中适应度较小的一半个体，分别将其作为参数注入 TD3 算法的策略网络 π 和目标策略网络 π_t，并从经验缓存区中抽取经验对策略网络和评价网络进行梯度更新。

TD3 中有两个评价网络和两个目标评价网络，其更新方式如下：

$$y \leftarrow r + \gamma \min_{i=1,2} Q^{\theta_i'}(s', \pi_t(s')) \tag{10}$$

$$\theta_i \leftarrow \text{argmin}_{\theta_i} \frac{1}{N} \sum (y - Q^{\theta_i}(s, a))^2 \tag{11}$$

其中，θ_i 和 $\theta_i^{'}$ (i=1,2)分别代表第 i 个评价网络和目标评价网络的参数，γ 是折扣因子。

TD3 的策略网络的更新所采用的损失函数如下：

$$J_\phi(s) = -Q^{\theta_1}(s,\pi(s)) \tag{12}$$

其中，ϕ 是策略网络的参数。

TD3 中的目标网络使用软更新的方法进行参数更新，目标评价网络和目标策略网络参数的更新方法如下：

$$\theta_i^{'} \leftarrow \tau\theta_i + (1-\tau)\theta_i^{'} \tag{13}$$

$$\phi^{'} \leftarrow \tau\phi + (1-\tau)\phi^{'} \tag{14}$$

其中，$\phi^{'}$ 是目标策略网络的参数。

网络更新完成后将更新后的策略网络的参数注入 **Prey** 矩阵。

最后，运用 MPA 中的涡流形成和 FADs 效应阶段进一步更新 **Prey** 矩阵，如下式所示：

$$\mathbf{Prey}_i = \begin{cases} \mathbf{Prey}_i + \mathrm{CF} \times [X_{\min} + \boldsymbol{R} \otimes (X_{\max} - X_{\min})] \otimes \boldsymbol{U}, & \text{if } r \leqslant \mathrm{FADs} \\ \mathbf{Prey}_i + [\mathrm{FADs} \times (1-r) + r] \times (\mathbf{Prey}_{r1} - \mathbf{Prey}_{r2}), & \text{if } r > \mathrm{FADs} \end{cases} \tag{15}$$

其中，$\boldsymbol{U}$ 是一个包含 0 和 1 的二进制 n 维向量；FADs 是一个影响优化过程的常数；r 是一个 [0,1] 中的随机数；r_1 和 r_2 是 **Prey** 的两个随机下标，$1 \leqslant r_1, r_2 \leqslant m$。

比较更新后的 **Prey** 矩阵中每个个体的适应度和上一代 **Prey** 矩阵对应个体的适应度，保留较优的个体，并选择 **Prey** 矩阵中的最优个体重新生成 **Elite** 矩阵。每轮迭代结束后，判断算法是否满足结束条件，如果满足则退出，否则进行下一轮迭代。

MPA-RL 中的解决涡流形成和 FADs 效应的方法本质上就是对 **Prey** 矩阵(种群)进行扰动，提高其全局搜索性能，使其尽可能跳出局部最优。CEM 方法作为一种蒙特卡洛方法，可以有效解决高维组合优化问题，具有较强的全局优化能力[15]。因此，可以将 MPA 中的用于解决涡流形成和 FADs 效应的方法替换为 CEM 方法。为进一步提升 MPA-RL 算法的全局探索能力和整体性能，本文进一步将 CEM 方法和 MPA-RL 算法融合，提出了一种 MPA-CEM-RL 算法。在 MPA-CEM-RL 的每轮迭代过程中，首先根据 MPA 的进化机制更新 **Prey** 种群；其次，根据适应度值对 **Prey** 种群进行排序，并将整个 **Prey** 种群分成两个同等大小的子种群，pop_1 和 pop_2，pop_1 为 **Prey** 种群中适应度较优的一半个体；用 CEM 算法根据 **Prey** 种群生成一个新的种群 pop，如下所示：

$$\eta = \sum_{i=1}^{k} \gamma_i \mathbf{Prey}_i \tag{16}$$

$$\Sigma = \sum_{i=1}^{k} \gamma_i (\mathbf{Prey}_i - \eta)^2 + \epsilon I \tag{17}$$

$$\mathrm{pop}^i \sim N(\eta, \Sigma) \tag{18}$$

其中，η 代表种群 **Prey** 中适应度较大的 k 个个体的均值；γ_i 表示第 i 个个体的权重，一般取 $\frac{1}{k}$；ϵI 表示添加到协方差中的噪声；pop^i 表示 CEM 生成的新种群 pop 中的第 i 个个体，pop 的规模和 **Prey** 种群一致。

