电子信息硕士学位论文

基于改进ACER算法的智能体离散动作控制决策研究

2025年6月

国内图书分类号：TP181

电子信息硕士学位论文

基于改进ACER算法的智能体离散动作控制决策研究

硕 士 研究生：

导 师：

申请学位级别：工程硕士

学 科、专 业：电子信息

所 在 单 位：

答 辩 日 期：2025年6月

授予学位单位：

Classified Index: TP181

Dissertation for the Master Degree in  
Election Information

**Research on Intelligent Agent Discrete Action Control Decision Based on Improved ACER Algorithm**

|  |  |
| --- | --- |
| **Candidate：** |  |
| **Supervisor：** |  |
| **Academic Degree Applied for：** | Master of Election Information |
| **Speciality：** |  |
| **Date of Oral Examination：** | June, 2025 |
| **University：** |  |

**学位论文原创性声明**

本人郑重声明：此处所提交的学位论文《基于改进ACER算法的智能体离散动作控制决策研究》，是本人在导师指导下，在攻读学位期间独立进行研究工作所取得的成果。据本人所知，论文中除已注明部分外不包含他人已发表或撰写过的研究成果。对本文研究工作做出贡献的个人和集体，均已在文中以明确方式注明。本声明的法律结果将完全由本人承担。

作者签名： 日期： 年 月 日

**学位论文使用授权书**

《基于改进ACER算法的智能体离散动作控制决策研究》系本人在大学攻读学位期间在导师指导下完成的学位论文。本论文的研究成果归大学所有，本论文的研究内容不得以其它单位的名义发表。本人完全了解大学关于保存、使用学位论文的规定，同意学校保留并向有关部门提交论文和电子版本，允许论文被查阅和借阅。本人授权大学可以采用影印、缩印或其他复制手段保存论文，可以公布论文的全部或部分内容。

本学位论文属于

保密 ，在 年解密后适用授权书。

不保密 √ 。

作者签名： 日期： 年 月 日

导师签名： 日期： 年 月 日

基于改进ACER算法的智能体离散动作控制决策研究

摘 要

近年来，强化学习及其深度强化学习衍生技术，已成为智能体决策领域的重要方。然而，针对离散动作空间的控制问题，现有的强化学习算法仍面临不小的挑战，尤其在大规模状态空间和复杂环境下，传统算法往往依赖环境反馈，探索效率低，且学习过程不稳定。为此，本文围绕ACER（Actor-Critic with Experience Replay）算法的改进进行研究，提出了两种创新方法：基于自注意力机制和优先级经验回放的演员评论家算法和基于元学习和自适应奖励的优先级经验回放演员评论家算法。这些改进旨在提升离散动作控制任务中的决策效率、策略优化效果和算法的泛化能力。

针对传统ACER算法在离散动作问题中的情况，如不能有效筛选对策略优化关键的经验样本，导致学习过程收敛慢且计算资源浪费等。本文引入优先级经验回放机制，使智能体在训练过程中更关注那些对策略改进有较大贡献的样本，从而加速收敛并提高训练效率。同时，结合自注意力机制，智能体能够精准关注关键的历史状态信息，使得智能体能够自动聚焦于当前决策最相关的特征。最后，还结合目标网络和梯度削减等技术，增强了策略优化的稳定性，确保算法能够高效且稳定地解决离散控制问题，避免传统方法中常见的过拟合和不稳定现象。

针对ACER算法处理离散动作控制的动态性决策问题的学习能力局限性和固定奖励函数的不足情况，为了提高智能体在离散控制的动态性任务中的学习能力以及适应性和鲁棒性，本文引入元学习框架，通过元学习自动调整奖励函数。元学习能够通过多任务学习经验，帮助智能体在新任务中快速调整和优化奖励机制，从而提升自适应能力。新的任务中，智能体能够根据具体需求，通过元学习框架动态调整奖励机制，确保奖励函数与任务目标保持一致，从而提高算法的鲁棒性和泛化能力。

关键词 深度强化学习；ACER算法；离散动作控制；自注意力机制；元学习

**Research on Intelligent Agent Discrete Action Control Decision Based on Improved ACER Algorithm**

Abstract

In recent years, reinforcement learning and its derived techniques from deep reinforcement learning have become important players in the field of intelligent decision-making. However, existing reinforcement learning algorithms still face significant challenges in controlling discrete action spaces, especially in large-scale state spaces and complex environments. Traditional algorithms often rely on environmental feedback, resulting in low exploration efficiency and unstable learning processes. Therefore, this article focuses on the improvement of the ACER algorithm and proposes two innovative methods: the Actor Critic algorithm based on self attention mechanism and priority experience replay, and the Priority Experience Replay Actor Critic algorithm based on meta learning and adaptive rewards. These improvements aim to enhance decision efficiency, policy optimization effectiveness, and algorithm generalization ability in discrete action control tasks.

Regarding the bottleneck of traditional ACER algorithm in discrete action problems, such as the inability to effectively screen empirical samples that are crucial for policy optimization, the learning process converges slowly and computational resources are wasted. Therefore, this article introduces a priority experience replay mechanism to make the agent pay more attention to samples that have made significant contributions to policy improvement during the training process, thereby accelerating convergence and improving training efficiency. At the same time, combined with self attention mechanism, the intelligent agent can accurately focus on key historical state information, enabling the agent to automatically focus on the most relevant features of the current decision, further improving the quality of strategy optimization. On this basis, by combining target networks and gradient reduction techniques, the stability of policy optimization is enhanced, ensuring that the algorithm can efficiently and stably solve discrete control problems, avoiding common overfitting and instability phenomena in traditional methods.

In response to the limitations of learning ability and the inadequacy of fixed reward functions in the ACER algorithm, in order to improve the learning ability, adaptability, and robustness of agents in discrete tasks, this paper introduces a meta learning framework that automatically adjusts the reward function through meta learning. Meta learning can help agents quickly adjust and optimize reward mechanisms in new tasks by learning from multiple tasks, thereby enhancing their adaptive capabilities. In the new task, the intelligent agent can dynamically adjust the reward mechanism through a meta learning framework according to specific needs, ensuring that the reward function is consistent with the task objectives, thereby improving the robustness and generalization ability of the algorithm.

**Keywords** Deep reinforcement learning; Actor-Critic with Experience Replay algorithm; discrete action control; self-attention mechanism; meta-learning;

目 录

[摘 要 II](#_Toc502756524)

[Abstract II](#_Toc502756525)

[第1章 绪 论 1](#_Toc502756526)

[1.1 课题背景及研究的目的和意义 1](#_Toc502756527)

[1.2 国内外研究现状 2](#_Toc502756528)

[1.2.1 强化学习与深度强化学习研究现状 2](#_Toc502756529)

[1.2.2 ACER算法研究现状 2](#_Toc502756530)

[1.2.3 智能体离散动作控制研究现状 3](#_Toc502756531)

[1.3 本文研究的主要内容 4](#_Toc502756533)

[1.4 本文结构安排 5](#_Toc502756533)

[第2章 相关理论研究 6](#_Toc502756534)

[2.1 引言 6](#_Toc502756535)

[2.2 强化学习和深度强化学习模型 6](#_Toc502756535)

[2.2.1 马尔可夫决策与贝尔曼过程 7](#_Toc502756537)

[2.2.2 深度Q网络 9](#_Toc502756537)

[2.2.2 演员评论家算法 1](#_Toc502756537)0

[2.2.2 演员评论家经验回放算法 11](#_Toc502756537)

[2.3 经验回放与优先级经验回放 13](#_Toc502756536)

[2.4 元学习模型 13](#_Toc502756544)

[2.5 本章小结 1](#_Toc502756545)7

[第3章 基于自注意力机制的优先级经验回放演员评论家算法 18](#_Toc502756534)

[3.1 引言 18](#_Toc502756535)

[3.2 算法框架设计 18](#_Toc502756535)

[3.2.1 算法架构概述 18](#_Toc502756537)

[3.2.2 环境交互与经验采样 19](#_Toc502756537)

[3.2.3 优先级经验回复池模块 19](#_Toc502756537)

[3.2.4 自注意力机制模块 19](#_Toc502756537)

[3.2.5 演员评论家训练模块 20](#_Toc502756537)

[3.3 自注意力机制的引入 21](#_Toc502756535)

[3.3.1 自注意力机制概述 21](#_Toc502756537)

[3.3.2 自注意力机制的具体实现 22](#_Toc502756537)

[3.3.3 引入自注意力机制的优势 23](#_Toc502756537)

[3.4 优先级经验回放技术的引入 23](#_Toc502756535)

[3.4.1 优先级经验回放机制概述 23](#_Toc502756537)

[3.4.2 优先级经验回放技术的具体实现 24](#_Toc502756537)

[3.4.3 优先级经验回放技术的优势 25](#_Toc502756537)

[3.5 目标网络更新机制的引入 25](#_Toc502756535)

[3.5.1 目标网络更新机制概述 25](#_Toc502756537)

[3.5.2 目标网络更新机制的具体实现 26](#_Toc502756537)

[3.5.3 目标网络更新机制的优势 26](#_Toc502756537)

[3.6 A2CPER算法伪代码 27](#_Toc502756535)

[3.7本章小结 27](#_Toc502756535)

[第4章 基于元学习和自适应奖励的优先级经验回放演员评论家算法 29](#_Toc502756534)

[4.1 引言 29](#_Toc502756535)

[4.2 算法框架设计 29](#_Toc502756535)

[4.2.1 算法架构概述 29](#_Toc502756537)

[4.2.2 环境初始化与任务选择 30](#_Toc502756537)

[4.2.3 元学习框架 30](#_Toc502756537)

[4.2.4 自奖励机制 31](#_Toc502756537)

[4.2.5 ACER核心优化 31](#_Toc502756537)

[4.3 元学习框架的引入 32](#_Toc502756535)

[4.3.1 元学习框架概述 32](#_Toc502756537)

[4.3.2 元学习框架的具体实现 33](#_Toc502756537)

[4.3.3 元学习框架的优势 34](#_Toc502756537)

[4.4 自奖励机制的引入 34](#_Toc502756535)

[4.4.1 自奖励机制概述 34](#_Toc502756537)

[4.4.2 自奖励机制的具体实现 35](#_Toc502756537)

[4.4.3 自奖励机制的优势 36](#_Toc502756537)

[4.5 MACPER算法伪代码 36](#_Toc502756535)

[4.6本章小结 37](#_Toc502756535)

[第5章 离散动作控制决策实验与分析 38](#_Toc502756534)

[5.1 引言 38](#_Toc502756535)

[5.2 A2CPER算法处理离散动作静态性控制决策实验与分析 38](#_Toc502756535)

[5.2.1 实验环境 38](#_Toc502756537)

[5.2.2 实验代码设置 42](#_Toc502756537)

[5.2.3 实验结果以及分析 44](#_Toc502756537)

[5.2.4 实验总结 50](#_Toc502756537)

[5.3 MACPER算法处理离散动作动态性控制决策实验与分析 51](#_Toc502756535)

[5.3.1 实验环境 51](#_Toc502756537)

[5.3.2 实验代码设置 53](#_Toc502756537)

[5.3.3 实验结果以及分析 54](#_Toc502756537)

[5.3.4 实验总结 57](#_Toc502756537)

[5.4本章小结 58](#_Toc502756535)

[结 论 59](#_Toc502756546)

[参考文献 61](#_Toc502756547)

[攻读硕士学位期间发表的学术论文及获得成果 65](#_Toc502756548)

[致 谢 66](#_Toc502756549)

# 绪 论

## 课题背景及研究的目的和意义

近年来，随着人工智能技术的不断进步，强化学习作为一种基于试错学习的决策方法，已在多个领域展现出了巨大的应用潜力[1]。尤其是深度强化学习的出现，极大地扩展了强化学习的应用范围[2]。在离散动作空间的控制问题中上述方法依然面临着许多挑战，离散动作空间指的是在每个决策步骤中，智能体只能选择有限的几个动作，而不像连续动作空间那样可以进行细粒度的调整[3]。

具体来说，传统强化学习算法在离散动作空间中的主要局限性首先表现为，由于深度强化学习模型通常需要大量的样本来进行训练，并且环境交互的反馈信息有时效率很低，使得训练过程往往需要经历大量的时间步，才能从环境中获得有用的经验[4]。其次，深度强化学习中常见的策略梯度和价值函数方法往往收敛速度较慢，尤其是在高维状态空间下。最后，传统深度强化学习算法在训练过程中容易出现策略波动，导致模型训练的稳定性差，最终影响智能体的决策质量。为了进一步提升深度强化学习在离散动作空间中的表现，本研究计划提出两种改进的深度强化学习算法，通过有效提升样本利用效率和加速策略优化，解决现有算法在离散动作问题中的不足。

本研究的主要目标是基于现有的演员评论家框架和经验回放机制ACER（Actor-Critic with Experience Replay）算法，提出两种新的改进方法，分别为基于自注意力机制的优先级经验重放演员评论家算法（Attention Actor-Critic with Priority Experience Replay,A2CPER）。和基于元学习和自适应奖励的优先级经验回放演员评论家算法（Meta Learning Actor-Critic with Priority Experience Replay，MACPER）。A2CPER算法结合了自注意力机制、优先经验重放和目标网络等技术，通过优化样本选择过程，提升了策略优化的效率和稳定性。另一个改进算法MACPER结合了元学习框架、自奖励机制、优先级经验回放和目标网络等技术，以应对不同任务和环境的需求，提升算法的泛化能力和鲁棒性。

通过本研究的推进，旨在为深度强化学习在离散动作控制任务中的应用提供新的理论框架和技术支持，并推动深度强化学习算法在实际环境中的推广与应用。特别是在自动化控制、机器人、智能交通等领域，A2CPER算法和MACPER算法有望为离散动作控制决策系统提供高效、稳定的技术支持。

## 国内外研究现状

本节从多个角度、层面开始全面论述国内、国外与之相关研究现状，以把握相关研究目前与未来的发展形势。

### 强化学习与深度强化学习研究现状

近年来，强化学习及其深度强化学习衍生技术已成为人工智能领域的核心研究方向之一。通过与环境的交互，使智能体能够通过奖励信号来优化决策策略，从而达到最大化累积回报的目标。自上世纪80年代提出以来，在控制系统、游戏对战、自动驾驶和机器人控制等多个领域取得了显著的成果[5]。

国外的研究团队DeepMind提出了深度Q网络（Deep Q-Network，DQN）成为了深度强化学习的开创性突破。DQN将深度神经网络与Q-learning[6]相结合，通过使用卷积神经网络来近似Q值函数，从而成功应对了高维状态空间的问题[7]。自DQN提出以来，深度强化学习在多个领域取得了显著进展，包括AlphaGo在围棋领域的突破。国外的DeepMind、OpenAI等机构的研究人员不断推动深度强化学习的前沿技术，也提出创新性算法，如A3C（Asynchronous Advantage Actor-Critic）[8]、PPO（Proximal Policy Optimization）[9]等，这些算法通过引入异步更新、正则化、限制策略变化等技术，显著提升了深度强化学习的稳定性和鲁棒性。

国内的研究则主要集中在深度强化学习的应用与优化方向。例如，清华大学提出的双重Q学习（Double Q-learning）[10]和深度确定性策略梯度（Deep Deterministic Policy Gradient，DDPG）[11]等方法，在许多应用中取得了不错的效果。与此同时，国内研究者也关注算法的理论研究与改进，如提出了多智能体系统中的强化学习方法、基于策略梯度的优化算法等，以提高算法的鲁棒性和适应性。

总的来说，强化学习和深度强化学习的研究已经取得了显著的进展，尤其是在大规模、高维环境中的应用和优化。然而，针对离散动作空间的强化学习问题，尽管已有一定的技术积累，但在算法的样本效率、收敛速度和策略稳定性等方面仍面临挑战。因此，如何进一步提高强化学习在离散控制任务中的表现，仍然是当前研究的热点和难点。

### ACER算法研究现状

ACER算法是近年来深度强化学习领域的一个重要研究成果，结合了演员评论家框架和经验重放机制，通过对智能体策略的评价与优化，解决了传统强化学习方法中存在的一些关键问题[12]。ACER算法的提出和发展主要源于强化学习在大规模、复杂环境中的应用需求，尤其是在离散动作空间和高维状态空间的问题中，ACER算法展现了优越的性能。因此，ACER算法不仅在理论上具有创新意义，在实际应用中也具有广泛的前景。

国内研究者主要集中在如何改进ACER算法的稳定性、样本效率和多任务适应性等方面[13]做出了研究。ACER算法的稳定性改进其核心问题之一是训练过程中的梯度不稳定，算法容易在高维度下出现过拟合和策略更新不稳定的问题。为此，研究学者结合目标网络与经验重放机制，提出了一种基于批量归一化的稳定性增强策略，使得算法在大规模状态空间下更加稳定[14]。此外，优先经验重放机制也是国内研究的一个方向。传统ACER算法无法分析经验重要性，这在复杂任务中降低了学习效率。因此提出了一种优先经验重放算法，在训练过程中为经验样本赋予优先级，优先选择对策略优化贡献较大的经验，从而提高了样本效率和算法的收敛速度[15]。

在国外，引入深度神经网络与卷积网络是科研人员针对ACER算法性能提升的一个研究方向。深度神经网络的引入能够提高策略网络对高维状态信息的表示能力，从而提升ACER算法在复杂环境中的策略优化能力[16]。在他们的研究中，深度卷积网络被用于处理高维输入数据（如图像），这使得ACER算法能够在视觉输入环境中稳定工作[17]。

ACER算法作为一种结合了演员评论家架构和经验重放机制的强化学习方法，在国内外的研究中已经取得了显著进展。随着深度学习、元学习和自注意力机制等技术的发展，ACER算法的理论和应用前景更加广阔，未来的研究将继续围绕其在复杂任务中的优化、跨领域应用以及算法扩展等方面展开。

### 智能体离散动作控制研究现状

在国内，智能体离散动作控制作为强化学习领域的一个重要研究方向，普遍存在于多种实际应用中，如机器人控制、智能交通系统以及游戏中的策略决策等。离散动作控制中的策略优化问题是国内研究的重点之一，研究学者提出了一种基于Q学习的策略优化方法，针对传统Q-learning算法在离散动作空间中收敛慢、样本效率低的问题，通过引入深度神经网络来优化动作价值函数，从而提升了算法在复杂离散控制任务中的表现[18]。此外，国内研究学者还关注离散动作控制中的强化学习算法稳定性问题。为此研究出基于策略梯度的离散控制算法，相关学者使用基于软策略的稳定性增强算法，结合熵正则化技术和动量梯度优化，显著提升了策略的收敛速度和稳定性[19]。

