计算机科学
COMPUTER SCIENCE

基于轨迹信息量的分层强化学习方法

徐亚鹏, 刘全, 栗军伟

引用本文

徐亚鹏, 刘全, 栗军伟. 基于轨迹信息量的分层强化学习方法[J]. 计算机科学, 2023, 50(12): 314-321.

XU Yapeng, LIU Quan, LI Junwei. Hierarchical Reinforcement Learning Method Based on Trajectory Information [J]. Computer Science, 2023, 50(12): 314-321.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.221100096

基于轨迹信息量的分层强化学习方法

徐亚鹏[1]　刘　全[1,2]　栗军伟[1]
1 苏州大学计算机科学与技术学院　江苏 苏州 215006
2 苏州大学江苏省计算机信息处理技术重点实验室　江苏 苏州 215006
(20205227104@stu.suda.edu.cn)

摘　要　基于option的分层强化学习(The Option-Based Hierarchical Reinforcement Learning,O-HRL)算法具有时序抽象的特点,可以有效处理强化学习中难以解决的长时序、稀疏奖励等复杂问题。目前O-HRL方法的研究主要集中在数据效率提升方面,通过提高智能体的采样效率以及探索能力,来最大化其获得优秀经验的概率。然而,在策略稳定性方面,由于在上层策略引导下层动作的过程中仅仅考虑了状态信息,造成了option信息的利用不充分,进而导致下层策略的不稳定。针对这一问题,提出了一种基于轨迹信息量的分层强化学习(Hierarchical Reinforcement Learning Method Based on Trajectory Information,THRL)方法。该方法利用option轨迹的不同类型信息指导下层动作选择,通过得到的扩展轨迹信息生成推断option。同时引入鉴别器将推断option与原始option作为输入,以获得内部奖励,使得下层动作的选择更符合当前option策略,从而解决下层策略不稳定的问题。将THRL算法以及目前优秀的深度强化学习算法应用于MuJoCo环境问题中,实验结果表明,THRL算法具有更好的稳定性以及性能表现,验证了算法的有效性。

关键词:option;分层强化学习;轨迹信息;鉴别器;深度强化学习
中图法分类号　TP181

Hierarchical Reinforcement Learning Method Based on Trajectory Information

XU Yapeng[1],LIU Quan[1,2] and LI Junwei[1]
1 School of Computer and Technology,Soochow University,Suzhou,Jiangsu 215006,China
2 Provincial Key Laboratory for Computer Information Processing Technology,Soochow University,Suzhou,Jiangsu 215006,China

Abstract　The option-based hierarchical reinforcement learning(O-HRL) algorithm has the characteristics of temporal abstraction,which can effectively deal with complex problems such as long-term temporal order and sparse rewards that are difficult to solve in reinforcement learning. The existing studies of O-HRL methods mainly focus on data efficiency improvement by increasing the sampling efficiency as well as the exploration ability of the agent to maximize its probability of obtaining excellent experiences. However,in terms of policy stability,the high-level policy guides the low-level action by only considering the state,resulting in the underutilization of option information,which leads to the instability of the low-level policy. To address this problem,a hierarchical reinforcement learning method based on trajectory information(THRL) is proposed. THRL uses different types of information of option trajectories to guide the selection of low-level actions,and also generates inferred options by the obtained extended trajectory information. A discriminator is introduced to use the inferred options and the original options as inputs to obtain internal rewards,which makes the selection of low-level actions more consistent with the current option policy,thus solving the instability problem of low-level policies. The effectiveness of THRL is verified by applying it to the MuJoCo environment,along with the best deep reinforcement learning algorithms,and experimental results show that the THRL algorithm has better stability and performance.

Keywords　Option,Hierarchical reinforcement learning,Trajectory information,Discriminator,Deep reinforcement learning

1　引言

强化学习[1](Reinforcement Learning,RL)作为机器学习领域的一个重要分支,它以马尔可夫决策过程(Markov Decision Process,MDP)为理论基础,解决智能体(Agent)在连续时间上的序贯决策问题。近年来,随着深度学习(Deep

到稿日期:2022-11-10　返修日期:2023-03-28
基金项目:国家自然科学基金(61772355,61702055,61876217,62176175);江苏高校优势学科建设工程资助项目
This work was supported by the National Natural Science Foundation of China(61772355,61702055,61876217,62176175) and Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions(PAPD).
通信作者:刘全(quanliu@suda.edu.cn)

Learning，DL)[2]的快速发展，将深度学习感知与强化学习决策相结合的深度强化学习(Deep Reinforcement Learning，DRL)[3]迎来了极大突破，例如围棋 AI 程序 AlphaGo[4]、自动驾驶[5]及医学领域新药设计[6]等。深度强化学习利用神经网络可以有效提取连续高维状态信息，解决了传统强化学习在高维状态空间中无法广泛应用的问题。Liu 等[7]通过对 DRL 算法的研究分析，总结了深度 Q 网络(Deep Q-Network，DQN)[3]、深度确定性策略梯度(Deep Deterministic Policy Gradient，DDPG)[8]、异步优势行动者-评论家(Asynchronous Advantage Actor-Critic，A3C)[9]等经典算法，并介绍了分层强化学习(Hierarchical Reinforcement Learning，HRL)[10]、多智能体强化学习(Multi-Agent Reinforcement Learning，MARL)[11]等研究热点。在后续的深度强化学习研究进展中，Schulman 等[12]提出了近端策略优化算法(Proximal Policy Optimization，PPO)，通过对训练过程中的新旧策略的比值进行剪枝来限制网络更新步长，从而使策略更新效果更加稳定。Haarnoja 等[13]提出了软行动者-评论家算法(Soft Actor-Critic，SAC)，通过将最大熵目标引入行动者-评论家框架，采用随机策略增大智能体的探索空间，来获得最优策略。

