软件学报 ISSN 1000-9825, CODEN RUXUEW
2025,36(10):4695−4709 [doi: 10.13328/j.cnki.jos.007296] [CSTR: 32375.14.jos.007296]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

扩散模型期望最大化的离线强化学习方法*

刘 全[1,2], 颜 洁[1], 乌 兰[1]

[1](苏州大学 计算机科学与技术学院, 江苏 苏州 215008)

[2](江苏省计算机信息处理技术重点实验室 (苏州大学), 江苏 苏州 215006)

通信作者: 刘全, E-mail: quanliu@suda.edu.cn

摘 要: 在连续且密集奖励的任务中, 离线强化学习取得了显著的效果. 然而由于其训练过程不与环境交互, 泛化能力降低, 在离散且稀疏奖赏的环境下性能难以得到保证. 扩散模型通过加噪结合样本数据邻域的信息, 生成贴近样本数据分布的动作, 强化智能体的学习和泛化能力. 针对以上问题, 提出一种扩散模型期望最大化的离线强化学习方法 (offline reinforcement learning with diffusion models and expectation maximization, DMEM). 该方法通过极大似然对数期望最大化更新目标函数, 使策略具有更强的泛化性. 将扩散模型引入策略网络中, 利用扩散的特征, 增强策略学习数据样本的能力. 同时从高维空间的角度看期望回归更新价值函数, 引入一个惩戒项使价值函数评估更准确. 将 DMEM 应用于一系列离散且稀疏奖励的任务中, 实验表明, 与其他经典的离线强化学习方法相比, DMEM 性能上具有较大的优势.

关键词: 离线强化学习; 扩散模型; 优势函数加权; 期望回归; 期望最大化

中图法分类号: TP18

中文引用格式: 刘全, 颜洁, 乌兰. 扩散模型期望最大化的离线强化学习方法. 软件学报, 2025, 36(10): 4695–4709. http://www.jos.org.cn/1000-9825/7296.htm

英文引用格式: Liu Q, Yan J, Wu L. Offline Reinforcement Learning Method with Diffusion Model and Expectation Maximization. Ruan Jian Xue Bao/Journal of Software, 2025, 36(10): 4695–4709 (in Chinese). http://www.jos.org.cn/1000-9825/7296.htm

Offline Reinforcement Learning Method with Diffusion Model and Expectation Maximization

LIU Quan[1,2], YAN Jie[1], WU Lan[1]

[1](School of Computer Science and Technology, Soochow University, Suzhou 215008, China)

[2](Jiangsu Provincial Key Laboratory for Computer Information Processing Technology (Soochow University), Suzhou 215006, China)

Abstract: Offline reinforcement learning has yielded significant results in tasks with continuous and intensive rewards. However, since the training process does not interact with the environment, the generalization ability is reduced, and the performance is difficult to guarantee in a discrete and sparse reward environment. The diffusion model combines the information in the neighborhood of the sample data with noise addition to generate actions that are close to the distribution of the sample data, which strengthens the learning and generalization ability of the agents. To this end, offline reinforcement learning with diffusion models and expectation maximization (DMEM) is proposed. The method updates the objective function by maximizing the expectation of the maximum likelihood logarithm to make the strategy more generalizable. Additionally, the diffusion model is introduced into the strategy network to utilize the diffusion characteristics to enhance the ability of the strategy to learn data samples. Meanwhile, the expectile regression is employed to update the value function from the perspective of high-dimensional space, and a penalty term is introduced to make the evaluation of the value function more accurate. DMEM is applied to a series of tasks with discrete and sparse rewards, and experiments show that DMEM has a large advantage in performance over other classical offline reinforcement learning methods.

Key words: offline reinforcement learning; diffusion model; advantage function weighted; expectile regression; expectation maximization

* 基金项目: 国家自然科学基金 (62376179, 62176175); 新疆维吾尔自治区自然科学基金 (2022D01A238); 江苏高校优势学科建设工程
收稿时间: 2024-05-06; 修改时间: 2024-07-18; 采用时间: 2024-09-05; jos 在线出版时间: 2025-02-19
CNKI 网络首发时间: 2025-02-19

强化学习 (reinforcement learning, RL)[1]是机器学习领域中的一种重要的学习方法. 通常以马尔可夫决策过程 (Markov decision process, MDP) 来表示环境的信息. 深度强化学习 (deep reinforcement learning, DRL)[2]将深度学习 (deep learning, DL)[3]与 RL 相结合, 兼备 DL 的信息感知能力和 RL 的决策控制能力, 形成一种端到端的完整智能系统. 离线强化学习 (offline reinforcement learning, ORL)[4]是强化学习的一个变种, 将经典的强化学习算法或者深度强化学习算法利用在静态数据集上, 预训练环节脱轨于大规模的数据收集, 在策略学习时不需要任何的交互. 解决了大规模在线预训练有限的问题, 减少了计算成本.

目前 ORL 是强化学习最热的子方向之一, 利用预先收集的大规模静态数据集来训练强化学习智能体. 其核心目的是在固定的数据集上训练出一个好的策略, 能在未知数据分布的数据集上表现良好. 但在这种环境下, 由于数据的分布不同, 会产生分布偏移, 也叫外推误差, 成为 ORL 的主要问题. 解决外推误差的方法大致可以分为 4 类: 策略约束[5]、值函数正则[6]、不确定估计[7]和基于模型的方法[8]. Fujimoto 等人[9]在双延迟深度确定性策略梯度算法 (twin delayed deep deterministic, TD3)[10]的基础上, 提出了批量受限深度 Q 学习算法 (batch constrained deep Q-learning, BCQ), 从理论上解释了在连续任务中, 外推误差出现的原因以及如何消除. 并提出利用批量约束和添加动作扰动模型来避免外推误差. 显示约束策略, 这是最早提出的离线强化学习方法. 这方法主要是将动作的选择限制在离线数据集分布上, 从而避免出现 Q 值高估分布外动作的问题. 而保守的 Q 学习算法 (conservative Q-learning, CQL)[11]提出在 Q 值函数更新中添加一个正则项, 使得估计的期望值低于真实值, 打破对 Q 值高估的限制, 提高了对策略进行正确评估的能力. 为了能在分布外的动作上也学习到好的分数, 基于模型的离线策略优化算法 (model-based offline policy optimization, MOPO)[12]在基于模型的方法基础上, 添加了一个奖励惩罚项, 提高了模型的泛化能力. Kostrikov 等人[13]提出隐式 Q 学习方法 (implicit Q-learning, IQL), 采用 SARSA 方法和期望回归方法更新价值函数, 直接确定 Q 值如何随着不同的动作而变化, 并借助随机动态对未来结果进行平均. 其一方面通过价值函数评估, 降低模型与估计之间的差距, 另一方面利用 SARSA 方式的随机性特点, 提高策略的泛化能力.