在国外，许多著名的强化学习算法，如DQN和A3C，已经在解决离散动作空间的控制问题中取得了突破。基于策略梯度的离散动作控制方法是国外研究的一个重点。国外学者提出的A3C算法通过引入多个并行智能体的训练方式，有效解决了深度强化学习在离散动作空间中的收敛速度和稳定性问题。多智能体系统中的离散动作控制在国外也得到了深入研究，通过引入基于Q-learning的多智能体强化学习算法，解决了在离散动作空间中多个智能体协作和竞争的策略问题[20]。

智能体离散动作控制问题是深度强化学习中的一个重要研究方向，国内外学者针对这一问题进行了广泛的研究。随着深度神经网络、元学习、自注意力机制等技术的发展，智能体离散动作控制的研究将进入更加复杂和多元化的阶段，未来的研究有望进一步提升算法的性能、稳定性和应用范围。

## 本文研究的主要内容

本研究针对离散动作控制问题中的深度强化学习算法展开了系统的研究，提出了基于改进ACER算法的创新方法。本文的研究工作主要集中在：

首先，本文提出了基于自注意力机制的优先级经验回放演员评论家算法。首先引入优先级经验回放机制，确保智能体在训练过程中更关注那些对策略优化贡献更大的高价值经验。其次引入了自注意力机制，使得智能体在处理历史经验时能够动态地关注关键的状态信息，从而提高策略网络对重要特征的学习能力。最后为了保证算法在训练过程中的稳定性，又结合了目标网络和梯度裁剪技术，来防止传统强化学习方法中常见的梯度爆炸和网络更新不稳定的问题。通过这些创新，A2CPER在离散动作控制的连贯性问题中展现出了显著的性能提升，尤其是在策略优化效率和训练稳定性方面。然后将A2CPER应用于GYM库中的经典离散控制问题，如小车上山（CartPole）和倒立摆（Inverted Pendulum）等环境，来验证A2CPER算法处理离散动作的静态性决策问题的优势性。

其次，针对A2CPER算法处理离散动作控制的动态性问题的不足，本文提出了基于元学习和自适应奖励优先级经验回复演员评论家算法，以应对传统强化学习中固定奖励函数带来的局限性和多任务学习能力的不足。传统强化学习算法通常依赖于人工设计的奖励函数，但是这些奖励函数在不同任务或环境中的适应性较差，尤其在面对复杂任务和动态环境的情况下。为此本文引入了元学习框架，使得智能体能够在多个任务中学习到适应性的奖励函数，并能够根据不同任务的需求动态调整奖励机制。并且通过自奖励机制，来让算法的奖励获取情况更有效。在通过优先级经验回放机制来管理环境的多样性问题。在本研究中，元学习框架被应用于迷宫求解等经典离散问题中，进一步增强了智能体对环境的适应能力。

## 本文结构安排

第一章为本次研究的绪论部分，首先介绍了课题的背景及研究的目的和意义，阐述了在解决离散动作控制问题时强化学习及其衍生的深度强化学习面临的挑战。接着，回顾了国内外在深度强化学习、ACER算法及智能体离散动作控制领域的研究现状，并指出了当前方法存在的不足。最后，本章总结了本文的研究目标、方法及创新贡献，并简要介绍了论文的结构安排。

第二章为本次研究的相关基础理论研究，介绍了强化学习和深度强化学习的基本理论和模型。首先详细阐述了马尔可夫决策过程与贝尔曼过程作为强化学习的理论基础。接着深入分析了演员评论家算法以及衍生算法的基本原理。此外，本章还详细讨论了经验回放及优先级经验回放的相关技术，并介绍了元学习模型的基本框架和方法，为后续算法的提出和研究奠定理论基础。

第三章为本次研究的第一个核心部分，实现了提出的A2CPER算法。本章详细说明了A2CPER的算法框架以及如何实现，包括自注意力机制的实现、优先级经验回放的实现、Actor-Critic架构的具体实现、目标网络和梯度裁剪技术的结合使用，共同组成了A2CPER算法，并给出了算法的相关信息。

第四章也是本次研究的重点部分，实现了另一个提出的MACPER算法。本章详细解释了元学习方法的使用，特别是在奖励函数优化中的应用。阐述了如何将元学习与自适应奖励机制结合，通过任务间的共享知识快速调整奖励函数，还有相关方法如优先级经验回放方法的实现等等，并给出了算法的相关信息。

第五章是本篇论文的相关实验部分。本章将对所提出的两种改进算法A2CPER和MACPER进行实验验证，展示其分别在处理离散动作连贯性问题和离散动作动态性问题的表现。通过与传统算法的对比，分析两种算法在单任务离散问题和多任务离散问题中的样本效率、收敛速度、策略稳定性等方面的优越性。

结论为本次研究的最后论文总结部分。本章总结了主要研究工作，回顾了提出的改进算法在离散动作控制任务中的创新性和有效性。指出了本研究的不足之处，并对未来研究方向提出展望，为后续相关领域的研究提供参考。

# 相关基础理论研究

## 引言

本章旨在为后续的算法设计与实验研究提供必要的理论支持。通过对基础理论的深入探讨，为后续算法改进的设计与分析奠定了理论基础，并为实验部分的展开提供了方法论支持。

## 强化学习和深度强化学习模型

强化学习是一类基于智能体与环境交互的学习方法。智能体通过在环境中采取行动并接收反馈奖励来优化其行为，从而实现长期目标的最大化[21]。与监督学习不同，强化学习不依赖于人工标注数据，而是通过环境交互发现策略。这一过程类似于“试错”过程，智能体在每个状态下采取一定动作后，环境反馈的奖励信号帮助智能体评估行为的好坏，并通过奖励调整未来决策，并且不断调整策略，来使得智能体的效果不断增强。

强化学习的基本组成要素包括状态、动作、奖励、策略和价值函数。其中，状态表示环境的当前状态；动作表示智能体在该状态下采取的行为；奖励是智能体执行动作后得到的环境反馈[22]。

深度强化学习是强化学习与深度学习相结合的产物，旨在解决传统强化学习在处理高维、复杂任务时的瓶颈[23]。深度强化学习通过引入深度神经网络，能够在未被显式标注的数据中自动提取特征，从而在解决高维状态空间、连续动作空间等复杂问题时具有显著的优势。

深度强化学习的工作流程与传统强化学习相似，主要由以下几个组成部分。

状态空间代表环境的状态集合，通常表示为；动作空间表示智能体可以在每个状态下采取的动作集合，通常表示为；策略作为智能体的决策规则，从状态空间映射到动作空间[24]。传统强化学习中，策略通常用表格或函数表示；在深度强化学习中，策略通常由深度神经网络表示。

深度强化学习通过使用深度神经网络来逼近这些函数，通常是逼近状态价值函数或动作价值函数。学习的目标是找到一个最优策略，使得智能体在给定状态下采取最优的行动，从而最大化累积奖励[25]。机器学习中主要分支关系见图1-1。



图1-1 强化学习和深度强化学习等关系

Fig. 1-1 The relationship between different machine learning methods

### 马尔可夫决策与贝尔曼过程

在强化学习中，马尔可夫决策过程（MDP）用于描述一个智能体与环境之间的交互过程。通过MDP建模，能够在任意状态下选择最优动作，并最大化长期奖励，马尔可夫决策过程见图2-2。MDP由五个基本要素组成：状态空间、动作空间、状态转移概率、奖励函数和折扣因子。

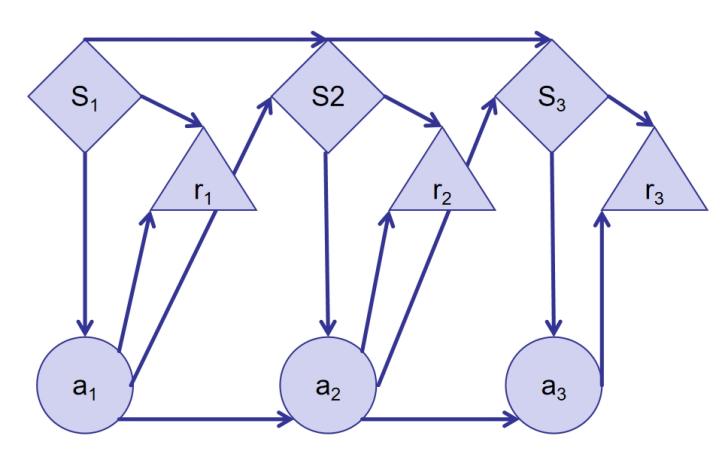


图2-2 马尔可夫决策过程

Fig. 2-2 Markov decision process

在MDP中，智能体的目标是通过一个策略来最大化从初始状态开始，执行一系列动作后所获得的累积奖励[26]。策略是从状态空间到动作空间的映射，即。基于此，MDP可以用以下数学表示：

其中：是状态空间，代表智能体可能的状态；是动作空间，代表智能体可以选择的动作；是状态转移概率，表示在状态下采取动作后转移到状态的概率；是即时奖励，表示在状态下采取动作后获得的奖励；是折扣因子，用来权衡即时奖励与未来奖励的重要性。

智能体在MDP中所期望的目标是最大化从初始状态出发，按照某个策略执行动作后获得的期望累计奖励。累积奖励是一个随机变量，通常用回报函数来表示。回报定义为：

其中：表示从时间步开始到未来的累计奖励，是折扣因子表示未来奖励的递减效应，是在第步执行动作后获得的即时奖励。

MDP中的关键任务之一是求解智能体的最优策略。最优策略是能够最大化智能体从任意状态开始的期望回报的策略。值函数用于评估某一策略的好坏，其定义为在状态下，按照策略执行动作后的期望回报。状态值函数表示在给定策略下，从状态开始的期望回报。它的数学表达式为：

其中：和分别表示在时间步时的状态和动作，是在时间步时执行动作所获得的即时奖励。动作值函数：表示在状态下采取动作，然后按照策略执行下去的期望回报。其数学表达式为：

贝尔曼方程的推导基于“当前状态的价值等于执行当前动作获得的即时奖励与未来状态价值的加权和”的思想[27]。状态值函数的贝尔曼方程：

其中：此方程表示，从状态出发，选择动作后转移到状态，所获得的期望回报等于当前的即时奖励加上折扣后的下一个状态的期望值。动作值函数的贝尔曼方程：

其中：在状态下执行动作，然后转移到下一个状态，所获得的期望奖励等于当前的即时奖励加上折扣后的下一个状态的期望值。

最优策略的目标是最大化从状态出发的期望回报。最优策略的求解是通过最优值函数来实现的。最优状态值函数和最优动作值函数满足以下最优贝尔曼方程，最优状态值函数的贝尔曼方程：

其中：该方程表示最优策略下，从状态出发，选择使期望回报最大化的动作。

最优动作值函数的贝尔曼方程：

其中：该方程表示在状态下执行动作后，转移到状态，然后根据最优策略选择最好的动作从而获得最大期望回报。

马尔可夫决策过程是强化学习的基础框架，它为智能体提供了数学化的方式来建模环境和学习策略。通过贝尔曼方程，智能体能够递归地计算状态的价值，并逐步优化策略，从而最大化累积奖励。贝尔曼方程是强化学习和深度强化学习中求解最优策略和优化值函数的关键工具。

### 深度Q网络

深度Q网络是基于Q学习的深度强化学习算法，通过深度神经网络对Q值进行逼近，从而在高维度、复杂的状态空间中实现有效的学习[28]。DQN在传统的Q学习的基础上，结合了深度神经网络的表达能力，从而扩展了Q学习的应用范围，尤其在图像和视频等复杂输入的任务中取得了显著的成功。

在传统的Q学习中，智能体根据当前的状态和采取的动作来估算Q值函数，表示在状态下采取动作所能获得的最大累积奖励。Q学习的核心是通过以下贝尔曼最优方程来更新Q值函数：

其中：是当前状态和动作的Q值;是学习率，控制每次更新的幅度;是智能体在时刻执行动作后获得的即时奖励;是折扣因子，决定了未来奖励的重要性;是状态下所有可能动作的最大Q值。

DQN使用一个深度神经网络，其中是神经网络的参数[29]。输入是当前状态，输出是所有可能动作的估计 。DQN的更新规则类似于传统的Q学习，利用以下损失函数来计算更新目标：

其中：是经验回放池，存储智能体的历史经验;是目标值，其中是目标网络的参数。

DQN的提出和应用极大地推动了深度强化学习的发展，尤其是在高维输入空间中的应用，如视觉数据处理、机器人控制等任务[30]。图2-3为DQN算法流程图。

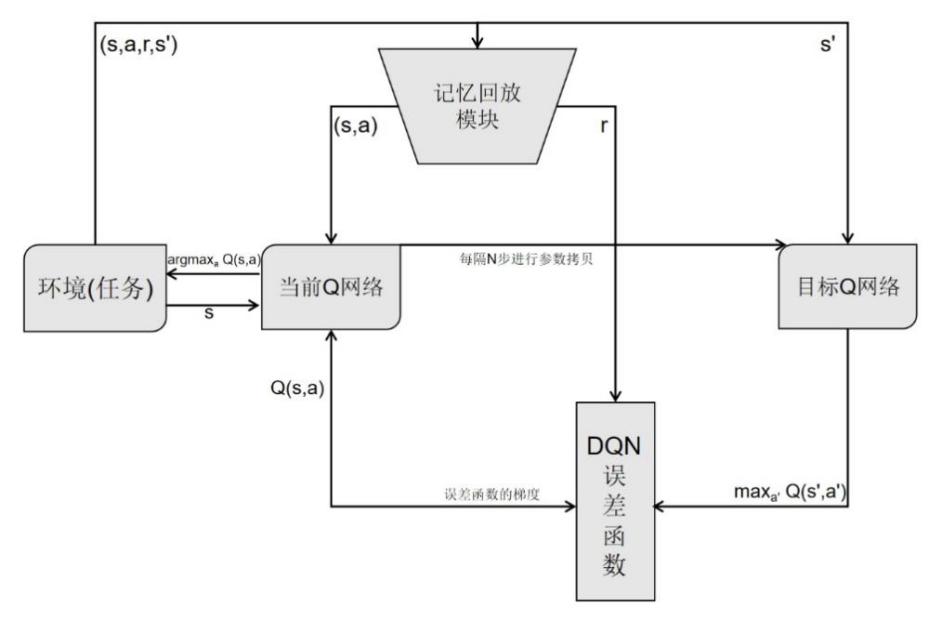


图2-3 DQN算法流程图

Fig. 2-3 DQN algorithm flowchart

### 演员-评论家算法

演员评论家算法结合了策略梯度方法和价值估计方法，其核心思想是将智能体的学习任务分为两部分，首先演员负责生成策略，通过不断调整策略来优化决策，其次评论家负责评估策略的好坏，通过估算状态值函数或状态动作值函数来为演员提供反馈。

在强化学习中，策略通常是通过神经网络进行逼近的，而价值函数 或也通常通过一个独立的神经网络进行逼近[31]。演员-评论家算法的目标是通过两个网络的合作不断提高策略的表现。

评论家的目标是估算价值函数，最常用的是状态值函数，其更新依据是时序差分误差。评论家的更新过程可以通过公式(2-11)来表示：

其中：是TD误差，表示实际回报与预测回报之间的差异，具体定义为：

其中：为当前状态到下一个状态的即时奖励；为折扣因子，控制未来奖励的影响；是当前状态的价值估计；为评论家网络的学习率。

演员的目标是根据评论家的反馈调整其策略，以最大化累计奖励。演员通过策略梯度方法进行更新，策略梯度的基本公式是基于优势函数，它用于度量当前动作相对于基准策略的优劣[32]。

其中：表示优势函数，是状态-动作值函数，表示在状态执行动作后的期望回报，而是状态值函数，表示状态的期望回报。

演员的更新规则通过最大化策略梯度来更新策略参数：

其中：是演员网络的参数；是演员网络的学习率；表示在状态下，演员选择动作的概率；是优势函数衡量动作相对于基准策略的优势。

### 演员-评论家经验回放算法

ACER算法目标是解决强化学习中的问题，包括样本效率、训练稳定性以及收敛速度 [33]。ACER的核心思想是通过从历史经验中优先采样关键样本，避免随机采样造成的低效学习，并通过策略修正与目标网络的引入保持训练过程的稳定性。对于每一对状态动作对，ACER通过演员网络生成动作的概率分布，通过评论家网络计算该状态的价值，然后用它们来指导策略更新。

在常规的演员评论家算法中，策略的更新往往会出现不稳定现象，特别是当策略更新过于剧烈时。为了解决这个问题，ACER引入了策略修正机制，通过限制策略更新的幅度来保持更新过程的稳定性[34]。公式(2-15)为ACER使用剪切策略梯度来限制每次策略更新的幅度：

其中：是学习率；是限制策略更新的范围。

在ACER中，引入目标网络以解决网络更新不稳定的问题。目标网络和是从主网络和的副本中复制出来的，这样可以避免在更新过程中产生过大的参数波动，进而提升学习的稳定性。目标网络的更新规则为：