深度强化学习目前可以较好地被运用于解决高维连续空间问题，但在一些复杂任务中，如稀疏奖励环境、顺序决策问题等，无法取得理想的效果。分层强化学习作为强化学习领域一个重要分支，通过分层抽象技术将任务过程划分为双层结构，下层智能体在上层策略引导下选择动作，有效解决了深度强化学习无法处理复杂控制任务的问题。基于 option 的分层强化学习是分层强化学习的一个研究方向，主要以半马尔可夫决策过程(Semi-Markov Decision Process，SMDP)为理论基础。Sutton 等[14]在马尔可夫决策模型之上抽象出 option 层，负责对下层动作进行有效指导，但在自主创建动作抽象的过程中面临着挑战。为解决 option 自主形成的问题，Bacon 等[15]推导出了 option 策略梯度定理，并提出了一种新的 option-critic 体系结构。该体系结构能够学习内部 option 策略和中断函数，且不需要提供任何额外的奖励或子目标。为解决不能在 option 框架上直接利用 MDP 模型中最新的策略梯度优化算法的难题，Zhang 等[16]提出将 option 框架中的 SMDP 模型重构为两个增广 MDP 模型，使得所有策略优化算法均可用于 option 策略和 option 选取策略的学习，并在每一层增广 MDP 模型上应用 actor-critic 算法，形成了双行动者-评论家框架算法(Double Actor-Critic，DAC)。为提升基于 option 分层强化学习方法的数据效率，Smith 等[17]将异策略引入 option 框架，并且使用推理方法同时对 agent 所有可用的 option 进行更新。Osa 等[18]提出了基于优势加权信息最大化的分层强化学习算法(Hierarchical Reinforcement Learning via Advantage-Weighted Information Maximization，Adinfo)，将 option 框架与双延迟深度确定策略梯度算法(Twin Delayed Deep Deterministic Policy Gradient，TD3)[19]进行结合，同时引入优势加权信息来使得状态-动作空间与对应的 option 策略相匹配。Li 等[20]借鉴 Adinfo 思想，提出了软 option 行动者-评论家框架(Soft Option Actor-Critic Architecture，SOAC)，不仅在分层结构中使用互信息作为内在奖励来增强内部 option 策略的判别能力，同时还进一步提高了 agent 的探索效率，将目前解决单一任务中的优秀算法 SAC 引入 option 框架，并利用概率推理模型将优化问题简化为拟合最优轨迹问题。

上述算法在提升分层结构数据效率的同时，还造成了策略训练过程的不稳定性。针对这一问题，本文提出了一种基于轨迹信息量的分层强化学习方法，利用 option 轨迹信息选择动作，组成新的轨迹信息从而产生推断 option，计算其与原始 option 的相似度，并将其作为下层训练的内部奖励。经过双层网络结构的迭代更新，下层动作的选择符合当前 option 策略，使算法更加稳定。

本文的主要贡献可以总结为以下 3 点：

1)提出了通过 option 轨迹信息引导智能体选择动作的思想，并利用鉴别器生成内部奖励，用于分层结构算法的训练。

2)将轨迹信息内部奖励与 DAC 框架结合，提出了 THRL 方法，进一步加强了双层结构策略训练的稳定性。

3)将 THRL 方法运用于经典的连续控制任务中进行对比实验，验证了本文方法的优越性。

2 相关工作

2.1 MDP 与 SMDP

强化学习旨在学习一种从情景到动作的映射，以使智能体得到的累计奖赏最大化。强化学习的目标是给定一个马尔可夫决策过程，通过 agent 与环境的不断交互，寻找其最优策略 π。定义有限空间下马尔可夫决策过程为一个五元组(S,A,P,R,γ)，其中 S 为有限的状态集，A 为有限的动作集，P 为状态转移概率，$P_{ss'}^{a}=\mathbb{P}[s_{t+1}=s' \mid s_t=s,a_t=a]$，$R$ 为奖赏函数，$R_s^a=\mathbb{E}[r_{t+1} \mid s_t=s,a_t=a]$，$\gamma$ 为折扣因子，$\gamma\in(0,1]$。在当前状态 s 下，agent 通过与环境进行交互，依据当前策略采取一个动作 a 后，会得到一个即时奖赏，之后根据状态转移概率 P 到达下一个状态 s'，直到 agent 到达终止状态或者限制的时间步长，最终根据其累计奖励构成的状态值函数和状态-动作值函数，来学习最优策略 π。

马尔可夫决策过程中，在当前状态选择动作后，根据状态转移概率跳转至下一状态，马尔可夫性表示其下一状态仅与当前状态有关。但在某些情况下，选择的动作在多个时间步长完成后才会体现其价值，对于这类情况，经典的 MDP 不能进行很好的处理，由此产生了半马尔可夫决策过程。一般的 MDP 描述的过程是具有离散时间特性的，时间点之间具有相同的间隔，agent 在每个时刻执行一个动作。而 SMDP 描述的事件过程是 MDP 的扩展，决策点之间的时间间隔通常不相同，而且以往对 SMDP 的研究不关注每个离散事件内部过程的特性。MDP 与 SMDP 的关系如图 1 所示。