泛化能力是指模型在遇到未曾见过的数据时的表现. 在离线强化学习领域, 模型通常基于离线数据集进行训练, 因此对泛化能力的关注尤为重要. 泛化性能差主要原因包括数据分布偏移和过拟合等问题. 常见的解决方案包括数据增强[14]、迁移学习[15]和对抗学习[16]等方法. Wang 等人[17]提出利用 3 种不同的状态增强技术: 随机失活法、混合法和缩放法, 以缓解离线多智能体分布偏移的问题. 通过对状态进行数据增强, 不仅扩大了数据集的规模, 还提高了局部泛化能力, 进而促进了算法更有效地学习策略. Qiao 等人[18]提出了软对抗离线强化学习算法 (soft adversarial offline reinforcement learning, SAORL), 通过降低 ORL 中对抗示例的攻击强度来学习软对抗示例. 对传统对抗示例提出了基于 Wasserstein 的约束, 学习软对抗示例的劣例优化问题, 提高智能体在最坏情况下的泛化能力. 但在稀疏奖励环境下, 经典 ORL 算法面临着无效增强、迁移后性能降低、计算成本增加等问题. 因此, 在离线强化学习的环境下, 为智能体提供有效泛化仍是亟待解决的难题.

而在 ORL 经典的学习任务中, 通常面临离散任务且稀疏奖励的问题, 即在一个很长的时间步内, 只有几个时间步可以得到奖励. 目前在强化学习中, Rengarajan 等人[19]提出离线指导在线学习算法 (learning online with guidance offline, LOGO), 使用离线演示数据, 将策略改进步骤与额外的策略指导步骤合并, 以次优策略定位学习策略, 同时又能够超越学习和接近最优策略. Liu 等人[20]提出了分层安全离线强化学习 (hierarchical safe offline reinforcement learning, HSORL), 利用分层强化学习框架, 通过用分层策略对不安全行为进行建模, 缓解了稀疏性问题. 其中, 在数据收集的时候, 不安全的状态-动作对通常是稀疏的, 因此不利于建模. 除了指导策略和分层强化学习外, Lin 等人[21]提出转换轨迹变压器算法 (switch trajectory Transformer, SwitchTT), 这是一种对轨迹变压器的多任务扩展, 利用稀疏激活模型, 来降低在多任务离线模型学习中的计算成本, 同时采用分布轨迹值估计器来提高在稀疏奖励环境下策略的性能. 本文引入扩散模型, 利用扩散模型的长视野的特点, 结合数据样本的邻域信息, 来缓解稀疏奖励的问题.

扩散模型 (diffusion models, DM)[22]类似于生成模型, 从噪声中生成数据样本, 是学习数据分布的一种方法. 随机采样数据样本, 加上随机噪声, 通过预测去噪网络参数, 生成目标数据. DM 结合数据样本的邻域信息, 在稀疏奖励的环境下, 即使不能得到下一状态的有效奖励, 也能通过扩散的形式, 生成接近样本数据的数据, 提升样本的利

用效率. 但是基于扩散模型的算法, 采样速度慢, 训练过程需要大量的时间和计算成本, 远没有经典的离线强化学习算法简单高效.

综上所述, 本文提出了一种基于扩散模型期望最大化的离线强化学习 (offline reinforcement learning with diffusion models and expectation maximization, DMEM) 方法. 该方法运用扩散模型改变了学习策略的网络结构, 并利用其预测噪声的参数学习数据分布, 生成目标数据. 再结合期望最大化框架, 通过猜测隐含参数, 极大化对数似然求解模型参数. 同时, 价值函数的更新使用期望回归和 SARSA 算法, 直接根据数据集选择动作, 而不是强制的选择价值函数最大的动作, 给予策略更多的泛化能力. 此外, 通过在不同随机种子和不同环境下的实验, 结果表明, DMEM 方法具有稳定且高效的性能.

本文的贡献主要包括以下 3 个方面.

(1) 引入扩散策略网络, 利用该网络来预测去噪参数, 并通过优势函数加权期望最大化的方法来更新策略, 解决样本数据中奖励稀疏的问题, 有效地提高了样本的泛化性能.

(2) 从高维的角度提出一种利用期望回归算法更新价值函数的方法. 通过添加一个惩戒项, 缓解了高维度带来样本偏差的问题. 另外从理论上, 证明了动作维度对价值损失函数的影响.

(3) 将 DMEM 方法应用于 AntMaze 环境的 6 个基准实验中, 通过与经典的离线强化学习算法对比, 验证了该算法的优越性.

1 相关工作

1.1 隐式 Q 学习

由于深度强化学习存在采样效率低、与环境交互成本高等缺点, 促使了离线强化学习的产生, 且逐渐成为机器学习领域的研究热点. 在离线 RL 中, 策略的学习与评估都使用一个大规模的静态数据集. 因此通常存在分布偏移, 即训练策略和行为策略不一致的问题. IQL 算法不直接学习分布外的动作 (OOD), 用数据集中已知的状态-动作对进行学习, 避免 OOD 带来的 Q 值高估问题和分布偏移问题. 但 IQL 算法也存在一些缺点, 包括样本效率低、过拟合风险高以及收敛速度较慢等. 由于该算法是基于已有的离线数据集进行学习, 如果数据质量不高或者覆盖范围不全面, 可能会影响算法的性能. 此外, 如果训练数据中存在噪声或偏差, IQL 算法可能会面临过拟合的风险, 导致所学到的策略在实际应用中表现不佳.