为更好平衡算法的全局搜索能力和局部搜索能力，在 MPA-CEM-RL 算法进化过程的不同阶段各子种群的角色不同，且采用不同的更新机制。因为在进化前期，算法更倾向于全局搜索，而 CEM 具有非常强的全局搜索能力，因此用 CEM 对 **Prey** 种群进行整体的更新可以提高对环境的探索能力，因此在进化过程的前 1/3 阶段，用 CEM 生成的 pop 种群替换整个 **Prey** 种群，并选取新种群中适应度较低的一半个体进行梯度更新；因为在进化中期，算法需同时关注全局和局部搜索，DRL 样本效率高，收敛速度快，对于 pop_2 中适应度较差的个体采用 DRL 进行梯度更新，可以提高学习效率，而对于 pop_1 中适应度较优的个体仍采用 CEM 进行更新，可继续保持全局搜索能力，因此在进化过程的中间 1/3 阶段，用 pop 中的随机的 $\frac{m}{2}$ 个个体替换 pop_1 中的个体，并对 pop_2 中的个体进行梯度更新；因为在进化后期，算法主要注重局部搜索，较优的 pop_1 采用 DRL 进行更新，可进一步加速收敛，而较差的 pop_2 采用 CEM 进行更新则在一定程度上保留了算法的全局搜索能力，使算法的鲁棒性和稳定性更好，因此在进化过程的后 1/3 阶段，用 pop 中的随机的 $\frac{m}{2}$ 个个体替换 pop_2 中的个体，对 pop_1 中的个体进行梯度更新。MPA-CEM-RL 算法的伪代码如算法 1 所示。

算法 1　MPA-CEM-RL

输入　算法在环境中运行的最大步数 max_steps，MPA 种群的大小 m，评价网络参数 θ 和目标评价网络参数 $\theta^{'}$，批经验的大小 mini_batch

输出　种群中最优策略的累积奖励

1. 初始化 MPA 算法的 **Prey** 矩阵和 **Elite** 矩阵

2. 初始化评价网络 Q^{θ} 和目标评价网络 $Q^{\theta^{'}}$

3. 初始化一个空的经验缓存区 $\boldsymbol{R}$

4. 初始化 total_steps = 0

5. while total_steps<max_steps：

6. 使用 MPA 算法的分段优化策略更新 **Prey** 矩阵

7. actor_steps = 0

8. for $i \leftarrow 1$ to m：

9. 将 $\mathbf{Prey}_i$ 作为策略网络 π 的参数，评价策略网络 π 在环境中运行一轮的适应度 fitness 和步数 steps，并将收集到的经验存入 $\boldsymbol{R}$ 中，actor_steps += steps

10. end for

11. total_steps += actor_steps

12. 根据适应度将 **Prey** 矩阵进行排序，并根据当前代所处进化阶段用 CEM 对种群进行优化

13. for $i \leftarrow 1$ to $\frac{m}{2}$：//根据当前代所处进化阶段选取 $\frac{m}{2}$ 个个体进行策略更新

14. 将策略网络 π 和目标策略网络 π_t 的参数都设置为 $\mathbf{Prey}_i$

15. 从 $\boldsymbol{R}$ 中抽取 $\frac{2 \times \text{actor\_steps}}{m}$ 批经验训练评价网络 Q^{θ}