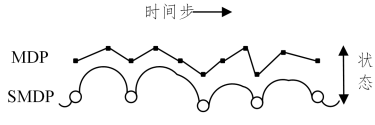


图 1 MDP 与 SMDP

Fig. 1 MDP and SMDP

2.2 Option 框架

根据 Sutton 等[14]提出的观点，基于 option 的分层方法兼具 MDP 和 SMDP 的特性。相比一般的 MDP，option 框架在单层结构上抽象出了一个 option 决策层，使整个过程具有像 SMDP 一样的离散特性；同时相比 SMDP，option 方法也关注每个 option 内部的过程变化，而且内部的变化过程也同样具有马尔可夫性。option 和一般的 MDP 中 action 的关系可以理解为，option 是时序扩展的 action。基于 option 的分层方法由三元组 (I_z,π_z,β_z) 构成，z 表示当前选择的 option，I_z 表示可以执行当前 option 的初始状态集合，β_z 表示当前 option 的中断函数，描述当前 option 的中断概率，中断则由顶层策略 π_Z 重新选择新的 option，其中 Z 表示 option 集合并假定其中所有的 option 都具备马尔可夫性。在 option 框架下，首先由顶层策略 π_Ω 选择 option，下层再根据选定的 option 选择 action。在时间步为 t 时，agent 在状态 s_t 以 $\beta_{z_{t-1}}(s_t)$ 的概率中止当前 option，并根据 $\pi_\Omega(\cdot|s_t)$ 选择一个新的 option z_t，或者以 $1-\beta_{z_{t-1}}(s_t)$ 的概率继续执行之前的 option z_{t-1}，此时 agent 依据状态转移概率 $p(\cdot|s_t,a_t)$ 到达新的状态 s_{t+1} 并获得奖赏 r_{t+1}。基于以上观点建立的模型，可以得到：

$$P(s_{t+1}|s_t,z_t)=\sum_a \pi_{z_t}(a|s_t)p(s_t,a) \tag{1}$$

$$p(z_t|s_t,z_{t-1})=(1-\beta_{z_{t-1}}(s_t))\phi_{z_{t-1}}=z_t+\beta_{z_{t-1}}(s_t)\ \pi(z_t|s_t) \tag{2}$$

其中，ϕ 为指示函数，定义状态-option 的奖赏函数为 $r(s,z)=\sum_a \pi_z(s,a)r(s,a)$，以及对于 SMDP 中构造的 option 和 MDP 模型，将状态-option-动作值函数定义为：

$$q_\pi(s,z,a)=\mathbb{E}_{s,z,r}\left[\sum_{i=1}^{\infty}\gamma^{i-1}r_{t+i}|s_t=s,z_t=z,a_t=a\right] \tag{3}$$

相应地，可以得到 SMDP 在策略 π 下的状态-option 值函数 $q(s,z)=\sum_a \pi_z(a|s)q_\pi(s,z,a)$ 和状态值函数 $v_\pi(s)=\sum_z \pi(z|s)q_\pi(s,z)$。

Bacon 等[15]在 option 理论基础上推导出了内部策略(intra-option)和中断函数的策略梯度定理，并将 option 框架与传统的无模型深度强化学习方法相结合，使智能体可以自行学习出 option 的选择策略，其形式与 MDP 的策略梯度定理保持高度一致。假定内部策略 $\{\pi_z\}_{z\in Z}$ 可以参数化为 φ，中断函数 $\{\beta_z\}_{z\in Z}$ 参数化为 ζ，证明得到：

$$\nabla_\varphi v_\pi(s_0)=\sum_{s,z}\rho(s,z|s_0,z_0)\left(\sum_a q_\pi(s,z,a)\nabla_\varphi \pi_z(a|s)\right) \tag{4}$$

$$\nabla_\zeta v_\pi(s_0)=-\sum_{s',z'}\rho(s',z|s_1,z_0)(q_\pi(s',z)-v_\pi(s'))\nabla_\zeta \beta_z(s') \tag{5}$$

2.3 双行动者-评论家框架

Zhang 等[16]将传统 SMDP 模型重构为两个增广 MDP 模型——上层 MDP 与下层 MDP，在上层 MDP 模型中，agent 根据主策略和终止函数进行决策，随后对其上层策略进行更新；在下层 MDP 模型中，类似地，agent 根据内部策略进行决策并更新。双层 MDP 模型虽然在结构上有很大差异，但它们都使用来自原 SMDP 模型的样本进行更新。

上层 MDP 模型的描述如下：对经典 MDP 模型的五元组构成进行延伸，定义上层 MDP 模型为 $M^H=(S^H,A^H,p^H,r^H,\gamma)$，其中，$S^H=Z\times S$，$A^H=Z$，$r^H(s_t^H,a_t^H)=r(s_t,z_t)$，$p^H(s_{t+1}^H|s_t^H,a_t^H)=\phi_{a_t^H=z_t}p(s_{t+1}|s_t,z_t)$。

下层 MDP 模型的描述如下：受限于上层 option 策略的下层增广 MDP 模型与经典 MDP 模型的定义类似，定义下层 MDP 模型为 $M^L=(S^L,A^L,p^L,r^L,\gamma)$，其中，$S^L=A\times S$，$A^L=A$，$r^L(s_t^L,a_t^L)=r(s_t,a_t)$，$p^L(s_{t+1}^L|s_t^L,a_t^L)=p(s_{t+1}|s_t,a_t)p(z_{t+1}|s_t,z_t)$。

DAC 框架中上层策略 π^H 表示为：

$$\pi^H(a_t^H|s_t^H)=\begin{cases}p(z_t|s_t,z_{t-1}), & t>1\\ \pi(s_t,z_t), & t=1\end{cases} \tag{6}$$

表示在状态 s_t 当前执行 option z_{t-1} 的情况下，下次执行 option z_t 的概率。

下层策略 π^L 为 $\pi^L(a_t^L|s_t^L)=\pi_{z_t}(a_t|s_t)$，表示在当前执行 option z_t 的前提下，下层在状态 s_t 执行动作 a_t 的概率。