Hong 等人[23]提出超越统一抽样算法 (beyond uniform sampling, BUS), 通过密度比加权方法抽样学习策略, 限制策略只选择数据集中“好数据”, 而不是采样学习所有的数据. 这种采样策略被构建为一个即插即用的模块, 解决在数据集倾斜或不平衡的情况下, 离线强化学习难以学习高回报策略的问题. Xu 等人[24]提出稀疏 Q 学习 (sparse Q-learning, SQL) 和指数 Q 学习 (exponential Q-learning, EQL), 为更深入地理解样本内学习范式的工作原理, 将隐式价值正则化应用于策略. 当离线数据集的质量较低时, 稀疏性会过滤掉那些 Q 值低于阈值的不良行为, 从而提高算法的性能. Garg 等人[25]提出极端 Q 学习算法 (extreme Q-learning, XQL), 在最大熵设置中直接估计最优软值函数, 而不需要从策略中采样. 根据极值定理 (EVT)[26], 利用 Gumbel 分布解决建模 Q 函数估计误差的问题. 本文提出在 IQL 算法优异性能的基础上, 对 V 值函数进行约束, 提高了价值函数评估的准确性和算法的稳定性.

1.2 稀疏奖励任务

在蚂蚁迷宫环境中, 主要需要解决稀疏奖励的问题. 即在稀疏奖励的环境中, 智能体仅在任务完成或关键事件发生时给出奖励信号. 目前解决这类问题的强化学习方法有两类: 一是利用数据改进智能体的学习, 通过已有数据或者使用外部信息, 从而改变样本利用率和训练速度; 二是改进模型, 提升模型在大状态、大动作空间下处理复杂问题的能力. 利用数据提高智能体学习能力, 一般从奖励重塑[27]、课程学习[28]、好奇心驱动[29]等角度考虑. 而模型的改进, 比如分层强化学习[30], 利用多层次的结构来学习不同层次的策略, 解决了大状态、大动作空间下的复杂问题. 也有基于模型的方法, 通过使用一个预训练的模型来预测交叉口中其他实体的行为, 从而能更准确地预测将来的奖励和行为, 从而缓解稀疏奖励问题[31].

扩散模型具有长视野特性, 通过扩散, 结合样本邻域的信息, 生成贴近样本数据分布的数据. Wang 等人[32]提出扩散 Q 学习算法 (diffusion Q-learning, Diffusion QL), 引入扩散量子学习, 提出用扩散模型来执行策略正则化的方法. 在稀疏奖励的环境, 利用扩散模型的特点, 该算法捕捉多模态分布, 具有很强的分布匹配技术. Kang 等人[33]提出高效扩散策略算法 (efficient diffusion policies, EDP), 采用动作近似方法. 即从已损坏的动作中构建一个替代动作, 这个替代动作可以轻松地在数据集中生成. 这种方法使得在每个训练步骤中只需要通过噪声预测网络进行一次预测, 从而显著缩短了训练时间, 避免了繁琐的采样过程. Chen 等人[34]提出的从行为候选人中进行选择算法 (selecting from behavior candidates, SFBC), 采用了一种生成式的方法, 将学习到的策略解耦为两部分: 表达性生成式行为模型和行动评价模型. 这种解耦避免了学习具有封闭形式表达式的显式参数化策略模型, 进一步避免选择样本外的行动, 提高了计算的效率. 利用扩散的特点, 有效学习数据的分布, 从而解决稀疏奖赏的问题.

2 背景知识

2.1 马尔可夫决策过程

强化学习任务通常使用马尔可夫决策过程来描述, 定义为一个五元组 $M=(S,A,P,R,\gamma)$, 其中 S 是有限的状态集, A 是有限的动作集. $P:S\times A\times S\to[0,1]$ 是状态转移模型, $R:S\times A\to\mathbb{R}$ 是即时奖励函数, $\gamma\in[0,1]$ 为折扣因子. 在离线强化学习中, 智能体得到一个由行为策略 π_β 收集的静态数据集 $D=\{(s^{(i)},a^{(i)},s'^{(i)},r^{(i)})\}_{i=1}^{N}$. 假设 $d^{\pi}(s,a)$ 是策略 π 的贴现状态动作分布, 则有 $(s^{(i)},a^{(i)})\sim d^{\pi_\beta}(\cdot,\cdot)$, $s'^{(i)}\sim P(\cdot|s^{(i)},a^{(i)})$, $r^{(i)}=R(s^{(i)},a^{(i)})$. 离线 RL 的目标是找到一个策略 $\pi:A\times S\to[0,1]$ 希望得到最大化的期望累积奖励. 期望回报 $G_t=\sum_{i=t+1}^{T}r(s_i,a_i)$ 作为政策优劣的判别指标, 其中 $s_i\sim d^{\pi}(\cdot)$, $a_i\sim\pi(\cdot\mid s_i)$ 且 $s_i\sim P(\cdot\mid s_i,a_i)$. 马尔可夫决策过程中的期望回报有两种价值函数表示法. 定义状态值函数为在状态 s 处遵循策略 π 所得到的期望回报 $V_\pi(s)=\mathbb{E}_\pi(G_t\mid S_t=S)$, 动作值函数为在状态 s 处执行动作 a, 遵循策略 π 所得到的期望回报 $Q_\pi(s,a)=\mathbb{E}_\pi(G_t\mid S_t=S,A_t=A)$. 为了提高策略的学习率并减小方差, 将策略的优势函数定义为 $A_\pi(s,a)=Q_\pi(s,a)-V_\pi(s)$. 但无论使用哪种价值函数学习策略, 都会产生 OOD 行动. 因此有很多方法, 如策略约束、值函数正则化等来解决 OOD 问题.

2.2 期望回归

假设期望回归方程为多元线性方程, 模型表达式定义如下:

$$\mathbb{E}(y_i)=x_i^{\mathrm{T}}\beta+b_i \tag{1}$$

其中, 自变量 $x_i=(x_{i1},x_{i2},\ldots,x_{ip})$, β 为系数向量, $\beta=(\beta_1,\beta_2,\ldots,\beta_p)^{\mathrm{T}}$, b_i 为误差项. 通常假设 y_i 和误差项的同方差性为一个特定分布. 而期望值回归[35], 不仅可以模拟 y_i 的期望值, 还可以模拟 y_i 的整个分布. 可以同时考虑多个自变量与因变量之间的关系, 克服了二元线性因考虑不全面导致模型偏差的问题.

期望回归可以定义为:

$$m_\omega(y_i)=\arg\min_{m}\sum_{i=1}^{n}W_\omega(y_i)(y_i-m_\omega)^2 \tag{2}$$

其中, 权重定义为:

$$W_\omega(y_i)=\begin{cases}\omega, & \text{if } y_i>m_\omega\\ 1-\omega, & \text{if } y_i<m_\omega\end{cases} \tag{3}$$

其实, 期望值可以看作一个加权平均值, 其权重取决于 y_i, 拟合值和当前的不对称水平 $\omega\in(0,1)$, 其中 $\omega=0.5$ 的值为均方误差方法的结果.