其中：是软更新的参数，设置为小于1的值，表示目标网络的更新速度较慢。

梯度裁剪在训练过程中发挥作用，梯度裁剪通过限制梯度的大小来防止“梯度爆炸”发生。设定一个阈值，如果梯度的L2范数大于，则对梯度进行裁剪：

其中：是计算得到的梯度，是梯度的L2范数，是裁剪后的梯度。

在ACER算法中，优化目标是通过最大化策略梯度和最小化值函数的误差来实现的[35]。ACER的策略梯度损失函数计算公式如下：

其中：是优势函数，表示在状态下采取动作相较于当前策略的优势。

评论家的损失函数是基于时序差分（TD）误差计算的，用于更新值函数：

其中：是时序差分误差，表示智能体在状态下选择动作时产生的估计误差。

ACER的训练过程可以简要描述为以下步骤：

1.环境交互：智能体与环境交互，并根据策略生成状态动作对。

2.经验回放：将状态、动作、奖励等信息存入经验池。

3.数据采样：从经验池中采样数据，利用TD误差调整优先级。

4.策略更新：基于策略梯度计算损失函数，更新演员网络的参数。

5.评论家更新：利用TD误差更新评论家网络的值函数。

6.梯度裁剪：对网络的梯度进行裁剪，避免梯度爆炸。训练过程流程图见图2-4。

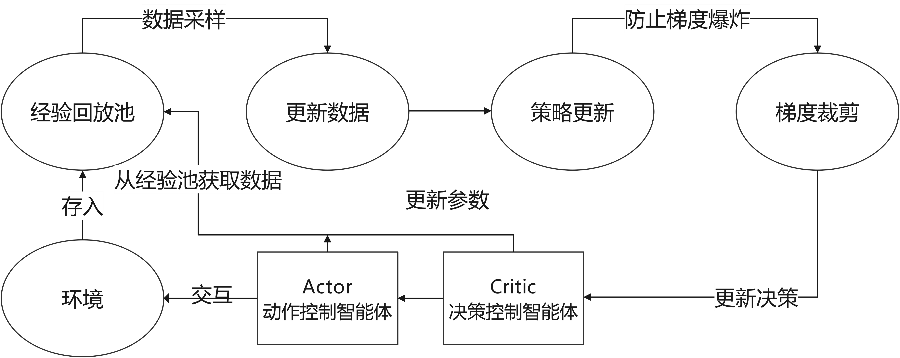


图2-4 ACER算法训练流程图

Fig. 2-4 ACER algorithm training flowchart

ACER算法结合了演员-评论家框架、经验回放、目标网络和策略修正等技术，极大地提高了强化学习模型在复杂环境中的稳定性和样本效率。尽管ACER算法在大多数问题上已取得显著效果，但在面对更复杂和高维的任务时，仍需要进一步优化和调整，特别是在资源受限的情况下。

## 经验回放与优先级经验回放

经验回放是深度强化学习中用于改善样本效率和提高算法稳定性的核心技术之一。其基本思想是将智能体与环境交互过程中产生的状态、动作、奖励、下一状态四元组存储在经验池中，然后从中随机采样一批样本进行训练，而非仅使用当前的交互数据[36]。经验回放能够打破训练样本之间的时序相关性，减少数据的相关性，从而使得训练更加稳定，避免了策略更新中的偏差。经验回放的基本步骤如下：

1. 在每个时间步，智能体从环境中获取四元组。
2. 将该四元组存储到经验池中，经验池大小为。

3. 从经验池中随机抽取一个小批量样本，并用它们进行训练。经验回放流程见图2-5。

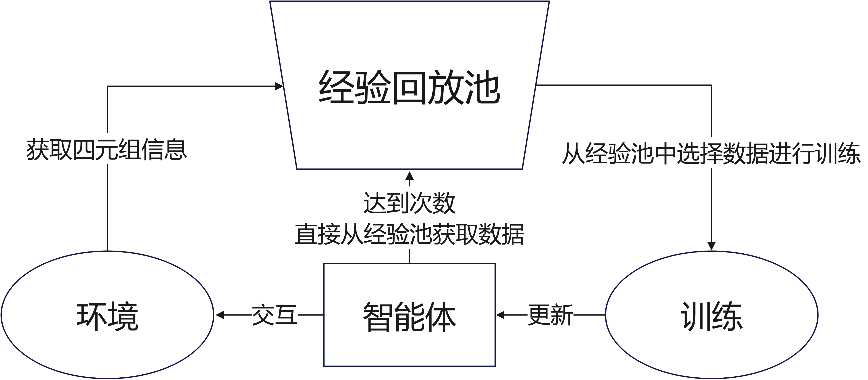


图2-5 经验回放流程图

Fig. 2-5 Experience replay flowchart

训练目标是通过最小化时序差分误差来更新值函数或策略，具体为：

其中：为当前策略下的状态动作值函数，为折扣因子，为基于当前状态的最优动作选择策略。

尽管经验回放能够减少样本之间的相关性并提高样本效率，但它也有一些局限性。例如，在大规模问题中，环境反馈的样本可能包含大量冗余或无关的信息，导致训练过程中的学习效率低下[37]。

为了克服传统经验回放方法的局限性，优先级经验回放为每个存储的经验样本分配一个优先级，使得具有较大时序差分误差的样本得到更高的采样概率。优先级经验回放的关键是如何评估经验的重要性，是基于每个经验样本的时序差分误差来设定优先级。经验样本的时序差分误差定义为：

其中：是经验的TD误差，表示当前Q值函数估计与实际获得的回报之间的差异。为了避免优先级为零的情况，通常在优先级计算中引入一个小常数，即：

其中：是样本的优先级，是一个小的常数，防止某些样本的优先级为零。

优先级经验回放主要通过以下步骤操作，首先是采样，根据每个样本的优先级，采用概率从经验池中进行采样。样本的采样概率由公式(2-23)给出：

其中：是样本的优先级，是控制优先级采样力度的超参数。通过调整，可以控制优先级经验回放在采样中的强度，时为普通的随机采样，时则最大化利用高优先级样本。其次是重要性采样，由于某些样本的采样概率较大，会引入偏差。为了修正偏差，需要引入重要性采样，在计算每个样本的更新时，通过对采样样本加权，修正优先级采样导致的偏差。权重的计算公式为：

其中：是经验池的大小，是样本的采样概率，是一个超参数，控制着重要性采样的调整力度。通过的调整，可以逐渐减小重要性采样的影响，以使得训练过程更加稳定。优先级经验回放思路流程图见图2-6。

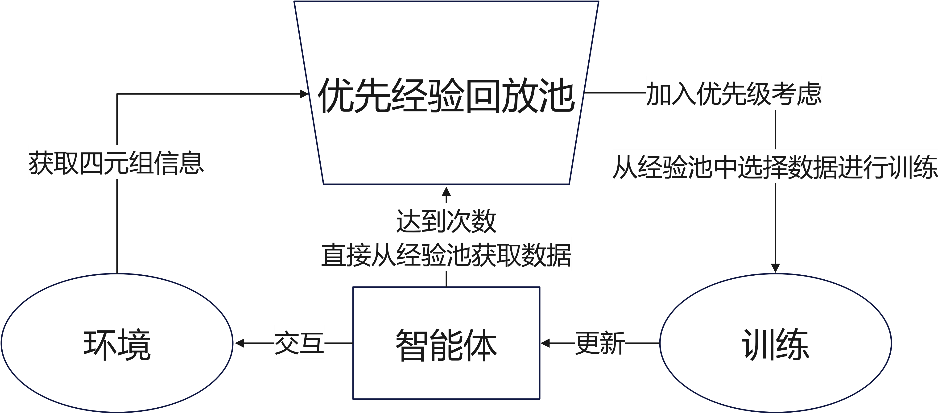


图2-6 优先级经验回放流程图

Fig. 2-6 Priority Experience Playback Process Diagram

在训练过程中，目标是最小化损失函数，公式2-25的函数目标函数是基于优先级和重要性采样加权后的损失函数：

优先级经验回放是传统经验回放的一种有效扩展，能够解决随机采样方式中低效和数据冗余的问题。通过为每个样本分配一个优先级，智能体能够更加高效地学习，同时保证了训练过程的稳定性[38]。优先级经验回放的实施也带来了额外的计算复杂度，例如需要计算每个样本的优先级和TD误差，同时也需要适当地调整采样和重要性采样的超参数。这些挑战为未来的研究提供了进一步优化的空间。

## 元学习模型

元学习，又称为“学习如何学习”，是一种通过从多个任务中学习经验，使得模型能够在遇到新任务时迅速适应并做出有效决策的学习框架[39]。它旨在通过对不同任务的学习过程进行总结和归纳，获取一种能够快速适应新任务的能力，从而显著提高学习效率和泛化能力。元学习适用于强化学习中，在面对多种任务、动态环境或复杂决策时，提供了一种能够快速迁移和调整的策略[40]。

元学习的核心思想是通过从多个任务中学习经验，提炼出一种能够适应新任务的学习策略。为了实现这一目标，元学习通常需要在任务空间上进行学习，使得模型能够在遇到新任务时，通过少量的训练更新，迅速提升其在新任务上的表现。元学习目标可以通过以下方式来表示：

其中：是任务集合，是在任务上的损失函数，是模型的参数，是优化后的最优模型参数。

元学习算法中MAML是一种模型无关的元学习方法，基本思想是学习一个适用于多任务的初始模型参数，从而使模型在遇到新任务时能够通过少数的梯度更新步骤快速达到较好的表现[41]。MAML的目标函数可以通过以下公式表示：

其中：是任务的损失函数，是学习率，是任务的梯度。

MAML通过对所有任务上的损失进行求期望，从而找到一个能够适应多个任务的初始参数。这个过程使得模型能够在经过少量梯度更新后，快速适应新的任务。MAML算法示意图见图2-7。

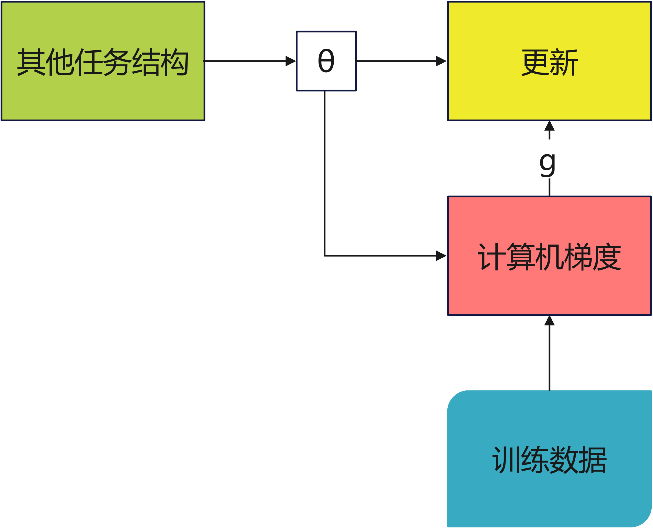


图2-7 元学习算法示意

Fig. 2-7 Schematic of Meta Learning Algorithm

与 MAML 不同，Reptile算法通过逐步调整模型的参数来进行元学习，而不是直接计算每个任务的梯度更新。Reptile的更新公式如下：

其中：是当前的模型参数，是在任务上训练后的参数，是更新步长。

通过多次任务训练，Reptile使得模型的参数逐渐趋向于一个适合大多数任务的参数，从而实现快速适应新任务的目标。

FOMAML是MAML的简化版本，去除了二阶导数的计算。传统MAML需要计算每个任务上损失函数的二阶梯度，这会显著增加计算成本，而 FOMAML通过只使用一阶梯度来近似二阶梯度，从而降低计算复杂度。FOMAML的优化公式为：

在强化学习中，元学习的应用主要体现在任务迁移和快速适应。在传统的强化学习方法中，智能体通常需要对每一个新任务进行完全的训练，而元学习能够通过从多个任务中学习共享知识，使得智能体在面对新任务时能够快速调整策略[42]。元学习的方法通过减少任务间的训练成本，显著提高了样本效率和训练速度。

## 本章小结

本章围绕强化学习及深度强化学习的理论基础进行了详细阐述，分析了相关模型及其发展，并介绍了与之相关的核心算法和技术，进一步为后续研究提供了理论支持。

首先，强化学习作为一种基于试探与反馈的学习方式，已经发展成为智能体决策的核心方法之一。本章首先介绍了强化学习的核心模型马尔可夫决策过程以及贝尔曼方程，通过这些数学框架，明确了状态空间、动作空间、奖励函数等关键组成要素，展现了强化学习的决策过程和算法优化的基本原理。

紧接着，本章对深度强化学习进行了深入探讨，通过将深度神经网络与Q学习结合，DQN成功解决了传统Q学习在处理高维空间时的瓶颈问题，极大地扩展了强化学习的应用范围。而演员评论家算法则通过将策略和价值函数结合，提出了一个更加稳定的策略优化框架。对于每个方法，本文给出了详细的公式推导和计算流程，突出了各自的优缺点和适用范围。

在强化学习算法的优化方面，本章讨论了经验回放和优先级经验回放的核心思想，重点介绍了如何通过优先级的样本选择来提高样本效率，加速智能体的训练过程，避免低质量样本对学习的干扰。

此外，元学习作为一种能够实现跨任务学习的技术框架，也在本章中得到了详细讨论。本章阐明了元学习在强化学习中的应用前景，尤其是在复杂任务和环境中，如何利用元学习提升智能体的自适应能力和泛化能力。

通过本章的研究，为后续的算法设计和实验研究提供了坚实的理论基础，也为改进传统ACER算法及其在离散动作控制任务中的应用奠定了基础。

# 基于自注意力机制的优先级经验回放演员评论家算法

## 引言

本章将详细介绍提出的创新性A2CPER算法。本章将重点阐述自注意力机制和优先级经验回放等一系列技术如何在ACER框架中融合并实现的过程。

## 算法框架设计

本节将详细介绍A2CPER算法的框架，包括每个模块的设计原理、算法流程及其如何互相协作以提高学习效率和稳定性。

### 算法架构概述

本次研究在A2CPER算法这一框架中引入了以下技术：

1.自注意力机制：该机制使得算法能够在处理历史经验时动态关注重要的状态特征，从而避免冗余信息的干扰，提高策略学习的效率。

2.优先级经验回放：通过根据经验样本的TD误差动态调整其优先级，确保在训练中更多地采样对策略优化有较大影响的样本，从而提高样本利用效率。

3.目标网络软更新：通过目标网络软更新的方式，使新的方式无法直接对参数进行更新，更好的防止学习参数崩溃。A2CPER算法的结构示意图如图3-1。

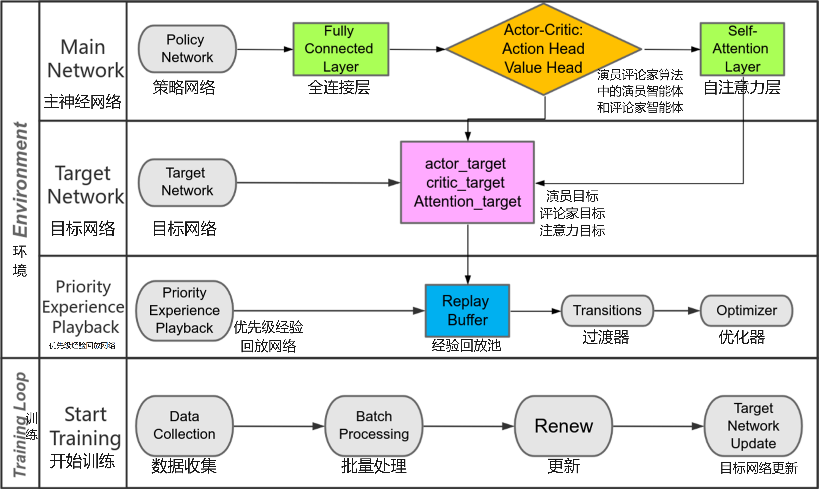


图3-1 A2CPER算法结构图

Fig. 3-1 A2CPER Algorithm Structure Diagram

### 环境交互与经验采样

A2CPER算法首先通过与环境的交互收集经验样本，在每一个时间步，智能体根据当前的状态选择一个动作，并根据该动作与环境的交互获得奖励和下一个状态。该经验样本的四元组被保存到经验回放池中，用于后续的训练。

智能体的动作选择遵循贪心策略，其中有一定的概率进行探索，其余时间则选择具有最大期望回报的动作，通过以下策略来表示：

其中：是当前状态下动作的价值函数。

### 优先级经验回放池模块

在A2CPER中，每个经验样本的优先级根据其对应的TD误差来决定。TD误差反映了智能体在进行决策时，当前策略与实际回报之间的差距，定义为：

其中：是TD误差，是评论家网络对状态的估值，是即时奖励，是折扣因子。每个经验样本的优先级定义为TD误差的绝对值加一个小常数：

然后，优先级经验回放池将每个经验样本的优先级存储在池中，并根据这些优先级来加权选择样本进行训练。A2CPER能够优先选择对策略优化影响较大的经验，避免无效数据对训练的干扰。

### 自注意力机制模块

在A2CPER算法中，自注意力机制被应用于演员网络的输入部分。自注意力机制可以根据状态序列的上下文信息动态调整每个状态的权重，从而使得智能体能够更加聚焦于那些对策略优化最为关键的状态信息。具体的计算流程为：

1.将状态序列转换为查询（Query）、键（Key）和值（Value）表示。

2.计算查询和键之间的相似度并使用softmax进行归一化，得到注意力权重。3.使用注意力权重对值向量进行加权求和，得到新的状态表示。

在A2CPER的应用中，注意力机制帮助演员网络在生成动作策略时，能够重点关注那些直接影响决策的状态特征，减少无关信息的干扰，提高有效性。

### 演员-评论家训练模块

演员网络和评论家网络的训练过程是A2CPER算法的核心。演员网络的目标是根据当前状态生成一个最优的动作策略，通过最大化未来奖励来更新参数。评论家网络则通过最小化当前状态值与目标值之间的均方误差来进行更新，从而为演员网络提供反馈。在A2CPER中，演员和评论家网络的目标函数分别为：首先是演员的目标函数：

其次是评论家的目标函数：

评论家的目标是最小化当前状态的值与目标值之间的误差，目标值通过以下公式计算：

A2CPER算法的流程图如图3-2所示。

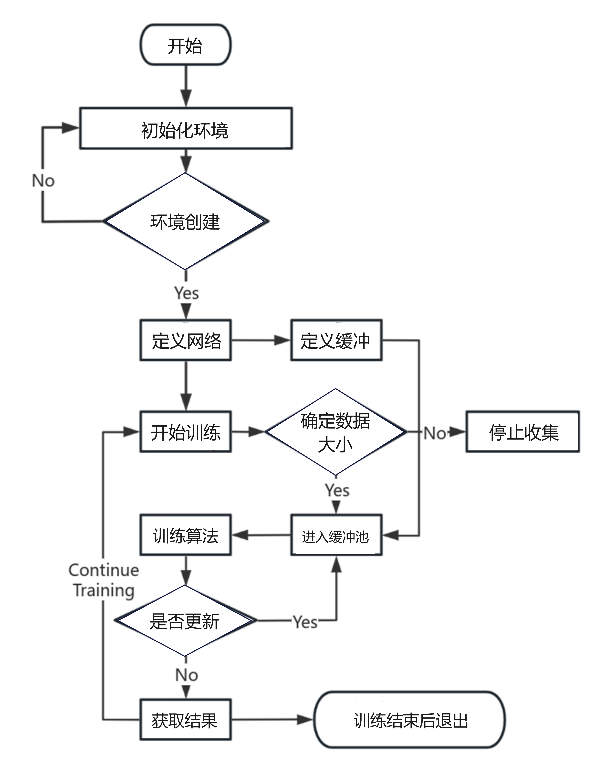


图3-2 A2CPER算法的流程图

Fig. 3-2 Flow chart of A2CPER algorithm

## 自注意力机制的引入

在深度强化学习中，自注意力机制在离散动作控制任务中，智能体需要处理环境中大量的状态特征，而这些特征之间可能存在复杂的相互依赖关系。为了有效捕捉这些特征之间的关联性，本节介绍了在A2CPER算法中如何引入自注意力机制，和自注意力机制如何通过对输入信息的加权处理，以及自注意力机制如何实现和它的具体优势。