为更好地理解上下层更新相互适应的特点，DAC 框架给出了两个解释：

1) M^H 依赖于 π_z，同时 π^H 依赖于 π 和 β_z；

2) M^L 依赖于 π 和 β_z，同时 π^L 依赖于 π_z。

其中，1)表示固定内部 option(即下层策略)π_z 更新上层策略 π^H(即隐式更新 π 和 β_z)；2)表示固定 option 选择策略 π 和终止函数 β_z(即上层策略组成部分)更新下层策略 π^L(即隐式更新 π_z)。

3 基于轨迹信息量的分层强化学习方法

本章详细介绍了 THRL 的工作原理，包括对 THRL 组成要素的引入分析、轨迹信息判别结构的介绍以及算法的整体训练流程。

3.1 问题分析

在许多分层强化学习问题中，上层 option 策略对下层智能体选取动作进行引导。在该过程中，受 MDP 特性的影响，动作的选取仅考虑了上一时刻的状态信息，而忽略了上层 SMDP 特性中增广状态带来的一系列额外策略信息。在 option 层，每个增广状态可能包含多个状态-动作信息，这些额外信息对智能体下一时刻的动作选取发挥着重要作用。

3.2 轨迹信息鉴别

为了解决分层强化学习算法更新过程中因未充分考虑轨迹信息所带来的下层策略不稳定问题，本文将更多的轨迹信息加入到下层动作选择的必要因素中，将鉴别器结构运用到 DAC 算法中。

通过不同轨迹信息输入以及 option 策略共同作用生成下层动作，该动作与输入的轨迹信息组成扩展轨迹信息，将扩展轨迹信息输入神经网络产生推断 option。为使下层动作的生成更加符合原 option 策略，将推断 option 与原 option 输入鉴别器，以计算其相似程度并将其作为下层训练过程中的内部奖励。通过智能体的策略评估以及更新的迭代过程，最大化内部奖励(即最大化推断 option 与原 option 的相似程度)，从而使下层动作的选取更满足上层策略的引导。轨迹信息判别结构如图 2 所示。

推断 option 和当前 option 的相似度衡量表示为：

$$d_\sigma(\cdot)=-\|MLP(\tilde{s}_c)-z_t\|_2^2 \tag{7}$$

其中，MLP 表示多层感知器，即多层全连接网络，$\tilde{s}_c$ 表示

扩展轨迹信息，即轨迹信息 $\tilde{s}_t$ 与当前动作 a_t 的组合，将扩展轨迹信息输入 MLP 网络产生推断 option，z_t 表示当前 option，σ 表示鉴别器网络参数。d_σ 越大，表示推断 option 与当前 option 越相似，即当前选择的动作越合适。

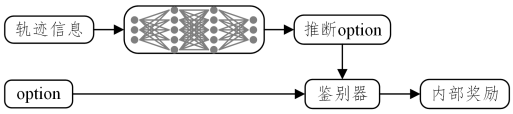


图 2 轨迹信息判别器

Fig. 2 Trajectory information discriminator

受 Levine[21] 的启发，THRL 将 SAC 算法作为底层实现，可在 DAC 框架中构建相应的上层状态值函数 $V^H(s_t^H)$、状态-动作值函数 $Q^H(s_t^H,a_t^H)$ 及下层状态-动作值函数 $Q^L(s_t^L,a_t^L)$。

$V^H(s_t^H)$ 与 $Q^H(s_t^H,a_t^H)$ 的关系表示为：

$$V^H(s_t^H)=\mathbb{E}_{\pi^H(a_t^H|s_t^H)}[Q^H(s_t^H,a_t^H)-\alpha_1\log\pi^H(a_t^H|s_t^H)] \tag{8}$$

$Q^H(s_t^H,a_t^H)$ 与 $Q^L(s_t^L,a_t^L)$ 的关系表示为：

$$\begin{aligned}Q^L(s_t^L,a_t^L)&=V^L(s_t^L)\\&=\mathbb{E}_{\pi^L(a_t^L|s_t^L)}[Q^L(s_t^L,a_t^L)-\alpha_2\log\pi^L(a_t^L|s_t^L)]\end{aligned} \tag{9}$$

$Q^L(s_t^L,a_t^L)$ 与 $V^H(s_t^H)$ 的关系表示为：

$$Q^L(s_t^L,a_t^L)=r(s_t^L,a_t^L)+d_\sigma(\tilde{s}_c,a_t^H)+\gamma\,\mathbb{E}_{p(s_{t+1}^H|s_t^H,a_t^H)}[V^H(s_{t+1}^H)] \tag{10}$$

其中，$\log\pi^H(\cdot)$ 与 $\log\pi^L(\cdot)$ 分别表示上层和下层的最大熵目标，增大智能体获取到最优 option 和最优动作的概率。内部奖励 d_σ 作为下层 Q 函数的一部分，与环境奖赏 r 共同影响当前状态-动作对的价值评估，使智能体优化当前动作策略。

3.3 THRL 算法

THRL 算法以 DAC 框架为主要的分层结构，并将 SAC 算法中的异策略与最大熵模型引入上下层的 Actor-Critic 算法来实现。THRL 算法的双层结构中，每层结构都包含两个 Q 评估网络、目标 Q 网络和策略网络，上下层采用相同经验缓冲池中不同批量样本进行训练，采用函数逼近和随机梯度下降方法来训练和评估上下层 Q 函数和动作策略。