2.3 策略学习

学习一个最优策略的目标, 是使智能体的期望折扣回报 $J(\pi)$ 最大化.

$$J(\pi)=\mathbb{E}_{s\sim d_\pi(s)}\mathbb{E}_{a\sim\pi(a|s)}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right] \tag{4}$$

其中, $d_\pi(s)$ 表示由策略 π 引起的非标准化折现状态分布, $\gamma\in(0,1)$ 是折扣因素, r 是奖赏函数.

$$d_\pi(s)=\sum_{t=0}^{\infty}\gamma^t p(s_t=s) \tag{5}$$

IQL 算法没有在策略评估器中添加显式的正则化以避免分布外的操作, 而是仅使用样本内的数据来学习最优的 Q 函数. IQL 算法使用非对称的 ℓ_2 损失来学习 V 函数, 这可以看作是对数据集支持的动作最大 Q 值的估计, 从而隐式 Q 学习可表示为:

$$\begin{cases}\min\limits_{V}\mathbb{E}_{(s,a)\sim\mathcal{D}}[|\tau-\mathcal{F}((Q(s,a)-V(s))<0)|(Q(s,a)-V(s))^2]\\ \min\limits_{Q}\mathbb{E}_{(s,a,s')\sim\mathcal{D}}[(r(s,a)+\gamma V(s')-Q(s,a))^2]\end{cases} \tag{6}$$

其中, $\mathcal{F}$ 是指示器函数. 策略学习借助优势函数近似最优算法, 在每次迭代中解决监督回归问题:

$$\pi_{k+1}=\mathbb{E}_{(s,a)\sim D}[\log\pi_k(a\mid s)\exp(\beta(Q_\theta(s,a)-V_\psi(s)))] \tag{7}$$

其中, $\beta\in(0,\infty)$ 是一个逆温度系数, 用来调节行为克隆和策略学习的比重. 策略的更新借助优势函数 $A(s,a)=Q(s,a)-V(s)$, 提高策略发现更大优势动作的概率, 其中状态 s 和动作 a 都是来自静态数据集 $\mathcal{D}$.

3 一种扩散模型期望最大化的离线强化学习方法

扩散模型期望最大化的离线强化学习方法 (DMEM) 结构示意图, 如图 1 所示. 将采样的一组数据传给 Q 和 V 网络, 用于评估策略网络的效果. 同时扩散模型的结果 a_0, 对值函数的评估有一定的惩罚, 使得评估效果更加精准. 另一方面也会将采样的状态作为扩散模型的输入, 在经过扩散模型的加噪和去噪的过程后, 输出 a_0. 其结合值函数的评估效果, 计算策略网络的损失函数, 同时使用最小化损失函数的方式, 使用策略梯度算法更新策略网络, 即更新扩散模型中去噪的网络参数. 同时也使用最小化损失函数更新值函数网络即评论家网络.

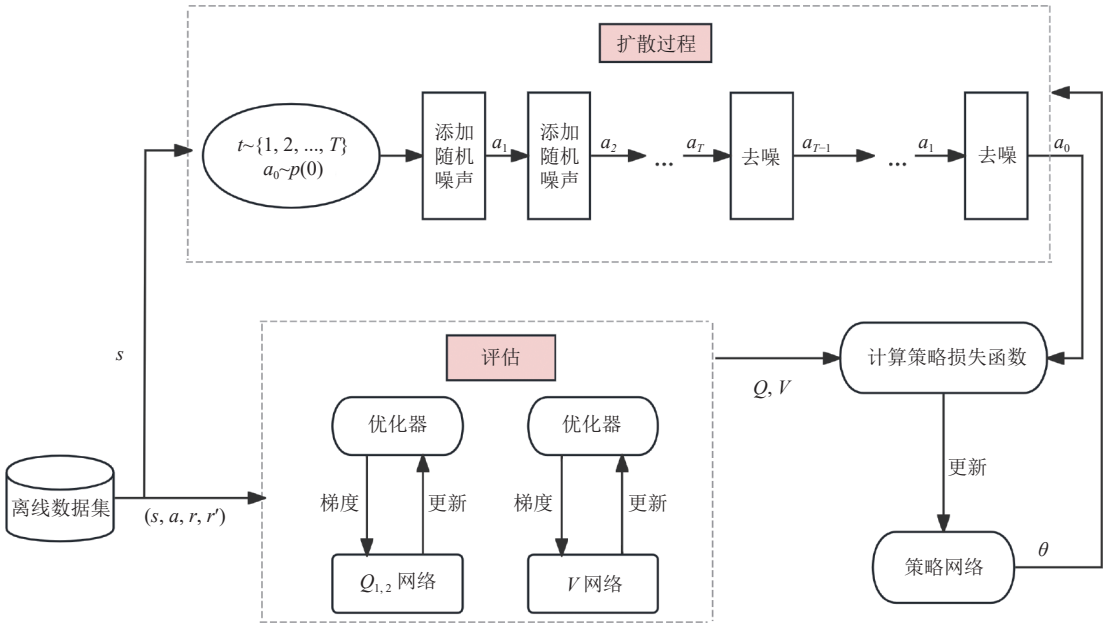


图 1 DMEM 结构

3.1 值函数的设定

本文利用期望回归模型, 允许异方差性, 丰富了模型的多样性. 由于期望回归模型使用最小二乘框架, 因此比分数回归模型更容易包含具有二次惩罚的平滑效应和复杂协变量结构. 对比 IQL 算法, 估计一个二维随机变量的状态条件期望. 本文利用期望回归模型, 估计一个高维状态动作空间的期望, 且引入一个惩戒项对平滑分量进行适

当的二次惩罚. 在 IQL 算法更新值函数的优势下, 添加一个惩戒项, 平滑地估计响应函数, 使得其更贴近响应函数的最大值.