### 自注意力机制概述

自注意力机制的核心思想是通过查询（Query）、键（Key）和值（Value）三者之间的关系来计算每个元素的加权表示。其中查询代表了希望聚焦的目标；键与查询进行匹配，用于评估信息的相关性；而值则是最终传递给下一层的实际信息。自注意力机制通过计算查询与键之间的相似度，来为每个值分配不同级别的权重，进而决定哪些信息值得关注[43]。自注意力的计算过程可以用以下的公式表示：

其中：为注意力函数，计算值的加权和；为点积测量相似性；是关键向量的维数；为缩放；softmax函数产生表示输入元素上下文表示的输出。

在本次研究的模型中特别采用了Xavier初始化，这种优先级排序对于放大具有可衡量结果的影响，对提高模型性能至关重要。相反，不重要的任务做出贡献或与次优结果相关的操作会按比例减少注意力权重。注意力奖励公式旨在反映这些原则，并针对以任务为中心的权重分配进行了优化：

其中：表示Q与的矩阵乘法；将查询转换为适合后续操作的空间；是与K对应的权重矩阵表示其转置；是向量与Q查询一起，将确定对值的注意力分布；此操作表示转换后的查询与转置后的转换后的键之间的点积；表示键向量的维数。

因此，公式可以理解为：注意力得分是转换后的查询和键之间的缩放点积运算的结果。这些分数随后在softmax步骤中用于获得最终的注意力权重，这些权重应用于注意力机制中的值V。自注意力机制示意图如图3-3。

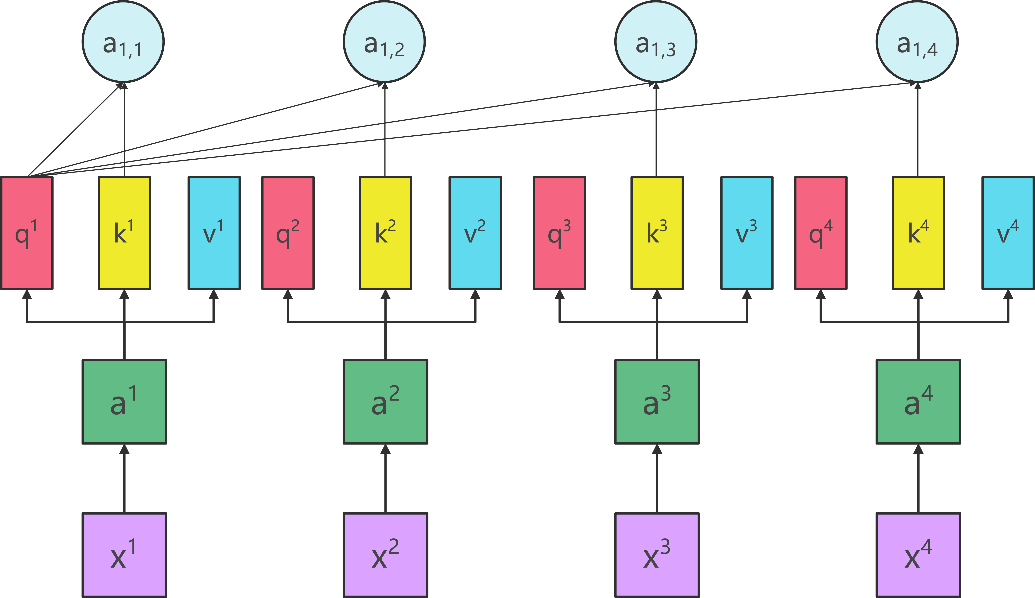


图3-3 自注意力机制示意图

Fig. 3-3 Schematic diagram of self attention mechanism

### 自注意力机制的具体实现

首先将状态映射至查询、键和值，A2CPER首先通过全连接层对输入的状态信息进行线性变换，得到查询、键和值三个向量。这三个向量分别用于描述当前状态的上下文信息及其对决策的影响。这些变换通过训练得到的可学习参数矩阵实现，具体计算公式如下：

其中：为输入的状态向量；、、分别为查询、键和值的权重矩阵。

然后要计算注意力得分，计算它们之间的相似度。自注意力机制通过计算查询和键之间的点积来确定当前状态特征的相关性，并将其作为加权的依据。为避免高维空间中梯度爆炸或消失的问题，对查询和键的点积进行缩放处理：

其中：表示计算得到的注意力得分矩阵，是键向量的维度。该公式通过对点积结果进行缩放，避免了在高维空间中计算时可能出现的数值不稳定性。

接下来是注意力得分归一化，为了使得注意力得分的权重能进行有效比较并转化为概率，利用softmax函数对注意力得分进行归一化处理。通过softmax操作，注意力得分转换成一个概率分布，表示每个状态特征的重要性。

其中：为归一化后的注意力权重，它确保了每个状态特征的权重之和为1。

最后是加权和值计算，通过归一化的注意力权重，对值向量进行加权求和，得到新的状态表示见公式(3-13)。

这个加权和作为网络的输出，提供了经过自注意力机制优化后的状态表示。经过加权后的状态表示更能够反映出当前决策任务中最为关键的特征，为策略网络和价值网络的后续计算提供了更加准确的输入。

### 引入自注意力机制的优势

引入自注意力机制后，A2CPER算法在多个方面实现了优化，具体的优势体现在以下几个方面：

提升关键特征的识别能力：在传统的强化学习算法中，状态空间中的所有特征往往被同等对待，这可能导致信息冗余或不相关的特征干扰策略优化。而通过自注意力机制，A2CPER能够自动识别出哪些特征对于当前决策最为关键，从而重点关注这些特征，避免了冗余特征的干扰。

增强状态间的依赖建模：在离散动作控制任务中，当前状态与过去的状态可能存在重要的依赖关系。自注意力机制通过建模这些全局依赖关系，使得A2CPER能够更好地理解和利用历史信息，从而提高决策的长远性和鲁棒性。

自适应调整特征关注度：不同的任务和环境可能对状态特征的关注度要求不同。自注意力机制使得A2CPER能够在训练过程中动态调整特征的权重，使得智能体能够根据环境变化灵活地优化策略。这种自适应的能力，使得A2CPER在复杂和动态环境下的表现尤为突出。

## 优先级经验回放技术的引入

为了解决重要的经验样本被忽视问题，本节介绍A2CPER如何引入优先级经验回放机制，以及如何加权、计算已经如何调用和这一机制有什么优势。

### 优先级经验回放机制概述

优先级经验回放的核心思想是，通过衡量每个经验样本的重要性来决定其被抽取的概率。优先级经验回放根据经验样本的TD误差来赋予样本一个优先级，TD误差较大的经验样本将具有较高的优先级，并因此更容易被抽取用于训练。TD误差反映了当前智能体对状态-动作对价值的预测误差，通常可以表示为：

其中：为时间步的TD误差；为即时奖励；为折扣因子；为下一个状态动作对的价值；为当前状态动作对的估计值。

TD误差的绝对值较大，意味着智能体的当前策略在该状态下的预测与实际奖励之间的差异较大，进而需要更多的关注和更新。

### 优先级经验回放技术的具体实现

为了提升学习效率并加速收敛，A2CPER采用了一种基于TD误差的优先级分配方法，高TD误差的经验被赋予较高的采样优先级。

首先进行经验存储与优先级分配，A2CPER算法与传统的经验回放方法类似，维持一个经验池，该池用于存储智能体与环境交互过程中获得的状态、动作、奖励、下一个状态及其对应的TD误差。每一条经验存储为一个四元组：

其中：为当前状态；为当前动作；为即时奖励；为下一状态。

接下来进行优先级的分配，经验的优先级则根据其对应的TD误差的绝对值来计算。较大的TD误差意味着该经验对学习任务有较大的贡献，因此该经验的优先级应该较高。经验优先级的计算公式为：

其中：表示当前经验的优先级；是TD误差；用于避免优先级为零的情况。

然后进行优先级经验回放的采样，在A2CPER中，经验回放池中的每一条经验都有一个相应的优先级，优先级较高的经验会被优先采样。A2CPER采用了加权随机采样方法，使用经验的优先级作为概率分布的权重，按照这个权重概率采样经验。在采样时，为了确保每次采样能够反映优先级，A2CPER算法采用了如下的采样过程：

1.计算每条经验的采样概率：

其中：表示归一化后的采样概率；是当前经验的优先级；是一个控制优先级影响程度的超参数，通常。

2.采样：根据归一化后的优先级概率分布，从经验池中随机选择一批经验。通过这种方式，具有较大TD误差的经验会被更频繁地选中，从而加速对策略有重要影响的部分进行训练。

3.重要性采样与偏差校正：为了减少采样性偏差，A2CPER引入了重要性采样权重来进行偏差校正。每个采样样本的重要性采样权重由以下公式给出：

其中：是样本的采样权重；是经验池中的样本数；是一个控制重要性采样权重修正的超参数，通常。当接近1时，权重的修正效果最大，从而消除偏差；当接近0时，重要性采样权重的修正较小，偏差较大。

4.权重更新：在A2CPER中，权重更新步骤结合了经验的优先级、采样概率以及重要性采样权重。在策略网络和价值网络的更新中，每个经验的更新都将乘以其相应的权重。具体的损失函数形式为：

其中：为TD误差；为经验的采样权重。

5.经验池更新：由于A2CPER是基于TD误差更新的，因此经验池中的TD误差会随着训练的进行而变化。每当一条经验被采样并用于更新策略时，都会根据当前的TD误差重新计算经验的优先级。然后，使用更新后的优先级调整其在经验池中的位置。

### 优先级经验回放技术的优势

优先级经验回放的引入，极大地提高了A2CPER算法的训练效率与学习质量。A2CPER在训练过程中展现了以下优势：

加速学习过程：通过优先选择对策略优化贡献较大的样本进行训练，A2CPER能够更快速地收敛，减少不必要的训练周期，提高训练效率。

提高样本利用率：相较于均匀经验回放，优先级经验回放可以更加有效地利用每个经验样本，避免无效样本的重复训练，减少训练资源的浪费。

提升模型稳定性：优先级经验回放可以减少训练过程中的波动性，使得策略优化更加平稳。通过加强对高TD误差样本的关注，A2CPER能够有效减少策略的偏差，提高模型在复杂任务中的稳定性和泛化能力。

## 目标网络更新机制的引入

目标网络常用于提高算法的稳定性。本节介绍在A2CPER算法中，如何引入目标网络机制，用以改善训练过程的稳定性和收敛速度，以及产生的优势。

### 目标网络更新机制概述

在A2CPER算法中策略更新的关键是策略值的技术。目标网络是一个与当前网络结构相同的网络，但它的参数更新频率较低，通过定期的软更新或硬更新方式与当前网络同步。

### 目标网络更新机制的具体实现

在A2CPER算法中，目标网络的引入首先体现在价值函数网络和策略网络的训练过程中。目标网络会在一定的时间间隔内进行更新，而不是在每个训练步骤中都进行更新。

本次研究引入的是软更新方式，软更新使用一个介于目标网络和当前网络之间的过渡系数，逐步调整目标网络的参数，使得目标网络逐步接近当前网络的参数，而不是完全替换。软更新的公式如下：

其中：为软更新的系数，通常设置为接近于0的小值，如0.001。

具体的实现首先是价值网络的目标值生成，在标准的AC算法中，Critic负责对当前策略进行价值评估，即估计每个状态的价值或每个状态动作对的Q值。而在使用目标网络的情况下，Critic不直接使用当前网络生成的值作为目标，而是使用目标网络生成的值作为训练的目标。目标网络的引入使得Q值更新更加稳定，减少了由于价值估计误差引起的更新震荡。在计算Q值时，目标网络通过以下公式生成目标值：

其中：表示目标网络对下一个状态动作对的Q值估计；是即时奖励；是折扣因子。

其次是策略网络的稳定训练，A2CPER算法中，策略网络的训练不直接依赖于当前网络的输出，而是使用目标网络生成的期望价值作为目标进行策略更新。策略网络的目标可以通过以下损失函数进行优化：

其中：是优势函数，表示当前策略相对于目标网络产生的估计值的优势。

### 目标网络更新机制的优势

引入目标网络后，A2CPER算法可以享受到以下几方面的优势：

训练稳定性：目标网络使得价值函数估计更加稳定，减少了由于目标值频繁波动引起的更新不稳定性，尤其在价值网络的训练过程中，软更新进一步减缓了目标网络的变化，避免了过拟合。

加速收敛：通过平滑的目标值更新，目标网络的引入能够加速策略优化过程。相较于直接使用当前网络输出作为目标值，目标网络能够在长时间内稳定提供参考，从而提高学习效率。

减少估计误差的传播：由于目标网络的更新周期较长，估计误差的传播被有效控制，避免了误差在多轮更新中积累放大。

## A2CPER算法的伪代码

表3-1 融合自注意力机制和优先级经验回放的改进ACER算法伪代码

Table 3-1 A2CPER algorithm pseudocode

|  |
| --- |
| 融合自注意力机制和优先级经验回放的改进ACER算法（A2CPER） |
| for episode = 1,M do  设置超参数 |
| env.seed(1), torch.manual\_seed(1) |
| state\_space, action\_space, MEMORY\_CAPACITY, batch\_size  初始化带自注意力机制的策略网络  state\_space, action\_space  optimizer ← Adam( policy.parameters , lr = 0.01)  初始化优先经验回放缓冲区  PrioritizedReplayBuffer( MEMORY\_CAPACITY)  if i\_episode ∈ {1, 2, ..., episodes} |
| state ← env.reset()  for episode 1 to T do  action\_probs ← policy(state) next\_state, reward, done ← env.step(action) |
| 计算TD误差并把四元组存入缓冲区并使用误差值作为优先级 |
| if ReplayBufferSize >= bufferSize |
| 进行样本采样 对每个 idx ∈ indices 执行数据 ← B[idx] |
| End for episode |

## 本章小结

本章详细介绍了基于自注意力机制和优先级经验回放的ACER算法的设计与实现过程。首先，通过引入自注意力机制，有效地增强了策略网络对重要信息的聚焦能力，使得算法在动态环境下能够更加精准地选择和优化策略。在强化学习任务中对当前状态与动作的决策起到了至关重要的作用，尤其是在复杂状态空间和高维特征的环境下。

其次，针对传统ACER算法中经验回放的不足，引入优先级经验回放算法。具体的实现中，详细讨论了如何将该机制与A2CPER算法有效结合，通过动态调整样本的优先级，实现了样本效率和学习速度的双重提升。

最后，本章还对目标网络的软更新策略进行了介绍，旨在进一步提升算法的稳定性，避免强化学习中常见的训练不稳定问题。通过目标网络的平滑更新，A2CPER算法能够在面对环境的动态变化时保持较好的稳定性与适应性。

# 基于元学习和自适应奖励的优先级经验回放演员评论家算法

## 引言

在强化学习的应用中，尤其是离散动作控制问题，智能体往往面临着较高的样本复杂度和训练时间，特别是在环境的变化导致任务不断更新时。基于传统强化学习框架（如A2C、DQN等）进行学习的智能体，往往需要通过大量的探索和经验积累才能收敛到一个较优的策略，而这对于一些需要迅速响应的动态环境而言，显然无法满足其需求。

为了解决这一问题，本项研究提出了基于元学习和自适应奖励的优先级经验回放演员评论家算法，引入元学习方法和自奖励机制来优化智能体的学习过程，从而在减少样本复杂度、提高学习效率的同时，更好地应对环境变化带来的挑战。本章主要介绍了提出的算法的框架，并说明了算法如何具体的实现。

## 算法框架设计

在MACPER算法中，元学习和自奖励机制被紧密地与ACPER算法的核心思想进行结合，主要目的是增强策略学习和经验重放过程中的优化能力。本节将介绍算法框架的设计及其每个部分如何与ACPER算法结合。

### 算法框架概述

MACPER算法框架基于ACPER算法，并在其中融入了元学习和自奖励机制，以解决在离散动作控制问题中的挑战。总体框架包括以下几个主要部分：

1. 环境初始化与任务选择：在MACPER中，智能体通过元学习任务集合来快速适应新任务，同时每个任务都可以是ACPER框架中进行训练的环境。

2. 元学习框架：通过元学习方法，智能体在多个任务上进行训练，从而获得快速适应新任务的能力。在每个新任务中，智能体能够利用先前任务的经验加速学习。

3. 自奖励机制：在训练过程中，智能体通过自奖励机制调整奖励信号，使得智能体能够更加高效地进行探索。自奖励机制不仅对环境奖励进行修正，还通过动态调整奖励的幅度来优化策略学习过程。

4. ACER核心优化：在加入元学习和自奖励机制的框架下，融合了AC框架和优先级经验回放机制，在策略优化和优先级经验回放的选择中，智能体将依据元学习的结果以及自奖励机制提供的动态奖励信号进行调整。

### 环境初始化与任务选择

在MACPER算法中，环境初始化和任务选择是智能体学习的基础步骤，直接影响其学习效率与泛化能力。任务选择的主要目标是使智能体能够通过多任务训练提高在新任务上的适应能力。

首先任务选择具有多样性，确保智能体能够接触到不同难度的任务环境，学习到不同的策略。设当前智能体在任务中的表现为，任务选择策略基于任务间的相关性、奖励反馈等因素，选择具有较高迁移潜力的任务。