THRL 算法的结构如图 3 所示，将训练过程分为两部分，在每次迭代循环中上下层以随机交替顺序分别对其行动者和评论家网络进行训练更新。算法结构中所有神经网络模型均采用全连接网络，具有相同维度的隐层单元，但不同的网络模型具有不同维度的输入与输出。在上层结构中，根据评论家网络产生的 Q 值与行动者网络产生的策略循环迭代更新，得到最优的 option 策略；下层结构中，行动者网络以状态-option 为输入，评论家网络以状态-option-动作为输入，鉴别器网络以推断 option 和当前 option 为输入，根据当前 option 策略和其历史轨迹信息来选择合适的动作，并且由历史轨迹信息和最新选取的动作组成扩展轨迹信息产生推断 option，计算其与当前 option 的相似度以产生内部奖励。将上述内部奖励作为下层 Q^L 函数的组成部分，从而在下层迭代循环中使 agent 逐渐稳定地选取合适的动作。THRL 的伪代码如算法 1 所示，其中 15－18 行和 19－22 行分别表示上下层网络训练的更新过程。在第 19 行，下层动作值函数 Q^L 由环境奖赏、内部奖励和下一时刻状态值函数的期望共同决定。在 14 行求解 Actor/Critic 网络梯度时，内部奖励的大小影响了 Q^L 对当前状态-动作对的评价。若轨迹信息与当前动作的组合信息在鉴别器中产生了较大的内部奖励，表明在当前状态下该动作符合 option 策略，继而产生了较大的动作值函数 Q^L。在网络模型参数更新过程中，参数的更新幅度越大，该动作被选择的概率越高。

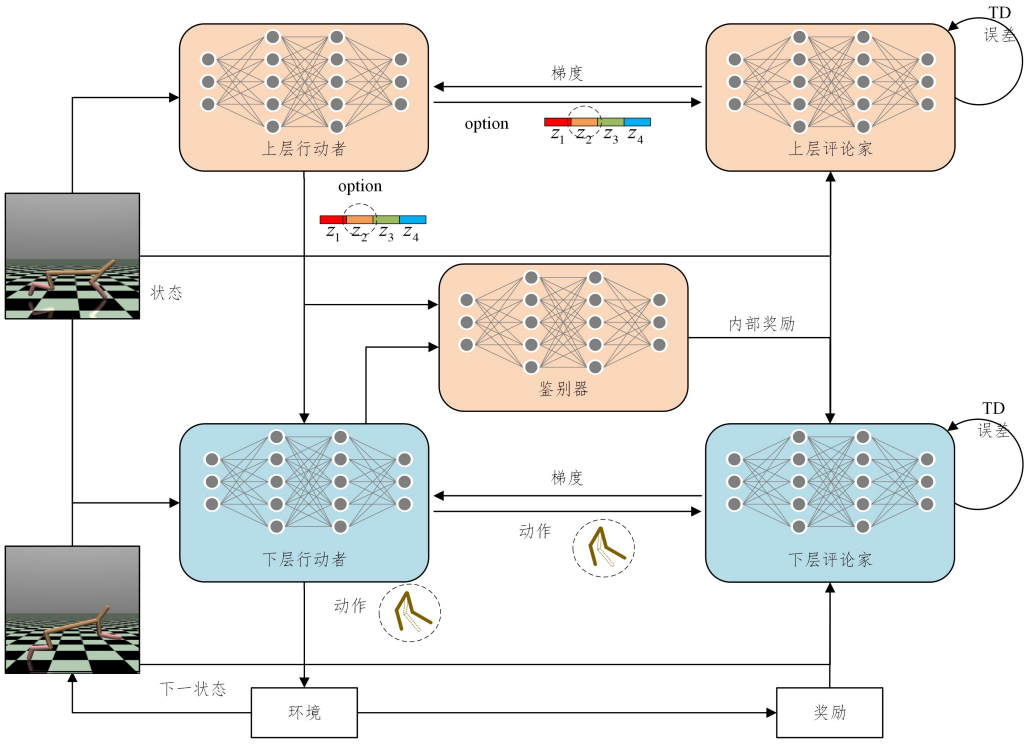


图 3 THRL 的结构

Fig. 3 Architecture of THRL

算法 1 THRL 算法

1. 参数初始化
2. 初始化双层 Actor/Critic 网络参数 $\phi_i, \theta_i, \zeta, \psi$
3. 将 Critic 网络参数赋值给目标网络 $\bar{\phi}_i \leftarrow \phi_i, \bar{\theta}_i \leftarrow \theta_i$
4. 将 Actor 网络参数赋值给目标网络 $\bar{\zeta} \leftarrow \zeta, \bar{\psi} \leftarrow \psi$
5. for 每次迭代 do
6. for 每个采样步 do
7. 根据上层策略 $\pi^H(\cdot|(z_{t-1}, s_t))$选择动作 z_t
8. 根据下层策略 $\pi^L(\cdot|(\tilde{s}_t, z_t))$选择动作 a_t
9. agent 执行动作 a_t 得到 r_{t+1}, s_{t+1}
10. 将$(s_t, z_t, a_t, r_{t+1}, s_{t+1})$存入经验池 D 中
11. end for
12. for 每个训练步 do
13. 从经验池 D 中随机选取 N 个样本进行训练
14. 分别计算 Actor/Critic 网络的梯度

// 两层训练优化过程以随机顺序交替进行层训练优化：

15. $\phi_i \leftarrow \phi_i - \lambda_{Q^H} \nabla_{\wedge\phi_i} J_{Q^H}(\phi_i)$, for $i \in \{1,2\}$
16. $\psi \leftarrow \psi - \lambda_{\pi^H} \nabla_{\wedge\psi} J_{\pi^H}(\psi)$

// 更新上层目标网络参数：

17. $\bar{\phi}_i \leftarrow \tau\phi_i + (1-\tau)\bar{\phi}_i$, for $i \in \{1,2\}$
18. $\bar{\psi} \leftarrow \tau\psi + (1-\tau)\bar{\psi}$