根据以上思想, V 值函数设定为:

$$V_\psi \to \mathbb{E}_{(s,a)\sim D}[L_2^\omega(Q_{\hat{\theta}}(s,a)-V_\psi(s)-\varsigma H(a\mid s))] \tag{8}$$

其中, $Q_{\hat{\theta}}(s,a)-V_\psi(s)$ 是价值函数之间的差异, $H(a\mid s)$ 是惩戒项. 在强化学习中, 动作值函数和状态值函数定义为:

$$Q(s,a)=\mathbb{E}[R_t\mid s_t=s,a_t=a] \tag{9}$$

$$V(s)=\mathbb{E}[R_t\mid s_t=s] \tag{10}$$

动作值函数, 是智能体在当前状态下, 选取动作 a, 取得直到终点的奖赏的期望. 状态值函数, 是智能体在当前状态下, 取得直到终点的奖赏的期望. 两个函数之间是可以相互转化, 且相互迭代, 可以得到公式 (11) 和公式 (12):

$$Q_k(s,a)=r+\gamma\sum p(s'\mid s,a)V_{k-1}(s') \tag{11}$$

$$V_k(s)=\sum\pi(a\mid s)\left[r+\gamma\sum p(s'\mid s,a)V_{k-1}(s')\right] \tag{12}$$

其中, $p(s'\mid s,a)$ 是状态转移函数. 修改 IQL 算法的 V 值函数更新公式中用到的 $Q_k(s,a)-V_k(s)$, 将 Q 看成 $|A|$ 个状态转移概率乘以确定性概率 1, 则可将优势函数扩展到高维状态动作空间中. 可以推导到公式 (13), 再利用最小二乘法求解.

$$\begin{aligned}Q_k(s,a)-V_k(s)&=\frac{1}{|A|}\left[\sum(r+\gamma\sum p(s'\mid s,a)V_{k-1}(s'))\right]-\sum\pi(a\mid s)\left[r+\gamma\sum p(s'\mid s,a)V_{k-1}(s')\right]\\&=r+\gamma\left[\frac{1}{|A|}\sum_{i=1}^{|A|}\sum p(s'\mid s,a)-\sum_{i=1}^{|A|}\pi(a_i\mid s)\sum p(s'\mid s,a_i)\right]V(s')\end{aligned} \tag{13}$$

其中, $|A|$ 是动作集数. 可将其中间复杂项化为:

$$|A|y_i-m_\omega(x) \tag{14}$$

其中, $m_\omega(x)=\eta_1x_1+\eta_2x_2+\ldots+\eta_{|A|}x_{|A|}$. 这意味着动作空间的维度影响算法的性能, 提出在 V 值函数更新公式中, 添加了一个有关动作的惩戒项 $\pi(a\mid s)$, 对 V 值函数进行适当的惩罚.

加一个有关动作的惩戒项不仅仅能促进对未来决策的合理预测, 还能缓解 Q 值函数过高估计的问题. 在当前状态 s 下, V 值函数是一个固定值. 当惩戒项 $\pi(a\mid s)$ 概率值较低时, 并不影响 V 值函数和 Q 值函数的差距; 但当惩戒项 $\pi(a\mid s)$ 概率值较高时, 若 Q 值函数过高估计, $Q(s,a)-\varsigma H(a\mid s)$ 则会降低与 V 值的偏差, 从而减小 V 值在该状态下上升的可能性, 提高价值函数评估的性能, 进而也提升了训练的效率和性能. 其中 ς 为温度系数, 对价值函数的影响可见第 4.3 节实验部分.

Q 值函数的更新是使用的均值方差 (MSE) 方法. 由于下一状态的评估是借助 V 值函数, 无需考虑下一动作是否取得最大 Q 值问题. 在某种意义上, 更新方式类似于 SARSA 算法评估, 更新公式如下:

$$L_\theta(Q)=\mathbb{E}_{(s,a,s')\sim D}[(r+\gamma V_\psi(s')-Q_{\hat{\theta}}(s,a))^2] \tag{15}$$

3.2 扩散模型

扩散模型指扩散输入数据点的邻域信息. 在扩散过程中, 通过当前数据点与周围数据点信息的结合, 从而增强和扩展了当前数据点的特征. 扩散模型训练的关键是去噪过程.

图 2 展示了扩散模型在蚂蚁迷宫场景下去噪的过程. 经过随机高斯噪声的加噪后, 获取了当前的状态周围的信息. 对于未知的数据分布, 扩散模型也能在去噪过程中, 增加当前状态选择最优动作的概率. 结合算法采取多步规划的方式, 扩大了扩散的范围. 同时利用值函数 SARSA 式更新的优势, 使得学习的值函数更接近最优性的值函数, 从而产生一个更好的策略.

扩散模型是一个生成模型, 从一个噪声中生成一个真实的数据. 首先给数据逐渐加噪声, 使得数据逐渐变成一个完全随机噪声的过程. 从一个未知数据样本 $q(x)$ 中, 随机采样一个变量 $x_0\sim q(x)$. 扩散模型的前向过程, 则是 T 次对数据 x_0 添加高斯噪声的过程, 且 t 时刻仅与 $t-1$ 时刻有关. 结合马尔可夫的性质, 转移分布 $q(x_t\mid x_{t-1})$ 满足:

$$q(x_t \mid x_{t-1}) = N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{16}$$

其中, β_t 是高斯分布的超参数. 引入一个随机变量 $\xi \sim N(0, I)$, 样本 t 时刻 x_t 和 $t-1$ 时刻 x_{t-1} 可以表示为:

$$x_t = \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\xi_{t-1} \tag{17}$$

其中, $\alpha_t = 1-\beta_t$. 由于独立高斯分布的可加性, x_t 可直接由 x_0 和 α 表示:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\xi \tag{18}$$

其中, $\bar{\alpha}_t = \alpha_t\alpha_{t-1}...\alpha_1$. 而超参数 β 是随着 T 时刻变大而递增的, α 则随着时间递减. $\sqrt{\alpha_t}$ 是为保证 x_T 最后收敛到方差为 1 的标准高斯分布.