其次注意的是根据智能体在先前任务中的表现，动态选择具有适当难度的任务。任务难度可以通过智能体在任务中的累计奖励来衡量：

其中：是任务持续时间；是在每个时间步的奖励。

接下来在引入元学习后，任务选择不仅关注当前任务的难度，还会根据智能体对先前任务的学习成果进行优化。在MACPER中，智能体通过从不同任务中学习到的知识来加速新任务的学习。

其中：是智能体在任务上的奖励函数；是元学习模型的最优参数。

最后是任务适应与奖励机制，智能体基于自奖励机制调整其在不同任务中的表现，奖励信号被动态调整，以确保智能体能快速适应任务：

其中：和是超参数；是智能体对任务的适应性指数，用来量化当前任务对智能体学习进度的贡献。

### 元学习框架

元学习的核心是通过对先前任务的学习经验进行总结，使得智能体能够在面对新任务时，快速调整其策略或参数，从而加速学习过程。元学习框架包括：

任务的元训练：通过多个任务的训练，智能体学习如何有效地利用过去的经验来加速新任务的学习过程。这要求智能体能够从多个任务中提取共享的知识，并将其应用于新任务中。

学习速率的自适应调整：在训练过程中，元学习框架使智能体能够自适应地调整学习速率，从而在遇到新任务时，快速有效地优化策略。通过对历史任务的训练，智能体能够估计其在当前任务上的学习能力，并据此调整学习速率。

### 自奖励机制

自奖励机制需要在策略更新过程中额外计算并融入内在奖励。

1. 初始化：在训练开始时，智能体通过初始化自奖励参数（和）来控制奖励信号的生成。同时，策略网络和价值网络也被初始化，准备进行学习。
2. 探索过程中的自奖励计算：在每一步决策过程中，智能体根据当前的动作和状态，计算出相应的自奖励。此时，智能体的内在奖励由多样性和稀疏性两个部分构成。
3. 内在奖励的叠加：计算出的自奖励与外部奖励相加，得到智能体的最终奖励信号：

其中：是智能体在当前时刻接收到的总奖励，它结合了外部环境奖励和自奖励。

1. 策略更新：使用总奖励更新策略网络和价值网络的参数。具体而言，智能体根据策略梯度方法计算损失函数，并通过反向传播优化网络参数。

### ACER核心优化

ACER的一个关键特性是引入了重要性采样修正，以减轻策略更新过程中因策略变化导致的样本偏差。在标准的ACER算法中，智能体的策略会不断更新，这种更新可能导致从回放池中采样的经验与当前策略不再匹配，从而影响训练的效率和稳定性。

在MACPER中，重要性采样修正被与元学习机制相结合，特别是在元训练阶段中。MACPER通过元学习优化过程调整经验采样的策略，从而使智能体能够根据任务的不同难度进行自适应的策略更新。这一优化策略能显著提升智能体在面对新任务时的收敛速度，同时减少训练过程中的偏差问题。在实践中，重要性采样修正公式为如下形式：

其中：表示在当前策略和历史策略下的比率，控制更新的方向。和分别为当前和历史策略的动作概率分布。MACPER算法流程图如图4-1所示。

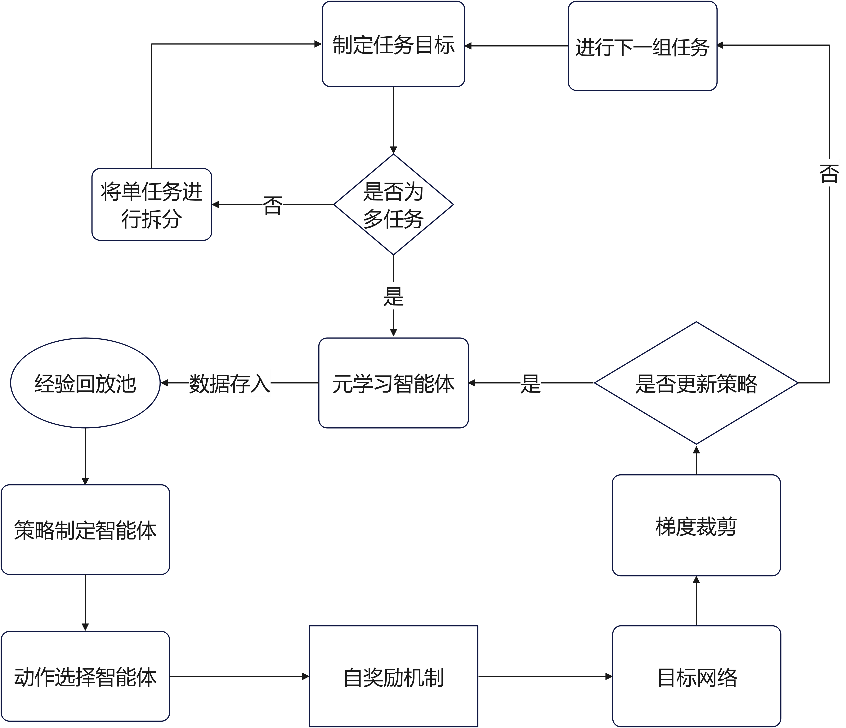


图4-1 MACPER算法流程图

Fig. 4-1 MACPER algorithm flowchart

## 元学习框架的引入

本节介绍元学习框架如何具体引入，来增强智能体在不同任务之间的泛化能力和适应性。并且在MACPER算法中具体实现这个部分，并介绍了优势部分。

### 元学习框架概述

元学习通过从多个任务中积累经验，智能体能够捕捉到任务间的共性和差异，从而快速适应新任务的挑战。首先元学习学习任务间的共享结构，通过在多个任务上进行训练，使得智能体能够提取任务之间的共有信息。其次要快速适应新任务，在面对新的任务时，通过少量的经验（例如少量的训练样本）迅速优化策略，从而避免重新开始训练。

### 元学习框架的具体实现

在元学习框架被集成到MACPER的训练过程中，通过少量的更新步骤使智能体能够在新的环境中快速优化其策略。

（1）任务生成与适应

在MACPER中，任务生成模块负责根据任务的复杂度与目标动态生成训练任务。每个任务包括一个特定的环境配置、奖励函数和目标。元学习框架需要智能体在这些任务上进行训练，学习如何在多任务的环境下进行策略优化。任务生成模块的输出是一个任务集，每个任务都包含了相应的状态空间和动作空间。

每次开始训练时，元学习框架会从任务集合中选择一个任务进行训练。任务的选择可以基于当前任务的难度，或者基于当前模型的学习进度。在每个任务中，智能体会根据任务的环境进行试探学习，从而获得策略更新所需的信息。

（2）内部模型与任务适应

为了实现任务间的快速适应，MACPER算法中的元学习框架引入了一个元策略网络。这个网络通过训练学习跨任务的共享策略参数，使得在新任务中能够快速得到有效的动作选择。

在每个任务上，智能体首先通过现有的策略执行探索，并根据任务的反馈（奖励和下一状态）计算损失函数。损失函数反映了当前策略在任务上的表现，公式如下：

其中：是智能体在任务上的即时奖励。

根据计算得到的损失，使用梯度下降法更新策略网络的参数：

其中：是任务学习的学习率，用于调整每次更新的步长。

（3）元梯度更新

元学习的关键部分是通过元梯度更新策略。元梯度通过计算每个任务的梯度来优化元策略网络的参数。对于每个任务，元学习的目标是通过计算该任务的损失函数梯度，并将其反向传播到元策略网络中，从而更新元策略的参数。元梯度更新公式如下：

其中：是任务集合上的总损失函数，计算方式为所有任务的损失函数的加权和：

该更新过程是通过反向传播进行的，目标是通过梯度下降的方式优化元策略，使得智能体能够更好地适应不同的任务。

（4）内外部策略调整

MACPER中的元学习框架通过内外部策略的协同工作来加速任务适应。外部策略负责在每个任务上进行训练，并通过任务特定的反馈信息来优化策略。内部策略则通过元学习框架快速适应不同任务，每当遇到新任务时，元策略通过微调调整外部策略，达到快速适应新任务的效果。

（5）快速任务适应

通过元学习的训练，MACPER算法能够快速适应新的任务。在每次遇到新的任务时，元策略网络能够通过少量的梯度更新，快速调整策略，确保智能体能够在短时间内掌握新任务的优化方向。元策略网络会根据新的任务特征和奖励信号，迅速通过少量更新步骤提高性能。在此过程中，更新公式如下：

### 元学习框架的优势

元学习框架的核心优势之一是其快速适应新任务的能力。元学习通过在多个任务上共享经验，使得智能体能够基于先前任务的学习，迅速调整策略并适应新任务。这种快速适应的能力得益于元策略网络的设计，能够在遇到新任务时通过少量梯度更新就迅速调整策略，避免了从头训练的时间开销。

在MACPER算法中，元学习框架通过元梯度的更新和跨任务知识的共享，使得智能体能够适应更广泛的任务，提升了其在未知环境中的表现和稳定性。即使面对与训练任务不同的新环境，智能体也能够利用先前任务中的学习经验，较少地依赖于大量的额外训练数据。

## 自奖励机制的引入

本节介绍在MACPER算法中，如何结合自奖励机制，来提升智能体在复杂环境中的自主决策能力，并将该机制在算法中具体实现，并且说明具体的优势。

### 自奖励机制概述

自奖励机制的核心思想是，智能体根据自己的历史决策过程和经验，动态地生成奖励信号，从而引导其在当前状态下选择最优的动作。自奖励机制通过计算智能体的“内部奖励”来替代外部奖励，使得智能体在处理稀疏奖励或无监督奖励的任务时，仍能自主学习。自奖励机制的实现通常涉及以下几个关键步骤。首先是根据当前状态，计算该状态对智能体长期目标的贡献。其次是基于智能体在该状态下的历史表现（例如策略更新、价值函数评估等），动态生成奖励信号。最后是将生成的奖励信号反馈到系统中，作为策略优化的依据。

### 自奖励机制的具体实现

在MACPER算法中，自奖励机制的引入主要体现在两个方面：首先是对智能体的状态进行评估，并根据其历史经验生成自奖励；其次是将这些自奖励信号融入到优先级经验回放中，以进一步改善策略和价值函数的学习。

首先是状态评估与自奖励生成。在每个时间步，MACPER算法根据当前状态生成一个自奖励值，该奖励值不仅依赖于环境中的即时奖励，还受到智能体过去经验的影响。这一过程中，智能体根据任务的不同，计算状态动作对的自奖励。具体公式如下：

其中：表示在时间步时刻，状态的自奖励值；是一个函数，综合考虑了策略网络和价值网络的参数，对当前状态进行加权评估；为当前状态。

其次是自奖励与外部奖励的结合。在很多情况下，环境中的外部奖励与智能体自身的学习目标并不完全一致。因此，MACPER算法不仅会考虑环境提供的外部奖励，还会将生成的自奖励信号与外部奖励信号结合起来共同推动策略优化。算法通过引入一个加权系数来平衡外部奖励和自奖励之间的权重，最终得到每个状态动作对的总奖励信号：

其中：是状态动作对的总奖励，作为优化目标用于更新策略和价值函数；是环境提供的外部奖励；是基于自奖励机制生成的奖励信号；是加权系数，用于控制外部奖励和自奖励之间的相对重要性，通常在训练过程中进行调节。

最后是策略更新与自奖励反馈。在策略更新的过程中，MACPER不仅依赖外部奖励信号，还充分利用了自奖励生成的信号来指导策略的优化。自奖励机制引入后，MACPER的策略更新公式为：

其中：表示基于当前策略参数的目标函数，旨在最大化期望总奖励；是总奖励，包括环境奖励和自奖励；是状态的价值函数，表示在给定策略下智能体从该状态出发的预期回报。

### 自奖励机制的优势

通过引入自奖励机制，MACPER算法不仅能够有效应对环境奖励稀疏和延迟的问题，还可以减少智能体对外部奖励的过度依赖，提升学习效率和任务适应能力。自奖励机制能够为智能体提供更为细致的反馈，帮助其在任务的各个阶段都能够自主调整和优化策略，进而提高算法的性能。

自奖励机制的引入为MACPER算法提供了一种灵活且高效的奖励信号生成方式，增强了智能体在多任务环境中的学习能力和适应性。通过动态生成奖励信号并与外部奖励信号结合，MACPER能够更好地引导智能体优化策略，同时减少对外部奖励的依赖，提高了算法在复杂环境中的表现和稳定性。

## MACPER算法伪代码

表4-1 基于元学习和自适应奖励的优先级经验回放演员评论家算法伪代码

Table 4-1 MACPER algorithm pseudocode

|  |
| --- |
| 基于元学习和自适应奖励的优先级经验回放演员评论家算法伪代码 |
| for episode = 1,M do |
| 设置超参数 |
| state\_space, action\_space, MEMORY\_CAPACITY, batch\_size |
| procedure MACPER |
| 初始化元学习网络 |
| (state\_space, action\_space)state\_space, action\_space |
| 初始化优先经验回放缓冲区 |
| PrioritizedReplayBuffer( MEMORY\_CAPACITY) |
| if i\_episode ∈ {1, 2, ..., episodes} |
| for episode 1 to T do |
| action\_probs ← policy(state) |
| next\_state, reward, done ← env.step(action) |
| 动作 ← 策略网络(状态) 动作采样 ← 样本(动作) |
| 计算TD误差 |

续表(4-1)

|  |
| --- |
| if优先级经验回放缓冲区长度 ≥ 批量大小 then |
| 批量数据, 索引, 权重 |
| for episode 1 to T do |
| 更新策略网络: |
| 状态, 动作, 奖励, 下一状态, 结束标志 |
| Q目标←计算Q目标(奖励, 下一状态, 值函数网络) |
| 策略损失←计算策略损失(状态, 动作, 自奖励, Q目标) |
| 自奖励←计算自奖励(状态, 策略网络, 值函数网络) |
| end for |
| 优化器.zero\_grad() |
| 策略损失.backward() |
| 值函数损失.backward() |
| 优先级经验回放缓冲区.更新优先级(索引, TD误差) |
| end if |
| end if |
| end for episode |

## 本章小结

本章主要介绍了基于元学习与自奖励机制的改进算法（MACPER）的设计与实现。首先，详细阐述了MACPER算法的框架，包括如何将元学习与自奖励机制与ACER算法相结合，以提升智能体在动态和复杂环境中的学习效率。接着，详细描述了算法的各个关键模块，包括环境初始化、任务选择、元学习框架的引入、自奖励机制的实现及ACER核心优化等。

在算法实现方面，通过元学习框架对策略和价值函数网络进行自适应优化。通过对元学习与自奖励机制的引入，MACPER能够在短期内更快地收敛，并且具有更强的适应性和鲁棒性，尤其在高维、连续的任务中展现出了其优势。

# 离散动作控制决策实验与分析

## 引言

本章旨在通过一系列实验对比分析A2CPER和MACPER算法在离散动作控制决策任务中的性能表现。离散动作控制决策问题主要是指智能体以离散的方式控制目标进行任务的完成，要求智能体在有限的动作空间中进行高效决策。离散动作控制决策问题还分为静态性和动态性，是以任务的目标不变，但是中间的任务过程是否改变来进行区分。

本章首先进行A2CPER算法在离散动作静态性控制决策中的实验设计与分析，接下来进行实验验证MACPER算法在离散动作动态性控制决策中表现。具体地，本章将通过多个经典环境进行实验验证，分析不同算法在面对复杂的离散控制任务时的适应能力和效率。

在实验过程中，除了对比实验结果外，还将详细讨论算法的相关性能数据等多个方面，为改进强化学习算法在实际应用中的表现提供实证支持。通过这些分析，可以深入理解如何利用先进的算法架构优化离散动作控制任务的决策过程。

## A2CPER算法处理离散动作静态性控制决策实验与分析

### 实验环境

本实验选择了以下三个经典的离散动作控制环境来评估A2CPER算法的性能。这些环境均来自OpenAI的Gym库，分别代表了不同的控制任务和挑战。通过这些环境的实验结果，可以全面评估A2CPER在不同类型的任务中的表现，尤其是其在处理离散动作决策问题时的优越性。

（1）CartPole-v1（小车倒立摆）

CartPole-v1是OpenAI Gym中一个经典的强化学习环境，它模拟了一个小车和杆子的系统，智能体的目标是控制小车的左右移动，使得垂直的杆子保持平衡。这个任务是一个简单的控制任务，但同时也充满挑战，要求智能体根据状态做出快速、准确的决策。

a.任务的目标：

智能体需要通过控制小车左右移动，尽可能长时间地保持杆子的平衡。杆子倒下或超过某个角度时，任务失败。

b.状态空间：

表5-1 CartPole-v1的状态空间

Table 5-1 The state space of CartPale-v1

|  |  |
| --- | --- |
| 状态空间 | 设定值 |
| 位置 | 范围为[-2.4, 2.4] |
| 速度 | 范围为[-1, 1] |
| 角度 | 范围为[-0.2095, 0.2095] |
| 角速度 | 范围为[-1, 1] |

因此，状态空间的维度为4维，每个维度描述了小车和杆子的具体状态。

c.动作空间：

表5-2 CartPole-v1的动作空间

Table 5-2 The action space of CartPale-v1

|  |  |
| --- | --- |
| 动作空间 | 动作 |
| 0 | 将小车向左移动 |
| 1 | 将小车向右移动 |

因此，动作空间的维度为2维，智能体可以选择左或右移动小车。

d.奖励机制：

表5-3 CartPole-v1的奖励机制

Table 5-3 The reward mechanism of CartPole-v1

|  |  |
| --- | --- |
| 奖励 | 奖励值 |
| 每步持续保持杆子平衡 | 给予+1的奖励 |
| 当杆子倒下或偏离过大 | 任务失败，奖励为0，环境重置 |