16. 从 $\boldsymbol{R}$ 中抽取 actor_steps 批经验训练策略网络 π

17. 将 $\mathbf{Prey}_i$ 更新为训练后的策略网络 π 的参数，并重新计算适应度

18. end for

19. 保留海洋记忆，更新 **Prey** 矩阵，并根据适应度最优的个体生成新的 **Elite** 矩阵

20. end while

3　仿真实验与结果分析

实验环境：Nivida GeForce RTX 3060ti GPU、Intel(R) Core(TM) i5-12600KF 和 Win11 系统。为了验证本文提出的算法的有效性，使用 Mujoco 物理引擎中的 4 个连续控制任务 HalfCheetah-v2、Hopper-v2、Walker2d-v2 和 Swimmer-v2 进行仿真实验，各环境的动作维度和状态维度见表 1[16]。

本文选择 ERL、CEM-RL 以及 ERL 的改进算法 Surrogate-assisted-ERL[17] 作为对比算法。实验中各算法采用的超参数值见表 2，各算法在各实验环境中均训练 1 000 000 步。

表 1　Mujoco 物理引擎中各环境的动作维度和状态维度

Table 1　Action and state dimensions of each environment in the Mujoco

环境	动作维度	状态维度
HalfCheetah-v2	6	17
Hopper-v2	3	11
Walker2d-v2	6	17
Swimmer-v2	2	8

表 2　实验中各算法采用的超参数值

Table 2　Hyperparameter values used in each algorithm in the experiment

参数	值
评价网络的隐藏层数	2
策略网络的隐藏层数	2
评价网络隐藏层的神经元个数	250
策略网络隐藏层的神经元个数	250
策略网络隐藏层的激活函数	Tanh
策略网络输出层的激活函数	Tanh
评价网络隐藏层的激活函数	Relu
评价网络输出层的激活函数	None
优化器	Adam
评价网络的学习率	0.001
策略网络的学习率	0.001
每批抽取的经验数 mini_batch	256
折扣因子 γ	0.99
目标网络软更新权重 τ	0.005
经验缓存区大小	1 000 000
种群大小	10

鉴于 EvoDRL 算法存在随机性，本组实验中每个算法在每个实验环境中执行 5 次，记录每种算法的 5 次独立试验的平均值，以衡量其性能。本文将第 t 轮中群体中最佳个体的性能视为所有进化深度强化学习算法在第 t 轮的性能。

5 种算法在 4 个环境中的最佳适应度曲线如图 3 所示，曲线的阴影部分为 5 次实验的标准差，横坐标表示算法在环境中运行的步数，纵坐标表示种群中最优个体的适应度。由于 Walker2d-v2 和 Hopper-v2 的每一轮迭代与环境交互的步数是可变的，因此以 20 000 步为间隔，将每段间隔中最优的一轮性能作为该间隔的性能。从图 2(a)可以看出，在 HalfCheetah-v2 环境中，MPA-RL 算法优于 ERL 及 Surrogate-assisted-ERL 算法，但是和 CEM-RL 算法相比存在差距；而融合了 CEM 的 MPA-CEM-RL

算法的性能明显优于 MPA-RL 算法,且从曲线上可以看出 MPA-CEM-RL 在性能上优于 CEM-RL;从图 2(b)可以看出,在 Hopper-v2 环境中,CEM-RL、MPA-RL 和 MPA-CEM-RL 算法的最终性能比较接近,都明显优于 ERL 算法,Surrogate-assisted-ERL 的性能也较好,但和本文提出的 MPA-CEM-RL 相比仍存在一定差距;从图 2(c)中可以看出,在 Walker2d-v2 环境中,CEM-RL 的算法性能的波动较大,但峰值较高,而 MPA-RL 因为保留了海洋记忆,即当前的 **Prey** 种群和上一代 **Prey** 种群进行比较,保留两个种群中对应个体中较优的一个,所以 MPA-RL 的算法性能是非递减的,MPA-RL 的标准差更小、更加稳定,但最佳适应度的峰值没有 CEM-RL 高,MPA-CEM-RL 算法综合了 CEM-RL 的收敛性和 MPA-RL 的稳定性等优点,不仅具有最高的最佳适应度峰值和较小的标准差,而且性能曲线是稳定上升的;从图 2(d)中可以看出,CEM-RL 算法在 Swimmer-v2 环境中的性能较差,ERL 算法具有较优的性能,本文提出的 MPA-CEM-RL 算法性能最好,因为 DRL 方法会提供欺骗性的梯度信息,不利于网络参数的收敛,由于 ERL 尽可能保持 EA 的探索性,而 CEM-RL 算法则可以利用更多的梯度信息,可能会导致干扰,从而使 CEM-RL 在 Swimmer-v2 中的性能不佳。