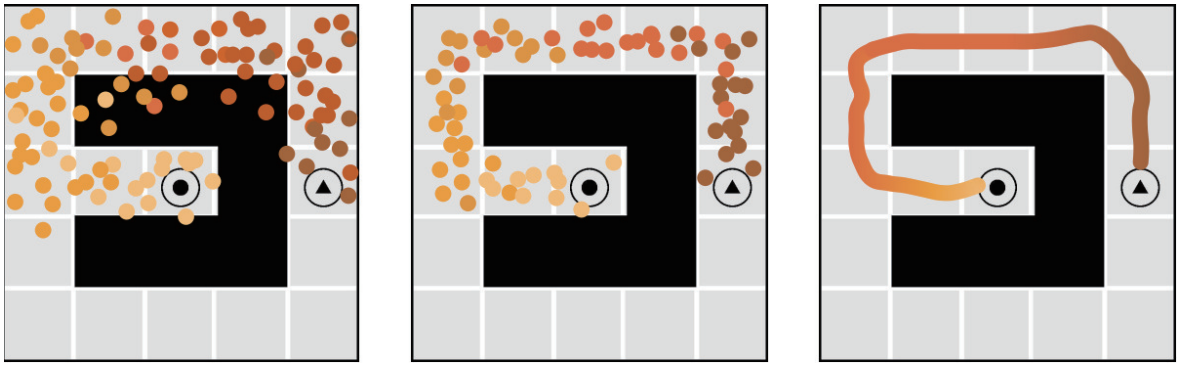


图 2 Maze 环境下轨迹去噪过程

加噪声是为了配置一个数据, 而扩散模型的关键是, 学习从中去掉噪声. 由于 $q(x_{t-1}x_t)$ 的数据分布是未知的, 只能借助一个已知的线性模型 $p_\phi(x_{t-1} \mid x_t)$ 去近似 $q(x_{t-1}x_t)$. 可定义为:

$$p_\phi(x_{t-1} \mid x_t) = N(x_{t-1}; \mu_\phi(x_t, t), \beta_t) \tag{19}$$

这里方差直接是 β_t, 是因为方差不需要网络去估计. 在理论上计算均值和方差需要用到贝叶斯公式:

$$P(AB) = P(A)P(B \mid A) \tag{20}$$

则在已知 x_0 和 x_t 的情况下, 可推导出:

$$q(x_{t-1} \mid x_t, x_0) = q(x_t \mid x_{t-1}, x_0)\frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)} \tag{21}$$

将正态分布概率密度函数定义:

$$\rho(x) = \frac{1}{\sqrt{2\pi\sigma}}\mathrm{e}^{-0.5(\frac{x-\mu}{\sigma})^2} \tag{22}$$

带入公式 (21), 得到如下公式:

$$q(x_{t-1} \mid x_t, x_0) = \exp\left(-0.5\left(\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)x_{t-1}^2 - \left(\frac{2\sqrt{\alpha_t}}{\beta}x_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0\right)x_{t-1} + C\right)\right) = \exp(-0.5(ax^2+bx+c)) \tag{23}$$

其中, $a = \dfrac{\alpha_t}{\beta_t} + \dfrac{1}{1-\bar{\alpha}_{t-1}}$, $b = -\left(\dfrac{2\sqrt{\alpha_t}}{\beta}x_t + \dfrac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}x_0\right)$. 则方差 $\tilde{\beta}_t = \dfrac{1}{a}$ 可转化为:

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \tag{24}$$

均值 $\tilde{\mu}_t(x_t, x_0) = -\dfrac{b}{2a}$ 可转化为:

$$\begin{aligned}\tilde{\mu}_t(x_t, x_0) &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_t)}\left(x_t - \sqrt{1-\bar{\alpha}_{t-1}}\xi_t\right) \\ &= \frac{1}{\sqrt{\bar{\alpha}_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\xi_t\right)\end{aligned} \tag{25}$$

在训练时, 使用极大似然估计, 求模型的参数. 在计算损失时, 使用 KL 散度或者 MSE 方法等, 计算分布 p 和分布 q 之间的差异. 假设分布 p 和 q 都服从高斯分布, 且方差定为常数. 那么只需要优化二者均值之间的差, 去噪网络参数更新可表示为:

$$\begin{aligned}\mathcal{L}_d(\phi)&=\mathbb{E}_{\xi\sim N(0,I),x\sim N(0,I),s\sim\mathcal{D}}\left[\|\xi-\xi_\phi(x_t,t)\|^2\right]\\&=\mathbb{E}_{\xi\sim N(0,I),x_0\sim p(x),s\sim\mathcal{D}}\left[\|\xi-\xi_\phi(\sqrt{\bar{\alpha}_t}x_0+\sqrt{1-\bar{\alpha}_t}\xi),s,t)\|^2\right]\end{aligned}\tag{26}$$

通过梯度下降方式最小化损失函数.

扩散模型从一个潜在变量中, 生成一个新的样本, 其分布与先验数据分布相同. 面对稀疏奖励不能立刻给出反馈, 从而导致局部最优问题. 扩散模型通过学习噪声, 结合数据点邻域的信息, 更准确地描述了数据分布, 提高数据的质量. 扩大扩散范围和增大扩散模型时间步, 能更加准确的学习去噪过程, 从而提高了策略网络的精准性和稳定性.

3.3 DMEM 算法

在 DMEM 算法中, 评估的策略利用了状态值函数和动作值函数. 其中, $Q(s,a)$ 价值函数代表回报的均值, 更新方式如公式 (6), 其中涉及的元素, 动作和状态都是采样于静态数据集. 因此无需考虑数据集外的 OOD 动作, 避免对此的值函数高估问题. 利用 SARSA 方法更新评估网络, 而不是直接选择最大动作 Q 值, 给予策略更多泛化的能力.

而状态值函数, 在 IQL 算法的 V 值更新方式的基础上, 针对高维度问题, 进行了如下修改. 从第 3.1 节推导过程中, 得到 $\pi(a\mid s)$ 是 $V(s)$ 影响贴近 $Q(s,a)$ 的一个重要因素. 对此, 添加一个有关策略选择动作的惩戒项. 同时这个策略引入扩散模型, 而扩散模型中的关键是去噪的参数. 那么值函数 $V(s)$ 的更新也考虑了去噪参数带来的影响. 得到新的值函数 $V(s)$ 的更新公式为:

$$L_V(\psi)=\mathbb{E}_{(s,a)\sim\mathcal{D}}\omega_\tau(Q(s,a),V(s))[Q(s,a)-V(s)-\varsigma\log(\pi(s))]^2\tag{27}$$

其中, ς 是一个常数, 控制动作在值函数更新中的影响. 通过 MSE 方法来计算损失函数, 衡量预测值 $V(s)$ 和实际值 $Q(s,a)$ 之间的误差. 结合期望回归的思想, 重新考虑损失对值函数更新的影响. 然后通过梯度下降的方式进行优化.