CartPole-v1实验模型如图5-1所示。

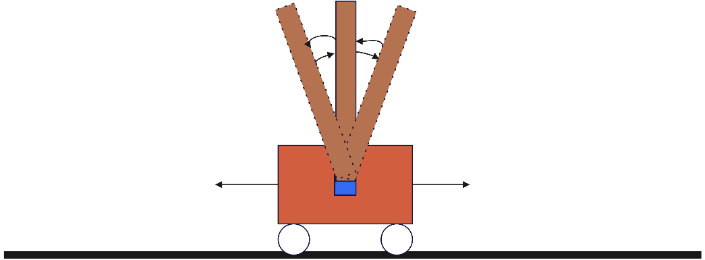


图5-1 CartPole-v1实验模型

Fig. 5-1 CartPtole-v1 experimental model

（2）Acrobot-v1（双摆系统）

Acrobot是一个复杂的控制任务，涉及两个连杆和一个重物的双摆系统。智能体的目标是通过控制两个杆子的运动，使得系统的末端能够达到指定的高度。

a.任务目标：

通过控制加速器的运动，使得末端的杆子达到指定的目标位置，智能体需要在有限的时间内使系统的末端摆臂尽可能接近目标位置。

b.状态空间：

表5-4 Acrobot-v1的状态空间

Table 5-4 The state space of Acrobot-v1

|  |  |
| --- | --- |
| 状态空间 | 设定值 |
| 第一个杆子的角度 | 范围为 |
| 第一个杆子的角速度 | 范围为[-1, 1] |
| 第二个杆子的角度 | 范围为 |
| 第二个杆子的角速度 | 范围为[-1, 1] |

因此，状态空间的维度为4维，描述了两个摆臂的角度和速度。

c.动作空间：

表5-5 Acrobot-v1的动作空间

Table 5-5 The action space of Acrobot-v1

|  |  |
| --- | --- |
| 动作空间 | 动作 |
| 0 | 加一个力使驱动系统逆时针旋转 |
| 1 | 加一个力使驱动系统顺时针旋转 |
| 2 | 加一个力使驱动系统不旋转 |
| 3 | 加一个力使驱动系统强烈旋转 |

因此，动作空间的维度为4维，智能体可以选择四种不同的动作。

d.奖励机制：

表5-6 Acrobot-v1的奖励机制

Table 5-6 The reward mechanism of Acrobot-v1

|  |  |
| --- | --- |
| 奖励 | 奖励值 |
| 结束时末端摆臂达到目标位置 | 给予+1的奖励 |
| 结束时末端摆臂未达到目标位置 | 奖励为0 |

Acrobot-v1实验模型如图5-2所示。

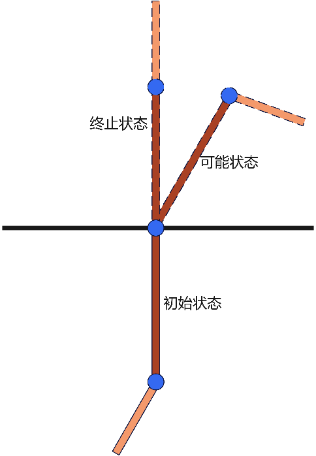


图5-2 Acrobot-v1实验模型

Fig. 5-2 Acrobot-v1 experimental model

（3）MountainCar-v0（山地车）

MountainCar-v0模拟了一个小车在山谷中的行为。智能体的任务是通过控制小车的运动，使其能够爬上山坡并到达目标位置。小车无法直接达到目标位置，必须利用惯性反复来回摆动，积累足够的动能后才能够爬上山坡。

a.任务目标：

通过反复摆动来积累足够的动能，使得小车能够爬上山坡并达到目标位置。智能体需要在有限的时间内到达目标位置。

b.状态空间：

表5-7 MountainCar-v0的状态空间

Table 5-7 State space of MountainCar-v0

|  |  |
| --- | --- |
| 状态空间 | 设定值 |
| 位置 | 范围为[-1.2, 0.6] |
| 速度 | 范围为[-0.07, 0.07] |

因此，状态空间的维度为2维，描述了小车在山谷中的位置和速度。

c.动作空间：

表5-8 MountainCar-v0的动作空间

Table 5-8 The action space of MountainCar-v0

|  |  |
| --- | --- |
| 动作空间 | 动作 |
| 0 | 加速 |
| 1 | 不加速 |
| 2 | 减速 |

因此，动作空间的维度为3维，智能体可以选择三种不同的动作。

d.奖励机制：

表5-9 MountainCar-v0的奖励机制

Table 5-9 Reward mechanism for MountainCar-v0

|  |  |
| --- | --- |
| 奖励 | 奖励值 |
| 每步执行时 | 奖励为-1 |
| 到达目标位置时 | 奖励为+100 |

e.任务挑战：

MountainCar-v0是一个较难的任务，尤其是小车无法直接爬上山坡。智能体必须通过反复利用惯性摆动来逐步积累动能，最终才能成功到达目标位置。这一任务对于强化学习算法在解决延迟奖励问题上的能力进行有效评估。MountainCar-v0的实验模型图如图5-3所示。

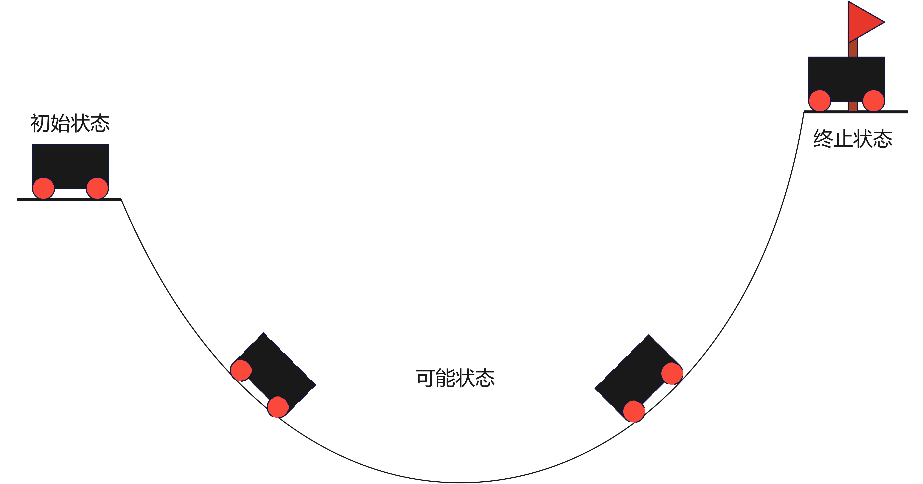


图5-3 MountainCar-v0的实验模型图

Fig 5-3 Experimental model diagram of MountainCar-v0

通过这些环境的实验，可以全面评估A2CPER算法的表现，特别是在处理不同类型的离散性控制任务时的效率、稳定性和收敛速度。每个环境在实验中扮演了不同的角色，测试了算法在简单、复杂和高维状态空间中的表现。接下来的实验将通过这些环境的训练结果，进一步分析A2CPER算法在实际应用中的优缺点。

### 实验代码设置

本实验采用了基于PyTorch的深度强化学习框架，并结合A2CPER算法处理离散动作控制决策任务。本节以下部分是实验中的关键设计部分，包括初始化、模型设置、超参数选择及其优化器的设置、以及训练过程中的注意事项。

（1）环境初始化

实验选择了CartPole-v1、Acrobot-v1和MountainCar-v0三个经典环境来验证算法的效果。在环境初始化时，需要根据任务要求设置合适的随机种子，以确保结果的可重复性。通过gym.make()加载环境后，需要对其进行状态和动作空间的定义。整个过程分为三部分进行实验，具体的实验环境在本章上节中已经给出，每次进行不同的实验，包括进行同一个实验中的不同目的性实验之前，均需要进行环境的初始化。

（2）模型设置

A2CPER算法的模型包括：策略网络、价值函数、网络网络架构和输入输出等。具体的模型架构在第三章有详细介绍。

（3）超参数设置

在A2CPER算法中，合理选择超参数对模型训练的效率和性能至关重要。本次实验所有超参数的选择均是在经历大量数据测试，并且翻阅相关资料之后，共同计算出的结果，保证整个实验的流畅性、完整性、可复现性和科学严谨性。三个实验所设计的重要且影响结果的超参数均由下表给出，如果没在下表中给出，则说明该参数不影响整体实验的流程，并且对实验结果也不会产生影响。以下是常用的超参数：

表5-10 实验1超参数设计表

Table 5-10 Experiment 1 Superparameter Design Table

|  |  |
| --- | --- |
| 超参数名称 | 设定值 |
| 学习率 | learn = 0.01 |
| 折扣因子 | gamma = 0.99 |
| 批量大小 | Size = 64 |
| 经验回放池大小 | replaysize = 10000 |
| 目标网络更新频率 | update\_target = 1000 |
| 折扣因子 | gamma = 0.99 |
| 步长 | episodes = 2000 |
| 最大步数 | max\_time = 200 |
| 最终探索率 | epsilon\_final = 0.01 |
| 探索衰减率 | epsilon\_decay = 0.995 |
| 评估频率 | evaluation = 200 |

（4）优化器设置

使用Adam优化器来优化策略网络和价值函数网络。具体的优化器设置见下表5-11。

表5-11 优化器设置

Table 5-11 Optimizer settings

|  |  |
| --- | --- |
| 优化器 | 设定值 |
| Adam | Adam |
| 学习率 | 0.001 |
| β1 | 0.9 |
| β2 | 0.999 |

### 实验结果以及分析

（1）CartPole-v1的实验

本研究进行的实验经过严格控制，以确保所有参数的一致性，并在所有实验方面遵循最佳原则[44]。使用比较方法评估了各种深度强化学习算法，包括 DQN、Policy Gradient、AC、ACER 和实验提出的 A2CPER 算法。

比较是从三个不同的角度进行的：小车保持平衡的总时间步数、小车保持平衡的平均持续时间以及小车获得的平均奖励值。为了保持严谨性，执行了1000次模拟实验，每次实验都有5个不同的随机种子，并计算平均值进行比较分析。

图5-4到5-6表示了不同算法让小车成功保持平衡的总时间步数。AC 框架内的算法比其他算法整体上更早达到500的目标时间步长。通常这些算法在大约100到200次迭代后会达到性能峰值，而其他算法往往在200次迭代后才达到第一个性能峰值。这一差异凸显了AC算法在初始数据收集方面的出色表现，这得益于其Critic网络的出色数据分析能力[45]。本文提出的A2CPER算法在100次迭代左右就能达到第一个性能峰值。这一加速进步得益于优先经验重放策略，该策略有效减少了不相关采样数据的使用，促进了对高优先级数据的学习，从而加快了性能峰值的达到。

此外，在检查每种算法的性能时，观察到实验过程中的波动。波动程度与表中的数据点数量相对应。值得注意的是，由于结合了自注意力机制，A2CPER算法表现出最小的数据波动[46]。这一添加通过优先关注数据波动显著提高了算法的稳定性。较大的波动通常是由于神经网络参数的过度更新造成的，A2CPER算法通过在目标网络中引入延迟参数更新有效地缓解了这一挑战。

最后观察到，在实验的中后期，使用AC框架的算法的训练水平明显低于其他算法。随后，它们的性能在大约100次迭代后趋于正常化。这种现象主要归因于损失函数中的梯度相关问题。A2CPER通过结合梯度裁剪和加权损失计算方法来优化损失函数，有效地解决了这一挑战。

总体而言，实验揭示了A2CPER算法的优势，特别是在快速收敛、稳定性和增强训练效率方面。

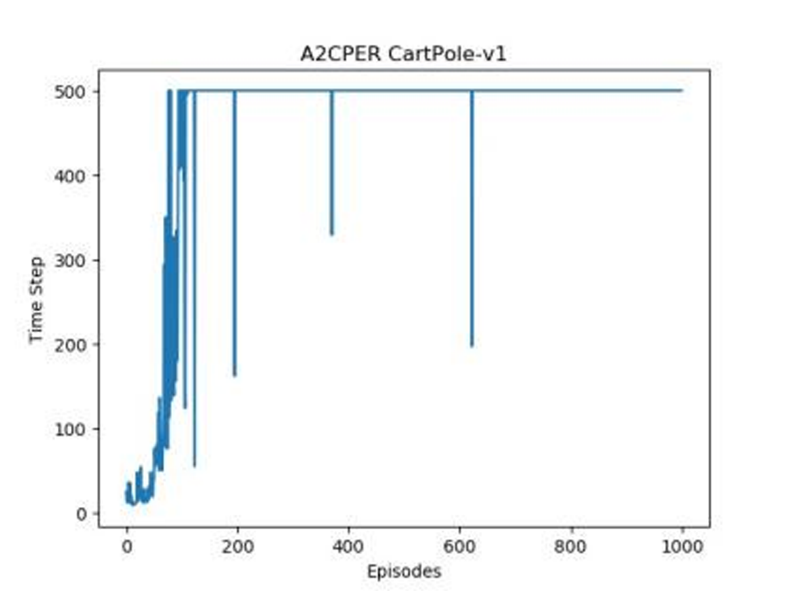


图5-4 A2CPER算法实验数据

Fig. 5-4 A2CPERAlgorithm Experimental Data

|  |  |  |
| --- | --- | --- |
|  |  | |
| a) DQN算法实验数据 | b) AC算法实验数据 |
| 图5-5 不同算法实验数据  Fig. 5-5 Different algorithm experimental data | |
|  |  | |
| a) ACER算法实验数据 | b) Policy算法实验数据 |
| 图5-6 不同算法实验数据  Fig. 5-6 Different algorithm experimental data | |

5-4至图5-6展示了Policy、DQN、AC、ACER、A2CPER这几种算法维持整车平衡的时间步长。图中标题代表实验所用的算法和实验环境；横坐标为实验的步长，单位为次；纵坐标为实验车每一步保持平衡的时间，单位为毫秒。

图5-7展示的是小车成功保持平衡的平均时长。值得注意的是，A2CPER算法的平均时长明显高于其他算法，体现出其引入的优化效果，而其他算法在这方面则存在不同程度的不足。在平均值下降的阶段，各算法都出现了明显的波动，值得注意的是，ACER的平均表现与DQN[47]算法基本持平，但仍略有差距。造成这种差异的主要原因可以归因于ACER无法有效管理目标网络内的参数优化延迟，从而使得参数更新过快，产生波动。

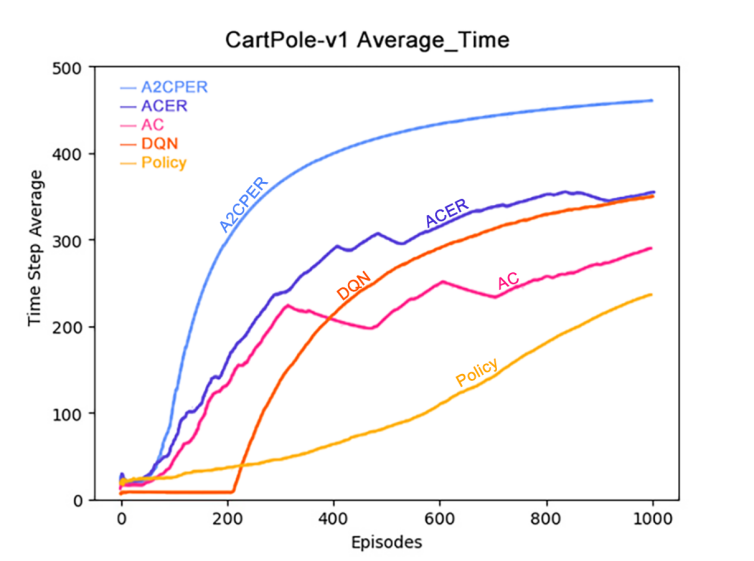


图5-7不同算法平均时间步数

Fig. 5-7 Average time steps of different algorithms

图5-7中数据表示：标题为实验环境和实验目标，横坐标为实验步数，纵坐标为小车保持平衡的平均时间，图左上角代表不同的算法，用不同的颜色表示。以色盲友好的方式绘制图片。

图5-8描绘了小车获得的平均奖励值。为了减轻个别训练场景偶尔波动的影响，降低了每次训练的奖励值的权重。这种调整确保了偶尔的波动对整体数据集的影响有限。

如图5-8所示，A2CPER算法在400次迭代左右达到平均奖励值峰值，并在较长时间内保持这一水平。这一观察结果强调了实验的优化在解决与本研究中提出的离散问题类似的离散问题方面的有效性。此外，曲线表现出显著的平滑度，表明对波动进行了有效控制。这些发现增强了A2CPER算法在解决具有挑战性的强化学习任务时保持一致和稳定的性能的能力。

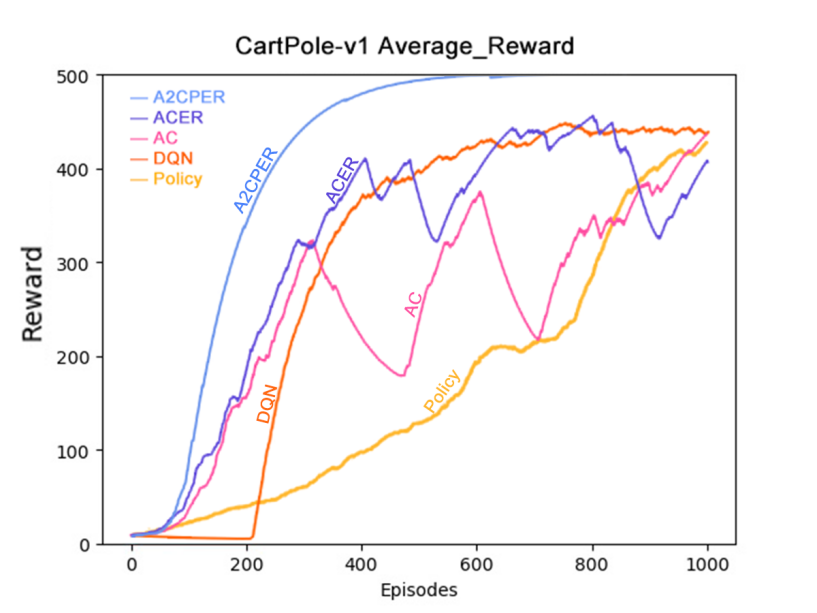


图5-8不同算法的平均奖励值

Fig. 5-8 The average reward value of different algorithms

图5-8中数据表示：标题为实验环境和实验目标，横坐标为实验的步数，纵坐标为小车保持平衡的平均奖励，图左上角代表不同的算法，用不同的颜色表示。以色盲友好的方式绘制图片。

（2）Acrobot-v1实验

在CartPole-v1环境中进行实验后，将调查扩展到Acrobot-v1环境。实验的核心方法与CartPole-v1相似，重点分析达到目标高度和获得奖励所需的时间，并根据计算出的平均值进行比较[48]。实验中使用的超参数在附录中有详细说明，这些超参数是根据大量测试选出的最适合的。