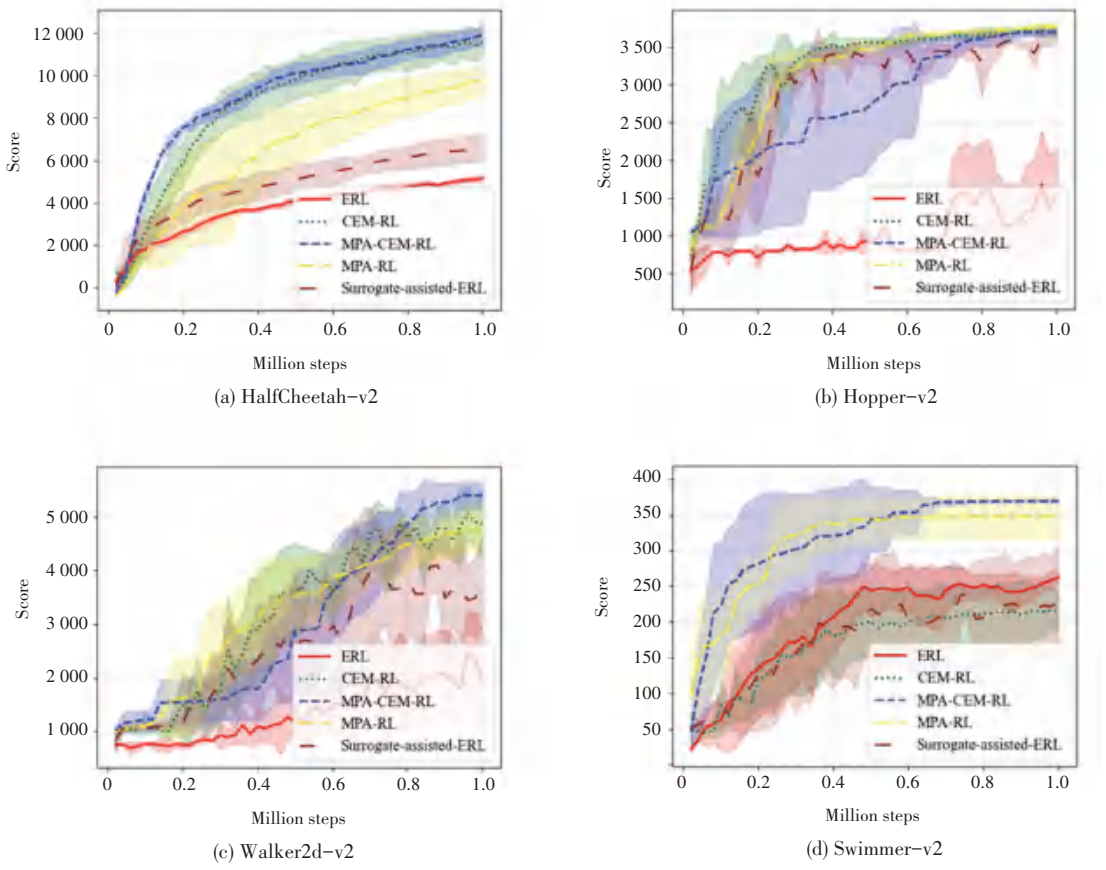

图 2 5 种算法在 4 个环境中的最佳适应度曲线

Fig. 2 Optimal fitness curves of five algorithms in four environments

MPA-CEM-RL 可以在重复利用梯度信息的同时还可以避免错误梯度信息的干扰,具有更高的稳定性。综上,在多种 Mojuco 连续任务环境中,本文提出的 MPA-CEM-RL 算法和目前主流的几种进化深度强化学习算法相比,其性能优势明显。