// 下层训练优化：

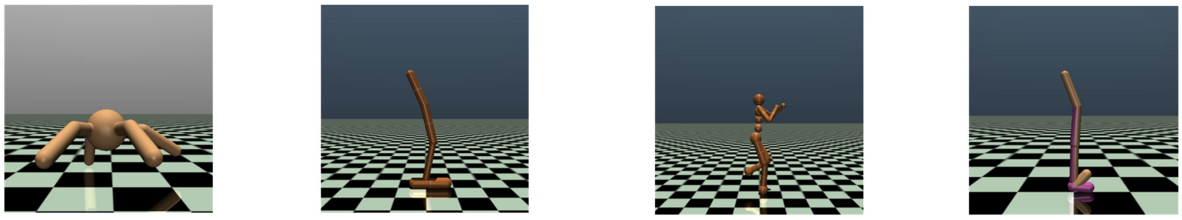
19. $\theta_i \leftarrow \theta_i - \lambda_{Q^L} \nabla_{\wedge\theta_i} J_{Q^L}(\theta_i)$, for $i \in \{1,2\}$
20. $\zeta \leftarrow \zeta - \lambda_{\pi^L} \nabla_{\wedge\zeta} J_{\pi^L}(\zeta)$

// 更新下层目标网络参数：

21. $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$, for $i \in \{1,2\}$
22. $\bar{\zeta} \leftarrow \tau\zeta + (1-\tau)\bar{\zeta}$
23. end for
24. end for

4 实验及分析

为了验证 THRL 算法的有效性，本文主要在 4 个密集奖励和 1 个稀疏奖励环境任务上进行实验，同时增加了不同类型轨迹信息引导下的 THRL 算法的性能对比实验。所有实验参数的设置如下：经验池容量设置为 100 万，学习率为 3×10^{-4}，折扣权重为 0.99，神经网络激活函数为 ReLU，优化器选择 Adam，分层强化学习方法 option 个数为 4，每个情节中最大时间步数为 1000。



(a) Ant-v2 (b) Hopper-v2 (c) Humanoid-v2 (d) Walker2d-v2

图 4 MuJoCo 环境

Fig. 4 MuJoCo environments

4.1 密集奖励实验

4.1.1 实验设置

为了验证本文算法的有效性，将其与经典的深度强化学习算法 DAC，SAC，SOAC 运用于 MuJoCo[22]环境任务中进行实验。MuJoCo 环境常见任务如图 4 所示，环境参数如表 1 所列。以下为实验所用环境对应的任务介绍：

1) Ant-v2：使四足生物尽可能快地往前爬。

2) Hopper-v2：使二维一足机器人尽可能快地向前跳。

3) Humanoid-v2：使三维双足行走的机器人尽可能快地向前走。

4) Walker2d-v2：使二维双足行走的机器人尽可能快地向前走。

在相同实验环境下对这 4 种算法进行 3 次随机种子的实验，并取平均值来比较算法的性能。

表 1 环境参数

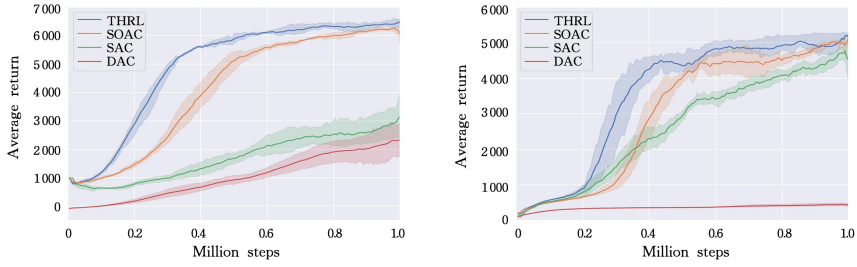
Table 1 Environment parameters

环境	状态维度	动作维度
Ant-v2	111	8
Hopper-v2	11	3
Humanoid-v2	376	17
Walker2d-v2	17	6

4.1.2 实验结果分析

将基于 option 的分层强化学习算法、经典 SAC 算法与 THRL 算法运用于 Ant-v2，Humanoid-v2，Hopper-v2 和 Walker2d-v2 实验中，THRL 算法的总体表现明显优于对比算法。其实验结果如图 5 所示，图中横坐标表示智能体执行的环境步数，纵坐标表示算法执行多次之后的回报均值，阴影部分表示多次实验的回报方差。从图 5 中可以看出，在性能方面，THRL 算法在训练 60 万步时 Ant-v2 环境中已经获得大约 6000 的性能分数、Humanoid-v2 环境中获得大约 4800 的性能分数、Hopper-v2 环境中获得大约 3400 的性能分数以及 Walker2d-v2 环境中获得大约 4300 的性能分数，其他对比算法的性能表现不佳；在收敛速率方面，THRL 算法在 Ant-v2，Humanoid-v2 环境中 60 万步时基本完全收敛，在 Hopper-v2 环境中 30 万步时基本收敛，对比算法中表现较好的 SOAC 算法在训练过程中的收敛速率也滞后于 THRL 算法；在稳定性方面，图中曲线阴影部分可以表现出算法在不同随机种子训练过程中的稳定程度，THRL 算法在 Ant-v2，Hopper-v2 和 Walker2d-v2 环境中具有良好的稳定性，由于 Humanoid-v2 环境的复杂性，THRL 算法在训练稳定性方面和对比算法较为接近。在 4 种不同的 MuJoCo 环境中，THRL 算法均明显优于

原始算法，主要考虑 THRL 算法在 DAC 框架的基础上融入 SAC 算法，在上层 option 引导下层动作时嵌入历史轨迹信息，使得 THRL 算法能够更快更稳定地达到收敛状态。算法在各实验环境中的最终性能数据如表 2 所列。