图5-9显示了五种算法，分别是A2CPER、ACER、AC、DQN和Policy算法使Acrobot代理达到目标高度所需的平均时间。从图中可以看出，除了 A2CPER算法外，所有其他算法在试验期间的平均时间都出现了不同程度的增加，表明学习出现了倒退。相比之下，A2CPER算法的平均时间始终呈下降趋势，没有任何倒退，这突显了其优于其他算法的一致性。这种一致性归因于引入了优先经验重放机制，该机制可防止训练环境中出现可能导致负面学习结果的重大差异。这种现象在ACER中尤为明显，在大约300次训练迭代中，该算法受到经验重放缓冲区中“脏数据”的严重影响，导致负面学习。

此外，与其他算法相比，A2CPER算法的平均时间始终较低，证明了其在解决这一离散问题方面的有效性。这种性能优势归功于自注意力机制的结合以及与目标网络的集成，这确保了参数更新既有针对性又循序渐进。

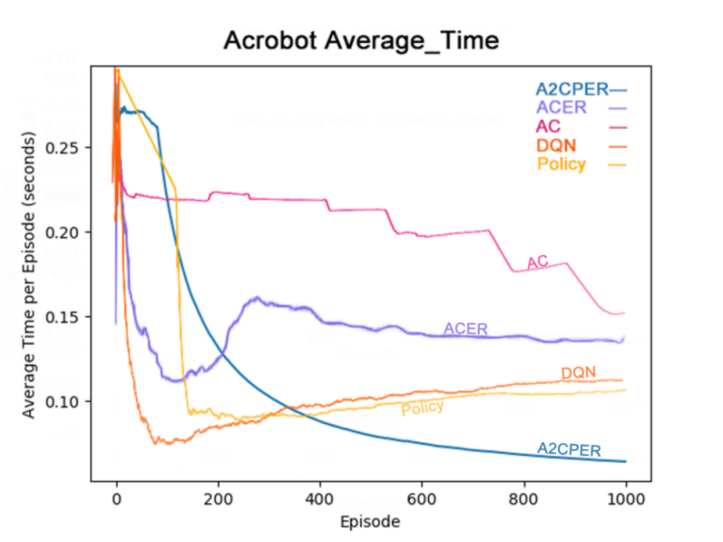


图5-9不同算法使Acrobot-v1代理达到目标高度所需的平均时间

Fig. 5-9 The average time required for different algorithms to reach the target height

图5-9提供了机械臂达到目标高度所需的平均持续时间数据，以秒为单位。x轴表示实验步骤数，而y轴表示平均时间。图表的右上角为颜色图例，用于区分不同的算法。

图5-10说明了在Acrobot-v1环境中由相同算法A2CPER、ACER、AC、DQN 和 Policy获得的平均奖励。该图表明A2CPER和ACER算法在奖励获取方面表现相似，但A2CPER始终优ACER。两者在效率上都大大超过了 AC 算法，凸显了实施经验重放机制的明显优势。与平均时间数据一样，其他四种算法的平均奖励值也表现出波动，这表明 A2CPER 算法中注意力机制和其他方法的集成有效地最大限度地减少了这种变化。

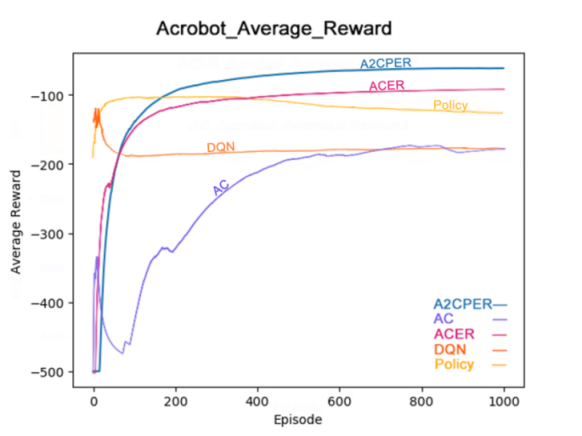


图5-10不同算法平均奖励获取情况

Fig. 5-10 Average reward acquisition of different algorithms

图5-10中显示的数据表示机械臂达到目标高度时获得的平均奖励值。x轴表示实验步骤数（以迭代为单位），y轴表示平均奖励值。图表的右下角使用不同的颜色区分算法，并且图表设计为色盲友好型，

（3）MountainCar-v0实验

最后，在MountainCar-v0环境中进行了测试。由于环境局限性，无法有效观察奖励值，所以将实验重点放在汽车到达山顶所需的时间上，比较不同算法的平均值[49]。每一种算法都经过 400 次训练，以确定哪种算法表现最佳。

下图5-11显示了Policy、DQN、AC、ACER和A2CPER算法将汽车带到目标点所需的平均时间。A2CPER和ACER算法表现出整体稳定性，没有明显的波动。这表明经验重放机制显著增强了算法的稳定性。相比之下，其他三种算法经历了大幅波动，这表明发生了过度提取“脏数据”导致这些变化。A2CPER 最终的平均时间接近0毫秒，证明了自注意力机制与多种方法的协同作用，最终取得了优异的结果。

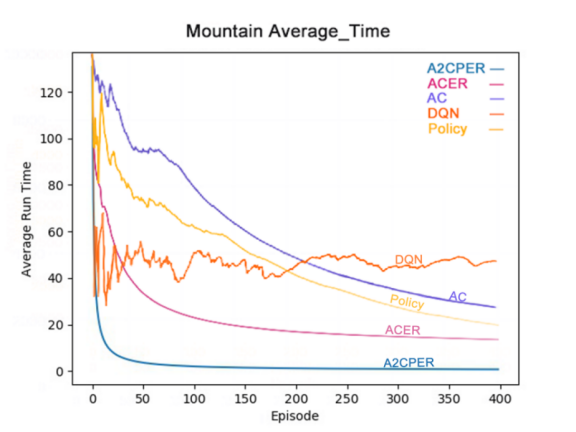


图5-11不同算法在MountainCar-v0环境中到达目标位置的平均时间。

Fig. 5-11 The average time for different algorithms to reach the target location.

图5-11所显示的数据表明了实验环境和目标，其中x轴表示实验步骤数（以迭代为单位），y轴表示汽车到达目标所需的时间（以毫秒为单位）。不同的算法由图表右上角的不同颜色表示。

### 实验总结

（1）Cartpole-v1的实验结果

A2CPER算法的性能通过小车平衡的持续时间和累积奖励来衡量，与其他算法相比，它强调了它在这种离散设置中的优化优势。它通过改进数据采样过程增强了AC算法，纠正了拥有经验重放的ACER算法中非优先级数据收集，缓解了与 DQN相关的延长数据采样持续时间，并加快了通常限制策略梯度方法的收敛速度。

在处理 CartPole-v1环境时，A2CPER算法的时间复杂度为，其中表示输入数据的维数，表示神经网络中的参数数量。算法的运行时间随着训练迭代而逐渐减少，比如不同算法的平衡持续时间，其他算法一般需要300次迭代才能收敛，而A2CPER算法只需要100次迭代就能收敛。

具体到CartPole-v1环境中，A2CPER算法的实现显著提升了小车的性能。这种提升首先源于自注意力机制的引入，该机制在小车制定策略时将注意力集中在保持平衡的关键目标上。这种机制有效地防止了在学习过程中加入过多和不相关的因素。此外，优先级经验回放技术提供了一个回放空间，小车只需要根据每次训练的优先级权重从缓冲区中检索经验，过滤掉大量无意义的数据。这种方法还包括使用目标网络来调节参数更新和梯度裁剪，以防止梯度爆炸。通过多次训练，模型性能逐渐提高，最终学习到最大化累积奖励的最佳策略。

（2）Acrobot-v1的实验结果

平均时间和平均奖励值的比较分析表明，A2CPER算法在这种离散动作控制环境中具有出色的优化性能。它不仅解决了稳定性问题，防止了负面学习，而且还改善了 AC 算法中观察到的数据收集效率低下以及ACER经验重放缓冲区中“脏数据”的问题。此外，它还增强了DQN和Policy算法中缺乏的稳定性。

在解决Acrobot-v1环境时，A2CPER算法的时间复杂度表示为，其中是输入数据的维数，是神经网络中的参数数量。算法的运行时间随着每次训练迭代而减少，平衡持续时间图中提供的性能数据显示时间超过500毫秒。图5-9中还描绘了收敛速度，显示A2CPER在大约200次迭代内实现有效收敛。

在Acrobot-v1环境下，A2CPER算法的实现大大提升了机械臂的应对挑战能力。首先，自注意力机制的引入使得机械臂能够快速找到最有效的运动策略以达到目标高度；实验中采用的优先经验回放技术使得机械臂能够根据动作的加权重要性进行选择，并根据结果进行调整，有效过滤掉不相关的数据；此外，目标网络的使用缓和了参数的更新，并使用梯度裁剪来防止梯度爆炸。经过多次训练，模型的性能逐渐提升，最终学会了最大化累积奖励的最优策略。

（3）MountainCar-v0 的实验结果

平均时间指标的比较分析表明，与其他算法相比，A2CPER算法在这种离散环境中表现出明显的优化优势。其最显著的影响是算法稳定性的增强；即使在计算通常不会出现较大波动的平均值时，其他算法也表现出明显的变化性，而A2CPER保持了接近于零的整体波动性并表现出色，有效解决了汽车的上坡挑战。它改善了AC算法的收敛速度慢的问题，解决了ACER算法的平均时间结果不佳的问题，并稳定了DQN和Policy算法表现出的异常高波动。

在管理MountainCar-v0环境中，A2CPER算法的时间复杂度表示为，其中表示输入数据的维数，表示神经网络中的参数数量。随着训练迭代的继续，算法的运行时间减少，到最后达到约2毫秒，A2CPER在大约20次迭代内实现了稳健收敛。

对于MountainCar-v0环境，A2CPER算法为汽车上坡导航提供了非常有效的解决方案[50]。最初，在训练过程中引入自注意力机制使系统能够快速辨别出到达顶峰的最有效运动策略。实验期间采用的优先经验重放技术使机械臂能够根据权重选择动作并根据结果进行调整，从而过滤掉大量不相关的数据。这包括使用目标网络来减慢参数更新速度和梯度剪裁以防止梯度爆炸。

## MACPER算法处理离散动作动态性控制决策实验与分析

### 实验环境

本实验基于MiniWorld仿真环境，选取迷宫任务作为测试平台，用于验证强化学习算法在复杂环境中的路径规划能力[51]。迷宫任务涉及智能体在未知环境中进行探索和导航，目标是找到最优路径从起点到达目标点。

（1）迷宫结构

迷宫由多个相互连接的房间、通道和障碍物组成，限制了智能体的可行路径[52]。不同迷宫实例的复杂度不同，主要体现在迷宫通道可能较为笔直或包含大量弯折、死胡同。随机生成障碍物，阻挡部分路径，提高探索难度。根据算法自动生成不同的迷宫环境，用于对不同算法进行训练和测试。迷宫结构大体的拓扑关系是类似的，不会出现某一个算法的迷宫过于负责或者过于简单的情况发生。迷宫示意图见图5-12。

|  |  |  |
| --- | --- | --- |
| 迷宫示意 | 迷宫走向示意 | |
| a) 迷宫示意图 | b) 迷宫正确路线示意图 |
| 图5-12迷宫示意图和迷宫正确路线示意图  Fig. 5-12 Maze diagram and maze correct route diagram | |

（2）状态空间

表5-12 迷宫问题的状态空间

Table 5-12 The state space of maze problems

|  |  |
| --- | --- |
| 状态空间 | 设定值 |
| RGB 图像 | 范围为[84,84,3] |
| 位置信息(x, y)坐标 | 智能体在迷宫中的绝对位置 |
| 朝向信息 | 范围为[0°, 90°,180°, 270°] |
| 部分可观测环境 | 无法直接获取全局地图 |

因此，状态空间的维度为4维，每个维度描述了智能体的不同状态

（3）动作空间

表5-13 迷宫问题的动作空间

Table 5-13 The action space of maze problem

|  |  |
| --- | --- |
| 动作空间 | 动作 |
| 0 | 前进：向前移动固定步长 |
| 1 | 后退：向后移动固定步长 |
| 2 | 左转：左转 90° |
| 3 | 右转：右转 90° |

因此，动作空间的维度为4维，智能体可以有四种选择进行移动，但是后退动作一旦产生，就说明智能体出现了错误判断，所以和接下来的奖励机制进行联系。

（4）奖励机制

表5-14 迷宫问题的奖励机制

Table 5-14 Reward mechanism for maze problems

|  |  |
| --- | --- |
| 奖励 | 奖励值 |
| 成功到达目标点 | +10 分 |
| 每步前进奖励 | +0.01 分 |
| 碰壁惩罚 | -1 分 |
| 时间步长惩罚 | -0.01 分 |
| 自适应奖励 | 智能体自行给出 |

奖励机制中的自适应奖励，则是智能体自行计算后得出的结果，也是本算法的优势性所在。

（5）训练与测试设定

训练阶段是在100个随机生成的迷宫中进行训练，迷宫拓扑结构在训练过程中不断变化，保证算法的泛化能力。测试阶段则是在未见过的30个迷宫上进行测试，评估算法的泛化能力。

### 实验代码设置

（1）环境初始化

与前面讨论的环境初始化相同，保持MiniWorld环境的设计，并引入ACER算法中所需的一些特殊处理。在初始化环境时需要确保MiniWorld设置与前述相同，设置网格大小、目标巡回任务和迷宫任务，以及自适应奖励机制；每个状态通常包含代理的当前位置，可能扩展为包括目标位置、历史状态等动作空间：使用离散动作空间（上、下、左、右），并根据任务的需求进行相应的扩展。

（2）模型设置

MACPER的模型设置大概包括：策略网络、价值函数网络、网络架构、输入输出、奖励模型等，详细的模型设计第四章有详细介绍。

（3）优化器设置与超参数设计

MACPER算法中使用了基于梯度的优化方法来更新演员和评论家的网络权重。核心的优化目标是最大化优势函数，即通过最大化当前策略相较于状态的价值函数的优势来提高策略。

表5-15迷宫问题的优化器设置与超参数设计

Table 5-15 Optimizer settings and hyperparameter design for maze problems

|  |  |
| --- | --- |
| 超参数名称 | 设定值 |
| 学习率 | learn = 0.01 |
| 折扣因子 | gamma = 0.99 |
| 批量大小 | Size = 64 |
| 经验回放池大小 | replaysize = 10000 |
| 目标网络更新频率 | update\_target = 1000 |
| 折扣因子 | gamma = 0.99 |
| 最终探索率 | epsilon\_final = 0.01 |
| 探索衰减率 | epsilon\_decay = 0.995 |
| 优化器 | Adam |
| 优化器学习率 | adam = 0.001 |

### 实验结果以及分析

（1）奖励获得情况对比

首先是训练阶段的奖励如图5-13所示，进行20次左右时每个算法都处于学习阶段，20次之后，MACPER算法的奖励获取情况明显高于其他算法情况，由此表示，MACPER算法对于新数据的训练情况有明显优势，要得益于元学习的多任务适应模式，其他算法都是一张地图进行学习训练，MACPER算法则是五张迷宫地图进行同时学习，所以学习效率要明显提升。

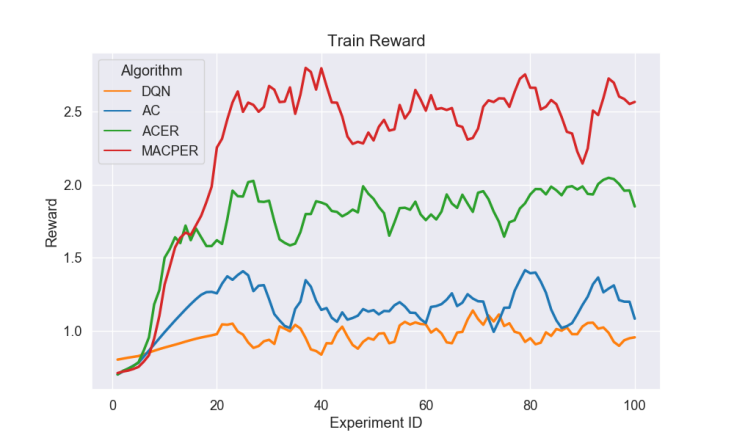


图5-13不同算法训练阶段奖励获取情况

Fig. 5-13 Rewards for different algorithm training stages

其次是测试数据奖励获取情况，由图5-14可以看出，所有算法在面对测试数据的时候，几乎没有波动或者波动情况很少，所以证明所有算法都对训练的迷宫有了一定的经验累计。整体上看，MACPER算法相对于其他算法，奖励值的平均获取情况有了明显的提升。这得益于自奖励机制的引入，通过自奖励机制的引入，MACPER算法针对于新的数据的奖励设立机制就更趋向于正确，使得智能体也可以顺着最高奖励值的方向前进。



图5-14不同算法测试阶段奖励获取情况

Fig. 5-14 Rewards for different algorithm testing stages

（2）适应奖励情况

适应奖励一定程度上反应了算法面对新环境的适应能力，由图5-15可以看出整体上看DQN算法面对新迷宫的适应能力要远远小于MACPER算法，这归功于优先级经验回放机制的引入，正因为有了经验回放机制，MACPER算法在面对新环境的时候，先将经验池中的环境数据取出进行比对，寻找最合适的数据进行类比，寻找之前训练的经验，可以大大减少新环境的训练时间，快速达到最佳效率。

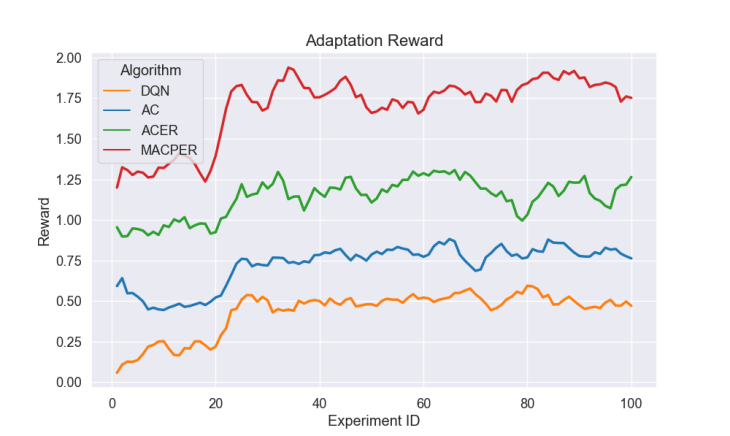


图5-15不同算法训练阶段适应奖励获取情况

Fig. 5-15 Adaptation rewards for different algorithm training stages

测试适应奖励获取情况和之前的测试奖励获取情况类似如图5-16所示，甚至整体奖励值都大于1.8，这说明元学习机制和自奖励机制在面对复杂的离散性问题时，可以很好的协助，达到快速适应环境的效果。