在 DMEM 算法中, 由 ϕ 参数化策略网络 $\pi(a\mid s;\phi)$, 在基于策略的强化学习模型下, 引入一个约束项, 用于惩罚策略去噪训练过程的目标. 定义 DMEM 策略学习的目标函数为:

$$J(\pi)=\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\exp(\delta_1(Q(s,a)-V(s))\right]\log\pi(a\mid s)+\delta_2\mathcal{L}_d(\phi)\tag{28}$$

其中, $Q(s,a)$ 表示动作值函数, $V(s)$ 表示状态值函数. DMEM 策略是基于策略学习的, 直接评判策略学习的好坏. 通过期望最大化来学习策略, $Q(s,a)-V(s)$ 为优势函数, 作为期望最大化的一种形式. δ_1 为控制优势函数的温度系数. $\exp(\delta_1(Q(s,a)-V(s))$ 可以看作一个权重, 是策略 π_{k+1} 学习 π_k 的权重, 用来衡量策略的好坏. $\mathcal{L}_d(\phi)$ 是一个惩戒项, 用来衡量噪声参数学习的偏差. δ_2 为惩戒项的温度参数, 控制策略学习与噪声误差之间的权重.

考虑简单的一步 MDP 问题, 用似然比计算策略梯度为:

$$\nabla_\phi J(\phi)=\sum_{s\in S}d(s)\sum\pi_\phi(a\mid s)\nabla_\phi\log\pi_\phi(a\mid s)r\tag{29}$$

其中, r 是一步时间后获得的即时奖赏. 推广到多步 MDP, 只需将即时奖赏 r, 换成长期的期望值, 可以用价值函数 $Q(s,a)$ 兼容近似表示. 但存在梯度方差的问题, 采用基线减少方差的思想, 即平均 Q 值减去均值. 增加比均值更好的动作的概率, 则策略梯度为:

$$\nabla_\phi J(\phi)=\sum_{s\in S}d(s)\sum\pi_\phi(a\mid s)\nabla_\phi\log\pi_\phi(a\mid s)[Q(s,a)-V(s)]\tag{30}$$

在稀疏奖励的环境下, IQL 算法基于策略梯度更新策略网络, 且利用 SARSA 方法更新评价网络, 而不是直接选择最大动作 Q 值. 本文还考虑利用扩散模型的优势, 学习策略网络. 即在不能得到立即奖赏的情况下, 也能通过邻域的信息, 增强数据的分布. 扩散模型的推导过程, 如第 3.2 节所述. 扩散模型的关键技术是去噪过程, 所以学习

去噪参数 ϕ 是扩散模型的重点. 则 DMEM 算法策略有关噪声的惩戒项可表示为:

$$\mathcal{L}_d(\phi)=\mathbb{E}_{\xi\sim N(0,I),(s,a_0)\sim\mathcal{D}}\left[\|\xi-\xi_\phi(\sqrt{\bar{\alpha}_t}a_0+\sqrt{1-\bar{\alpha}_t}\xi),s,t)\|^2\right] \tag{31}$$

其中, ξ 是扩散前向过程的加噪参数. $\xi\sim N(0,I)$ 意味着服从一个简单的高斯分布. ξ_ϕ 是去噪参数, 将加噪后的数据信息逐步处理. 而 $\sqrt{\bar{\alpha}_t}a_0+\sqrt{1-\bar{\alpha}_t}\xi$ 就是 t 时刻加噪后的样本数据. 利用均方误差计算损失函数, 计算出网络预测的去噪参数与实际值的差距.

ϕ 作为去噪的参数, 也是 DMEM 算法中策略的网络参数. 在扩散模型的逆向过程中, ξ_ϕ 是扩散模型学习预测出的噪声. 而参数 ϕ 在策略 $\pi_\phi(a\mid s)$ 中则用来产生动作概率, 两者输出的结果不同, 代表的含义也不同. 但两者之间可以相互转换, 其转换的公式为:

$$a_0=\mu+\mathrm{e}^{0.5\sigma^2}\xi_\phi \tag{32}$$

其中, 均值 μ 见公式 (25), 方差 σ 见公式 (24).

则 DMEM 算法策略的最终目标函数为:

$$L_\pi(\phi)=\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\exp(\delta_1(Q(s,a)-V(s))\right]\log\pi(a\mid s)+\delta_2\mathbb{E}_{\xi\sim N(0,I),(s,a_0)\sim\mathcal{D}}\left[\|\xi-\xi_\phi(\sqrt{\bar{\alpha}_t}a_0+\sqrt{1-\bar{\alpha}_t}\xi),s,t)\|^2\right] \tag{33}$$

其中, $\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[\exp(\delta_1(Q(s,a)-V(s))\right]\log\pi(a|s)$ 通过期望最大化来学习策略, 利用价值函数衡量策略的好坏. $\mathbb{E}_{\xi\sim N(0,I),(s,a_0)\sim\mathcal{D}}\left[\|\xi-\xi_\phi(\sqrt{\bar{\alpha}_t}a_0+\sqrt{1-\bar{\alpha}_t}\xi),s,t)\|^2\right]$ 则通过均方误差来降低噪声学习的误差. 策略的目标函数通过随机梯度下降的方式更新学习.

使用具有平滑系数 τ 的指数移动加权平均, 来软更新目标 Q 网络:

$$\bar{\theta}_i\leftarrow\tau\theta_i+(1-\tau)\bar{\theta}_i \tag{34}$$

其中, τ 的大小会影响训练的稳定性, τ 太小更新缓慢, 导致训练速度大大降低.

算法 1. 扩散模型的采样过程.

输入: 状态 s;
输出: 动作 a_0.