4 结束语

本文提出了一种结合 DRL 和启发式优化算法 MPA、CEM 的进化强化学习算法 MPA-CEM-RL。首先,融合 MPA 与 DRL 算法,使用 MPA 生成初始种群,并将种群个体与环境交互产生的经验数据存入经验回放缓冲区;其次,从该缓冲区采样数据,运用 DRL 对种群中的部分个体进行梯度更新,更新后的个体被重新注入 MPA 种群,实现 MPA 与 DRL 的双向交互与协同优化;引入 CEM 进一步优化,用 CEM 替换 MPA 中用于全局搜索的 FADs 效应机制,

并依据进化阶段选择不同部分的种群进行梯度更新，从而更好地平衡算法的全局探索与局部开发能力；最后，在 Mujoco 物理引擎的 4 个连续控制任务中进行了仿真实验。实验结果表明：MPA-CEM-RL 在性能和稳定性方面均优于本文中的所有对比算法。在未来的工作中，需要对算法的时间复杂度进行优化，并将其应用到多目标优化问题中。

参考文献

[1] 李波，黄晶益，万开方，等. 基于深度强化学习的无人机系统应用研究综述[J]. 战术导弹技术，2023(1)：58-68.

[2] PRASAD A，DUSPARIC I. Multi-agent deep reinforcement learning for zero energy communities[C]//Proceedings of 2019 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe). Piscataway，NJ：IEEE，2019：1-5.

[3] 吴思凡，段续庭，周建山，等. 基于深度强化学习的自动驾驶决策行为研究综述[J]. 人工智能，2023(5)：78-92.

[4] VIKHAR P A. Evolutionary algorithms：A critical review and its future prospects [C]// Proceedings of 2016 International Conference on Global Trends in Signal Processing，Information Computing and Communication. Piscataway，NJ：IEEE，2016：261-265.

[5] 吕帅，龚晓宇，张正昊，等. 结合进化算法的深度强化学习方法研究综述[J]. 计算机学报，2022，45(7)：1478-1499.

[6] SZITA I，LÖRINCZ A. Learning Tetris using the noisy cross-entropy method[J]. Neural Computation，2006，18(12)：2936-2941.

[7] FARAMARZI A，HEIDARINEJAD M，MIRJALILI S，et al. Marine predators algorithm：A nature-inspired metaheuristic[J]. Expert Systems with Applications，2020，152：113377.

[8] ZHOU S，LIU X，XU Y，et al. A deep Q-network (DQN) based path planning method for mobile robots[C]// Proceedings of 2018 IEEE International Conference on Information and Automation (ICIA). Piscataway，NJ：IEEE，2018：366-371.

[9] TAN H. Reinforcement learning with deep deterministic policy gradient[C]// Proceedings of 2021 International Conference on Artificial Intelligence，Big Data and Algorithms (CAIBDA). Piscataway，NJ：IEEE，2021：82-85.

[10] LI D，DONG C. Double-net DDPG with the optimal action selection mechanism[C]// Proceedings of 2022 IEEE 11th Data Driven Control and Learning Systems Conference (DDCLS). Piscataway，NJ：IEEE，2022：1166-1170.

[11] FUJIMOTO S，HOOF H，MEGER D. Addressing function approximation error in actor-critic methods[C]// Proceedings of International Conference on Machine Learning. PMLR，2018：1587-1596.

[12] BAI H，CHENG R，JIN Y. Evolutionary reinforcement learning：A survey[J]. Intelligent Computing，2023，2：0025.

[13] KHADKA S，TUMER K. Evolution-guided policy gradient in reinforcement learning[J]. arXiv，1805.07917，2018.

[14] POURCHOT A，SIGAUD O. CEM-RL：Combining evolutionary and gradient-based methods for policy search[J]. arXiv preprint arXiv，1810.01222，2018.

[15] MACHLEV R，CHOWDHURY N R，BELIKOV J，et al. Distributed storage placement policy for minimizing frequency deviations：A combinatorial optimization approach based on enhanced cross-entropy method [J]. International Journal of Electrical Power and Energy Systems，2022，134：107332.

[16] TODOROV E，EREZ T，MUJOCO Y T. A physics engine for model-based control [C]//Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway，NJ：IEEE，2012：5026-5033.

[17] WANG Y，ZHANG T，CHANG Y，et al. A surrogate-assisted controller for expensive evolutionary reinforcement learning[J]. Information Sciences，2022，616：539-557.