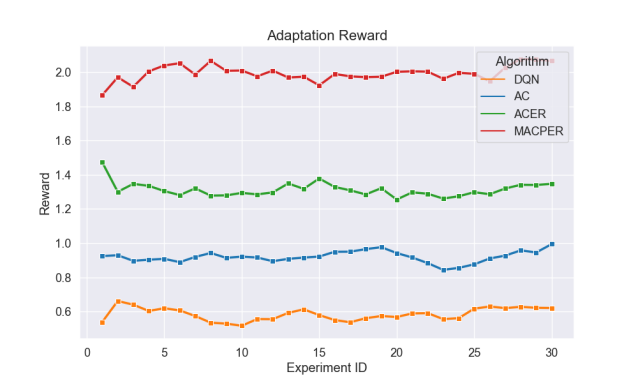


图5-16不同算法测试阶段适应奖励获取情况

Fig. 5-16 Adaptation rewards for different algorithm testing stages

（3）成功率对比

这两个情况可以一起分析，无论是训练情况下还是测试情况下，MACPER算法的成功率要明显优于其他算法，这说明在保证了奖励的获取、收敛速度快和动作选择正确的前提下，还能保证正确率。如图5-17所示。DQN算法受限于对状态空间的线性建模和较为简单的奖励机制，在较复杂的迷宫任务中，探索效率较低，导致成功率偏低。AC算法和ACER算法通过基于策略梯度的优化方法，较DQN在一些情况下表现更好，但在面对环境的变化时，成功率依然受到一定限制。MACPER算法则通过引入元学习和自适应奖励机制，增强了智能体在复杂环境中的探索能力和适应性，极大地提高了成功率。

|  |  |  |
| --- | --- | --- |
|  |  | |
| a) 不同算法训练正确率情况 | b) 不同算法测试正确率情况 |
| 图5-17不同算法的正确率情况  Fig. 5-17 Accuracy of different algorithms | |

### 实验总结

在本次实验中，基于DQN、AC、ACER和MACPER算法对MiniWorld迷宫任务进行的对比测试中，MACPER算法表现出了显著的优势。以下是MACPER算法的主要优势和实验结果的总结：

（1）奖励获取情况

在训练阶段，MACPER算法的奖励获取情况明显优于其他算法。在前20次训练中，所有算法都处于学习阶段，但在20次训练之后，MACPER算法的奖励获取速度明显加快。MACPER算法的优势源于其元学习的多任务适应模式，可以在训练过程中并行学习多个迷宫环境，而不是单一地图逐一训练。这使得MACPER能够更好地理解和泛化环境信息，提高了学习效率，达到了更高的奖励获取水平。

（2）适应奖励能力

适应奖励反映了算法在新环境中的适应能力，MACPER算法在这方面的表现明显优于DQN等其他算法。尤其是在面对新的迷宫环境时，MACPER通过经验回放机制有效减少了新环境训练的时间。当智能体遇到新环境时，它会从经验池中提取最相关的环境数据进行比对和学习，从而找到最合适的训练数据进行类比，快速适应新环境并达到最佳效果。这一机制大大提高了算法的适应速度，使其能够在更短的时间内完成有效的训练。

（3）成功率对比

在测试阶段，MACPER算法的成功率显著高于其他算法，达到了90%以上，远高于DQN、AC和ACER算法的50%左右。成功率的提高表明，MACPER不仅在奖励获取和适应性上表现优异，还在决策准确性和环境适应能力上提供了显著的优势。MACPER算法能够在复杂的迷宫任务中快速进行有效探索，做出正确的决策，保证智能体成功到达目标点。

（4）元学习和优先级经验回放机制的结合

MACPER算法的关键优势在于其结合了元学习和优先级经验回放机制。这使得算法能够通过多任务学习和经验回放有效减少在新环境中的学习时间。传统的算法如DQN和AC系列算法通常是逐个环境进行训练，这种方式在面对环境变化时效率较低。而MACPER通过并行学习多个环境，能够更好地将经验迁移到新的环境中，从而提升学习效率并加快收敛速度。

MACPER算法通过元学习和自适应奖励机制的引入，在MiniWorld迷宫任务中展现出了较强的优势。其在奖励获取、适应奖励、成功率和学习效率等方面均优于传统的DQN、AC和ACER算法。尤其是在面对新的环境时，MACPER能够通过经验回放和多任务适应的方式，快速适应并有效学习，从而大大提高了智能体的探索能力和决策准确性。实验结果证明，MACPER算法能够高效解决复杂的离散任务，具有更强的泛化能力和应用前景，尤其适用于多任务学习和复杂环境中的强化学习问题。

## 本章小结

本章通过两个智能体离散动作实验，分别验证了A2CPER算法在面对单离散动作任务和MACPER算法在面对多离散动作任务的优势，并由各种实验数据得出了两个算法的优势结论。

首先，A2CPER算法在单离散动作任务中的表现优于传统的强化学习算法，特别是在面对复杂任务时，能够有效地通过优先级经验回放和自注意力机制加速学习过程。通过引入自注意力机制，A2CPER能够在训练初期就快速收敛，并且在测试阶段展示了较高的成功率和奖励获取水平，体现了其在单一离散动作环境中的高效性。

其次，MACPER算法在多离散动作任务中展现了更加显著的优势。由于多任务学习的引入，MACPER能够在复杂环境中同时进行多个迷宫的训练，极大地提升了训练效率和探索能力。在面对多样化和复杂的任务时，MACPER能够通过并行训练和元学习机制有效适应环境，并且通过经验回放减少了学习时间，显著提高了任务的成功率和奖励获取水平。尤其是在新环境的适应性上，MACPER的表现更为突出，能够快速调整策略，减少训练所需时间。

# 结 论

本论文围绕智能体离散动作控制决策问题，针对传统强化学习算法在复杂离散动作任务中的不足，提出并研究了基于改进ACER算法的两种新型算法：A2CPER算法与MACPER算法。通过对这两种算法的创新性设计与改进，本研究有效提升了智能体在面对多种复杂环境下的学习效率和适应能力，并通过大量实验验证了算法的优越性。具体的创新性成果及贡献如下：

（1）提出了基于自注意力机制和优先级经验回放技术的A2CPER算法

本研究首先提出了基于自注意力机制的A2CPER算法。通过引入自注意力机制，A2CPER能够自动筛选重要信息，减少冗余数据的干扰，显著提升了智能体在面对复杂离散动作任务时的学习速度与决策准确性。同时，A2CPER算法结合了优先级经验回放技术，在训练过程中优先选择有效经验进行学习，避免了低效数据对训练结果的影响，进一步提升了学习的效率和稳定性。

（2）优化了损失值计算方法以适应复杂任务

在A2CPER算法的设计中，本研究对传统的损失值计算方法进行了创新性的优化。通过根据不同任务环境的特点，动态调整损失计算方式，使得算法能够在多任务环境下保持较高的适应能力。此优化不仅增强了算法的稳定性，还有效提升了训练过程中的收敛速度，特别是在面对变化较大的任务时，能够自适应地进行调整，确保了智能体始终能够获得更优的学习效果。

（3）提出了基于元学习与自适应奖励机制的MACPER算法

本研究提出的MACPER算法，通过引入元学习机制，使得智能体能够在面对不同的离散动作任务时，迅速适应新的环境，提高了训练的效率。MACPER算法通过并行学习多张迷宫地图，减少了单任务学习过程中的时间消耗，显著提高了学习过程的效率。结合自适应奖励机制，MACPER能够根据任务的动态变化自动调整奖励策略，确保智能体在不同的任务中都能够有效地获取奖励并不断优化其决策过程，从而提升了算法在复杂环境中的稳定性与适应性。

（4）创新性地研究了智能体控制离散动作的决策问题

本研究创新性地探讨了智能体控制离散动作的决策问题，并通过提出A2CPER与MACPER两种算法，解决了传统强化学习算法在处理离散动作控制决策时面临的挑战。通过对动作空间离散性与环境复杂性的深入研究，结合自适应奖励机制、优先级经验回放和元学习等创新方法，提升了智能体在复杂任务中的学习能力与适应能力，推动了离散动作控制领域的研究进展。

综上所述，本研究通过提出并验证A2CPER与MACPER两种改进的ACER算法，不仅显著提升了智能体在复杂离散动作任务中的表现，还为智能体离散动作控制决策问题的研究提供了新的思路和方法。这些创新成果不仅具有重要的理论意义，也为智能体在实际应用中的推广与发展提供了坚实的基础。未来，随着技术的进一步发展，基于元学习与自适应奖励机制的算法有望在更广泛的领域中得到应用，并取得更为显著的成效。

尽管本研究在改进ACER算法并提出A2CPER与MACPER算法方面取得了显著的成果，但仍存在一些不足之处。首先，虽然引入了元学习和自适应奖励机制，提升了算法在多任务环境中的适应性，但在某些极端或高度动态的环境中，智能体的学习效率和决策稳定性可能仍然受到一定的挑战，特别是在面临任务变化频繁或环境极度复杂的情况下。其次，尽管A2CPER和MACPER算法在实验中表现出较好的性能，但其在大规模任务或高维状态空间下的表现仍需要进一步评估。最后，尽管本研究对损失值计算方法进行了优化，但在某些特定情境下，如何进一步提高损失计算的精确性，仍是未来研究的一个重要方向。未来的工作可以在这些不足的基础上，进一步完善算法的稳定性和适应能力，探索更加高效的策略和优化方法，以提高算法在实际应用中的表现。

参考文献

1. 高阳, 陈世福, 陆鑫. 强化学习研究综述[J]. 自动化学报, 2004, 30(001): 86-100.
2. 刘全, 翟建伟, 章宗长, 等. 深度强化学习综述[J]. 计算机学报, 2018, 41(1): 1-27.
3. Shen Y C. 使用自適應性離散動作空間的基於模型強化學習[D]. 中国台湾： 國立臺灣大學, 2024: 1-36.
4. Mousavi S S, Schukat M, Howley E. Deep reinforcement learning: an overview[C] Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016: Volume 2. Springer International Publishing, 2018: 426-440.
5. 陈佳盼, 郑敏华. 基于深度强化学习的机器人操作行为研究综述[J]. 机器人, 2022, 44(2): 236-256.
6. Watkins C J C H, Dayan P. Q-learning[J]. Machine learning, 1992, 8: 279-292.
7. Osband I, Blundell C, Pritzel A, et al. Deep exploration via bootstrapped DQN[J]. Advances in neural information processing systems, 2016, 29.
8. Babaeizadeh M, Frosio I, Tyree S, et al. Reinforcement learning through asynchronous advantage actor-critic on a gpu[J]. arXiv preprint arXiv:1611.06256, 2016.
9. Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms[J]. arXiv preprint arXiv:1707.06347, 2017.
10. Hasselt H. Double Q-learning[J]. Advances in neural information processing systems, 2010, 23.
11. He N, Yang S, Li F, et al. A-DDPG: Attention mechanism-based deep reinforcement learning for NFV[C] 2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS). IEEE, 2021: 1-10.
12. Wang Z, Bapst V, Heess N, et al. Sample efficient actor-critic with experience replay[J]. arXiv preprint arXiv:1611.01224, 2016.
13. 刘庆强, 刘鹏云. 基于优先级经验回放的 SAC 强化学习算法[J]. 吉林大学学报 (信息科学版), 2021, 39(2): 192-199.
14. 周瑶瑶, 李烨. 基于排序优先经验回放的竞争深度 Q 网络学习[J]. Application Research of Computers/Jisuanji Yingyong Yanjiu, 2020, 37(2).
15. 郑敬华, 郭世泽, 高梁, 等. 基于多任务学习的大五人格预测[J]. 中国科学院大学学报, 2018, 35(4): 550.
16. Zhu X, Zuo J, Ren H. A modified deep neural network enables identification of foliage under complex background[J]. Connection Science, 2020, 32(1): 1-15.
17. Raudys A, Šubonienė A. A review of self-balancing robot reinforcement learning algorithms[C] Information and Software Technologies: 26th International Conference, ICIST 2020, Kaunas, Lithuania, October 15–17, 2020, Proceedings 26. Springer International Publishing, 2020: 159-170.
18. 郑思远, 崔苗, 张广驰. 基于强化学习的无人机安全通信轨迹在线优化策略[J]. 广东工业大学学报, 2021, 38(4): 59-64.
19. 赵冬梅, 陶然, 马泰屹, 等. 基于多智能体深度确定策略梯度算法的有功-无功协调调度模型[J]. 电工技术学报, 2021, 36(9): 1914-1925.
20. Kostrikov I, Nair A, Levine S. Offline reinforcement learning with implicit q-learning[J]. arXiv preprint arXiv:2110.06169, 2021.
21. 李凯文, 张涛, 王锐, 等. 基于深度强化学习的组合优化研究进展[J]. 自动化学报, 2021, 47(11): 2521-2537.
22. 杨思明, 单征, 丁煜, 等. 深度强化学习研究综述[J]. 计算机工程, 2021, 47(12): 19-29.
23. 董豪, 杨静, 李少波, 等. 基于深度强化学习的机器人运动控制研究进展[J]. 控制与决策, 2022, 37(2): 278-292.
24. 张荣霞, 武长旭, 孙同超, 等. 深度强化学习及在路径规划中的研究进展[J]. Journal of Computer Engineering & Applications, 2021, 57(19).
25. 陈佳盼, 郑敏华. 基于深度强化学习的机器人操作行为研究综述[J]. 机器人, 2022, 44(2): 236-256.
26. 刘克. 实用马尔可夫决策过程[M]. 清华大学出版社有限公司, 2004: 67-90
27. Bellman R. Dynamic programming[J]. science, 1966, 153(3731): 34-37.
28. 夏宗涛, 秦进. 一种深度 Q 网络的改进算法[J]. Application Research of Computers/Jisuanji Yingyong Yanjiu, 2019, 36(12).
29. 夏宗涛, 秦进. 基于优势学习的深度 Q 网络[J]. 计算机工程与应用, 2019, 55(20): 101-106.
30. 黄岩松, 姚锡凡, 景轩, 等. 基于深度 Q 网络的多起点多终点 AGV 路径规划[J]. 计算机集成制造系统, 2023, 29(8): 2550.
31. Romero A, Song Y, Scaramuzza D. Actor-critic model predictive control[C] 2024 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2024: 14777-14784.
32. Zhou R, Liu T, Cheng M, et al. Natural actor-critic for robust reinforcement learning with function approximation[J]. Advances in neural information processing systems, 2024, 36.
33. Huang X, Cheng Y, Yu Q, et al. Deep Reinforcement Learning for Autonomous Driving based on Safety Experience Replay[J]. IEEE Transactions on Cognitive and Developmental Systems, 2024.
34. 娄哲, 王燕, 山珊等. 离策略强化学习中的平衡优先经验回放[J]. 神经计算与应用, 2024: 1-17.
35. Horváth D, Martín J B, Erdos F G, et al. HiER: Highlight Experience Replay for Boosting Off-Policy Reinforcement Learning Agents[J]. IEEE Access, 2024.
36. Lipeng L, Xu L, Liu J, et al. Prioritized experience replay-based ddqn for unmanned vehicle path planning[J]. arXiv preprint arXiv:2406.17286, 2024.
37. 胡伟, 周颖, 何红伟. 基于噪声N步双深度Q网络和优先经验重放的移动机器人导航[J]. 电子技术, 2024, 13(12): 2423。
38. Lu C, Ball P, Teh Y W, et al. Synthetic experience replay[J]. Advances in Neural Information Processing Systems, 2024, 36.
39. Vettoruzzo A, Bouguelia M R, Vanschoren J, et al. Advances and challenges in meta-learning: A technical review[J]. IEEE transactions on pattern analysis and machine intelligence, 2024, 46(7): 4763-4779.
40. 王若男, 董琦. 基于学习机制的多智能体强化学习综述[J]. 工程科学学报, 2024, 46(7): 1251-1268.
41. 冯斌, 刘哲, 黄娜等. 基于成对元学习的生物活性基础模型[J]. 自然机器智能, 2024, 6(8): 962-974.
42. Hospedales T, Antoniou A, Micaelli P, et al. Meta-learning in neural networks: A survey[J]. IEEE transactions on pattern analysis and machine intelligence, 2021, 44(9): 5149-5169.
43. 夏庆锋, 许可儿, 李明阳, 等. 强化学习中的注意力机制研究综述[J]. Journal of Frontiers of Computer Science & Technology, 2024, 18(6).
44. Eberding L M. Comparison of machine learners on an aba experiment format of the cart-pole task[C] International Workshop on Self-Supervised Learning. PMLR, 2022: 49-63.
45. Peters J, Vijayakumar S, Schaal S. Natural actor-critic[C] Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16. Springer Berlin Heidelberg, 2005: 280-291.
46. Pan X, Ge C, Lu R, et al. On the integration of self-attention and convolution[C] Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022: 815-825.
47. 马浩东, 陈玲玲, 金小杭. 基于改进 DQN 算法的机器人路径规划[J]. 电脑与电信, 2024, 1(11): 37-41.
48. Gillen S, Molnar M, Byl K. Combining deep reinforcement learning and local control for the acrobot swing-up and balance task[C] 2020 59th IEEE Conference on Decision and Control (CDC). IEEE, 2020: 4129-4134.
49. Winberg A, Öhrstam Lindström O. Reinforcement Learning Methods for OpenAI Environments[J]. 2020.
50. Xu Z, Cao L, Chen X. Deep reinforcement learning with adaptive update target combination[J]. The Computer Journal, 2020, 63(7): 995-1003.
51. Chevalier-Boisvert M, Dai B, Towers M, et al. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks[J]. Advances in Neural Information Processing Systems, 2023, 36: 73383-73394.
52. 徐守江. 迷宫算法综述[J]. 信息与电脑: 理论版, 2009 (10): 91-92.

攻读硕士学位期间发表的学术论文及获得成果

致 谢