(a) Ant-v2　(b) Humanoid-v2

(c) Hopper-v2　(d) Walker2d-v2

图 5　连续控制任务训练曲线

Fig. 5　Training curves for several continuous control tasks

表 2　实验算法的最终性能

Table 2　Final performance of experimental algorithms

环境	THRL	SOAC	SAC	DAC
Ant-v2	**6519.02±98.92**	6197.50±363.79	3153.62±762.44	2314.76±546.71
Hopper-v2	**3443.26±56.56**	3367.04±139.97	2959.48±306.41	1884.72±345.35
Humanoid-v2	**5164.42±197.83**	4993.81±272.65	4548.49±447.03	416.28±23.09
Walker2d-v2	**4687.49±60.32**	4106.11±505.36	3792.67±689.91	1888.54±59.12

4.2　稀疏奖励实验

4.2.1　实验设置

为了进一步验证本文改进算法的有效性，将 THRL 算法运用到更加困难的稀疏奖励环境 SparseHumanoid-v2 中，并与经典的 SAC 算法进行比较分析。SparseHumanoid-v2 继承自 MuJoCo 平台的 Humanoid-v2 环境接口，与 Humanoid-v2 类似，每情节最大步数为 1000，且一旦智能体跌倒，该情节就结束。但不同的是，在 SparseHumanoid-v2 环境中只有智能体运动的质心高于 0.6 时才会获得 +1 的奖励。在相同实验环境下对这两种算法进行 3 次随机种子的实验，取实验结果的平均值来比较算法的性能。

4.2.2　实验结果分析

将经典 SAC 算法与 THRL 算法运用于稀疏奖励环境 SparseHumanoid-v2 中，从图 6 中可以看出，THRL 算法在 80 万步时已经基本收敛，并具有明显的性能优势。SAC 算法引入了熵正则化项，使其具有更高的探索效率，增加了智能体在连续状态空间中执行最优动作的概率，从而获得了不错的性能表现。但 SAC 算法在前期进行了大量的空间样本探索，导致其在训练过程中出现了严重的稳定性波动。THRL 算法在分层结构中分别将 SAC 算法作为底层实现，在训练策略过程中由于考虑了 option 历史轨迹信息，因此能够更加有效地引导动作选择，从而获得较高的稳定性。算法的最终性能数据如表 3 所列。

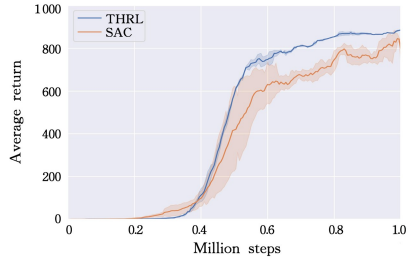


图 6　稀疏奖励任务训练曲线

Fig. 6　Training curves for sparse reward task

表 3　THRL 与 SAC 的最终性能

Table 3　Final performance of THRL and SAC

环境	THRL	SAC
SparseHumanoid-v2	**886.15±0.10**	798.57±39.62

4.3　轨迹信息对比

轨迹信息类型的选取包括状态、动作、状态动作和状态差异 4 个方面，分别考虑了下一时刻状态、当前时刻状态-动作、当前时刻动作-下一时刻状态以及下一时刻状态与当前时刻的差异。不同类型的轨迹信息引导的动作会有显著差异，可能使推断 option 时有很大不同，导致最后的相似程度较小。

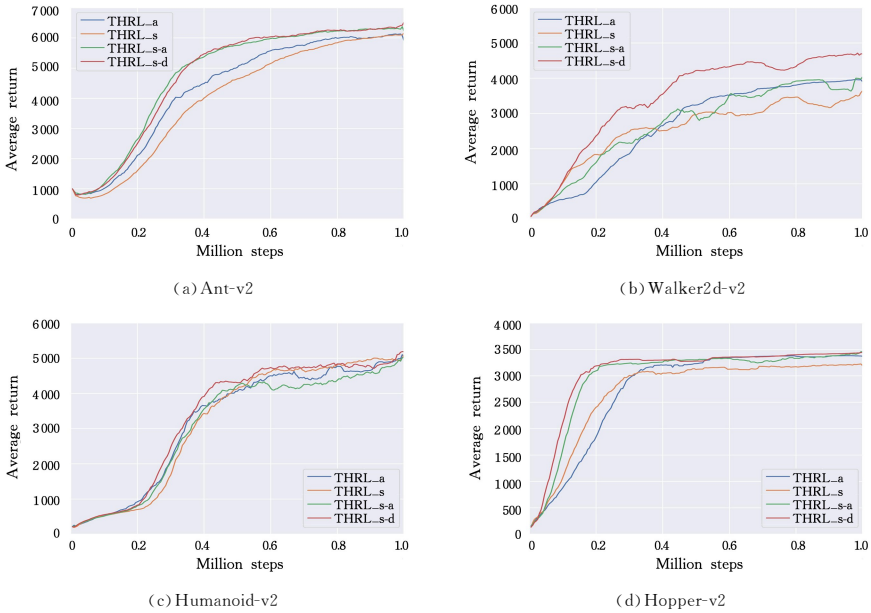
4.3.1　实验设置

为了验证 THRL 算法在不同类型轨迹信息引导下的

有效性，将 THRL_s（状态信息）、THRL_a（动作信息）、THRL_s-a（状态-动作信息）和 THRL_s-d（状态差异）4 种不同的 THRL 变体算法运用于 MuJoCo 环境任务中。在相同实验环境下对这 4 种算法进行 3 次随机种子的实验，取实验结果的平均值来比较算法的性能。

4.3.2 实验结果分析

图 7 表明，不同类型的轨迹信息引导在算法表现上存在着显著差异。基于状态差异信息的 THRL 方法（THRL_s-d）在大部分任务中取得了较好的表现，其他包含单一状态或动作的方法则表现不佳。THRL_s-d 将此刻状态与上一时刻状态的差异度量作为输入，充分考虑了相似状态下、相同 option 的智能体执行动作相似策略，能够更好地进行引导。其他类型的轨迹信息方法重点关注轨迹的可持续长度，通过最大化每个 option 的轨迹长度来保证稳定性。但随着状态信息的不断改变，之后的状态不再适应当前 option，从而导致算法不再具备较好的性能表现。



(a) Ant-v2 (b) Walker2d-v2

(c) Humanoid-v2 (d) Hopper-v2

图 7 不同轨迹信息方法的训练曲线

Fig. 7 Training curves for different trajectory information

结束语 本文针对原始基于 option 的分层强化学习算法在训练过程中会出现下层策略更新不稳定、收敛速率慢、总体性能差等问题，提出了一种基于轨迹信息量的分层强化学习算法——THRL 算法。该方法有效利用了 option 引导所产生的轨迹信息，并通过扩展动作信息来推断 option，从而使下层动作选择更加符合当前 option 策略。将 THRL 算法分别运用于密集奖励和稀疏奖励环境，对比验证该算法的有效性，同时根据不同类型的 option 轨迹信息对比实验来获得更合适的信息量。实验结果表明，THRL 算法的稳定性、收敛速率以及总体性能均优于原始的基于 option 的分层强化学习算法和经典 SAC 算法，同时表明了在不同环境中不同类型轨迹信息的有效性。

本文研究了多种类型 option 轨迹信息，但仅考虑了使用中断函数作为判定条件的单步信息。对于多步信息，通过固定 option 轨迹信息长度的方式无法进行有效评估，如何动态调整轨迹信息长度以最大化引导优势，将是下一步工作的研究重点。

参 考 文 献

[1] SUTTON R S, BARTO A G. Reinforcement learning: An introduction [M]. MIT press, 2018.

[2] GOODFELLOW I, BENGIO Y, COURVILLE A, et al. Deep learning [M]. MIT press, 2016.

[3] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning [J]. Nature, 2015, 518(7540): 529-533.

[4] SILVER D, SCHRITTWIESER J, SIMONYAN K, et al. Mastering the game of go without human knowledge [J]. Nature, 2017, 550(7676): 354-359.

[5] SALLAB A E, ABDOU M, PEROT E, et al. Deep reinforcement learning framework for autonomous driving [J]. Electronic Imaging, 2017, 2017(19): 70-76.

[6] GOTTIPATI S K, SATTAROV B, NIU S, et al. Learning to navigate the synthetically accessible chemical space using reinforcement learning [C]// International Conference on Machine Learning. PMLR, 2020: 3668-3679.

[7] LIU Q, ZHAI J W, ZHANG Z Z, et al. A survey on deep reinforcement learning [J]. Chinese Journal of Computers, 2018, 41(1): 1-27.

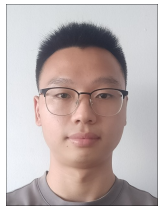
[8] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning [C]// ICLR. 2016.

[9] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning [C]// International Conference on Machine Learning. 2016: 1928-1937.

[10] BARTO A G, MAHADEVAN S. Recent advances in hierarchical reinforcement learning [J]. Discrete Event Dynamic Systems, 2003, 13(4): 341-379.

[11] RASHID T, SAMVELYAN M, SCHROEDER C, et al. Qmix: Monotonic value function factorisation for deep multi-agent rein-

forcement learning[C]// International Conference on Machine Learning. PMLR,2018:4295-4304.
[12] SCHULMAN J,WOLSKI F,DHARIWAL P,et al. Proximal policy optimization algorithms[J]. arXiv:1707.06347,2017.
[13] HAARNOJA T,ZHOU A,ABBEEL P,et al. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor[C]// International Conference on Machine Learning. 2018:1861-1870.
[14] SUTTON R S,PRECUP D,SINGH S. Between mdps and semi-mdps:A framework for temporal abstraction in reinforcement learning[J]. Artificial Intelligence,1999,112(1/2):181-211.
[15] BACON P L,HARB J,PRECUP D. The option-critic architecture[C]// AAAI Conference on Artificial Intelligence. 2017:1726-1734.
[16] ZHANG S,WHITESON S. Dac:The double actor-critic architecture for learning options[C]// Advances in Neural Information Processing Systems. 2019:2012-2022.
[17] SMITH M,HOOF H,PINEAU J. An inference-based policy gradient method for learning options[C]// International Conference on Machine Learning. PMLR,2018:4703-4712.
[18] OSA T,TANGKARATT V,SUGIYAMA M. Hierarchical Reinforcement Learning via Advantage-Weighted Information Maximization[C]// International Conference on Learning Representations. 2018.
[19] FUJIMOTO S,HOOF H,MEGER D. Addressing function approximation error in actor-critic methods[C]// International Conference on Machine Learning. 2018:1587-1596.
[20] LI C,MA X,ZHANG C,et al. SOAC:The Soft Option Actor-Critic Architecture[J]. arXiv:2006.14363,2020.
[21] LEVINE S. Reinforcement learning and control as probabilistic inference:Tutorial and review[J]. arXiv:1805.00909,2018.
[22] BROCKMAN G,CHEUNG V,PETTERSSON L,et al. Openai gym[J]. arXiv:1606.01540,2016.



XU Yapeng, born in 1996, postgraduate. His main research interests include hierarchical reinforcement learning and deep reinforcement learning.

LIU Quan, born in 1969, Ph.D, professor, Ph.D supervisor, is a member of China Computer Federation. His main research interests include deep reinforcement learning and automated reasoning.

(责任编辑:喻藜)