1. 初始化: 去噪网络 ξ_ϕ, 时间步 n_t, 超参数 $\{\beta_t\in(0,1)\}_{t=1}^{n_t}$, 静态数据集 $\mathcal{D}$
2. 前向过程 (添加噪声):
3. $a_0\sim q(a)$, $q(a)\sim\mathcal{D}$
4. $\alpha_t=1-\beta_t$
5. **for** $t=1,2,\ldots,n_t$ **do**
6. $\quad a_t\sim q(a_t\mid a_0)=N(a_t;\sqrt{\bar{\alpha}_t}a_0,(1-\bar{\alpha}_t)I)$
7. **end for**
8. 逆向过程 (去噪):
9. **for** $t=n_t,n_t-1,\ldots,1$ **do**
10. $\quad \xi_t\sim\xi_\phi(a_t,t,s)$
11. $\quad \mu_t=1/\sqrt{\alpha_t}\cdot(a_t-\beta_t/\sqrt{1-\bar{\alpha}_t}\xi_t)$
12. $\quad \sigma_t=\sqrt{1-\bar{\alpha}_{t-1}/1-\bar{\alpha}_t\cdot\beta_t}$
13. $\quad a_{t-1}=\mu_t+\mathrm{e}^{0.5\sigma^2}\xi_t$
14. **end for**

在扩散模型中, 输入状态信息. 算法 1 第 1 行, 初始化去噪的网络参数, 同时需要初始化扩散的时间序列长度、全局经验池以及高斯分布的方差. 算法 1 第 2–7 行, 给动作添加随机高斯噪声. 从与数据集同分布的数据中采样初始动作, 根据第 6 行的公式, 累积添加 T 次噪声. 算法 1 第 8–14 行, 将加噪后的动作输入去噪网络 ξ_ϕ 中, 输出当前

t 时刻预测的噪声. 通过预测的噪声, 计算出预测数据分布的方差和均值. 然后根据均值和方差, 得出去噪后的动作, 也就是 $t-1$ 时刻的动作. 重复 T 次, 最后输出动作 a_0.

算法 2. DMEM 算法.

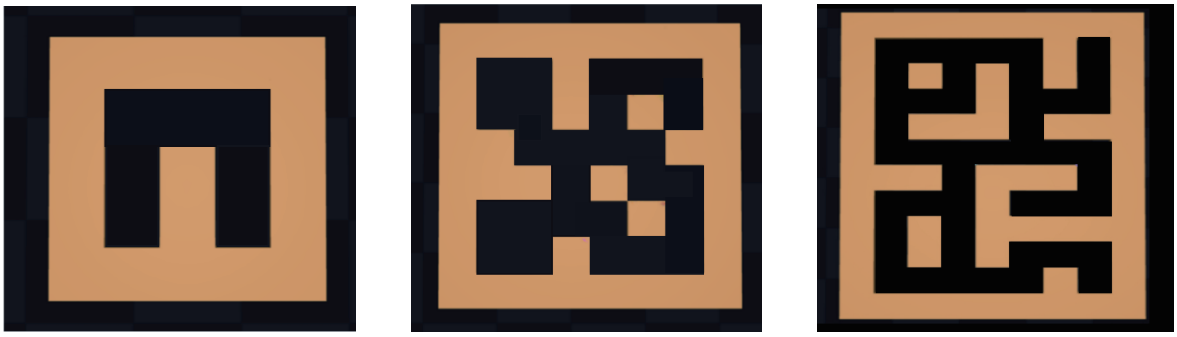
输入: 离线数据集 $\mathcal{D}$, 采样数量 N, 平滑系数 τ;
1. 初始化: 策略网络 $\pi(\cdot\,|\,\cdot;\phi)$, 评论 V 网络 V_ψ, 评论 Q 网络 Q_{θ_1} 和 Q_{θ_2}, 目标评论 Q 网络 $Q(\cdot\,|\,\cdot;\theta_1')$ and $Q(\cdot\,|\,\cdot;\theta_2')$
2. **for** 每步迭代 **do**
3. 从数据集 $\mathcal{D}$ 采样 N 对 (s,a,r,s') 样本
4. 从策略 $\pi(\cdot\,|\,s)$ 中采样动作 a_0
5. 更新 V 网络, 见公式 (27)
6. 更新 Q 网络, 见公式 (6)
7. **for** 每步迭代 **do**
8. 更新策略网络, 见公式 (33)
9. **end for**
10. 更新目标网络, 见公式 (34)
11. **end for**

在 DMEM 中, 输入全局经验池、随机小批量采样样本个数以及平滑系数. 第 1 行初始化策略网络、评估网络和目标网络. 算法 2 第 3, 4 行, 随机采样 N 组样本, 在 ϕ 参数化下的策略选择动作. 第 5, 6 行, 是评估网络参数的更新. Q 函数网络参数的更新不涉及数据集外的动作, 但 V 值网络参数的更新, 涉及策略选择的动作. 见算法 2 第 8 行, 通过梯度下降的方式更新策略, 策略参数的更新基于策略梯度方法. 值函数作为策略更新的权重, 同时引入扩散模型预测噪声的误差. 最后软更新目标网络的参数, 见算法 2 第 10 行.

4 实验分析

4.1 实验环境

D4RL[36]数据集种类丰富, 包括 Gym-Mujoco、AntMaze、Adroit 等. 其中 MuJoCo 是经典连续控制任务的平台. AntMaze 环境主要是面向稀疏奖励、无向和多任务数据问题的平台. 且数据是通过随机选择目标位置, 然后使用计划器生成航点序列来生成的. AntMaze 是一个导航任务, 其目标是训练一个四足智能体, 试图找到起点与目标点的最短路径. 根据路径的复杂度可以分为: umaze、medium、large 这 3 种, 如图 3 所示.



(a) Antmaze-umaze-diverse (b) Antmaze-medium-diverse (c) Antmaze-large-diverse

图 3 蚂蚁迷宫环境

在路径长度确定的情况下, 根据重置位置和目标位置的随机性, 可分为 3 类: 固定的重置位置和目标位置、play、diverse. 其中 Play 是指蚂蚁的重置位置固定, 但是选择的目标随机. Diverse 是指蚂蚁选择的重置位置和

目标位置都随机选择. 则蚂蚁迷宫有 6 个子环境. 其行动空间维度为 8, 意味着铰链关节有 8 个, 而动作代表在关节处的转矩, 大小在−1 到 1 之间. 在不考虑外部接触力的情况下, 空间维度是 27. 智能体使用不同部位铰链, 采取不同的动作. 在空间维度上相比于 Maze2D 多了 8 个自由度, 将二维平面空间上升到三维, 使得任务更加复杂.

4.2 实验设置

所有算法的每个任务, 都选取了 5 个不同的随机种子, 且都独立运行. 每个任务每次实验都要运行 100 万步, 每 5 万步评估 100 步作为估计值. 实验图中的实线是算法 5 次独立训练的平均性能结果. 实线周围的阴影则是训练的误差范围. 阴影的范围则代表算法训练的稳定性, 范围越大稳定性越差.

DMEM 算法包含扩散策略网络和评论家网络. 扩散策略中包含加噪过程和去噪过程, 其中加噪过程不涉及网络参数. 去噪过程中有 4 层线性神经网络, 均采用 Mish 函数作为激活函数. 其中需要将时间编码到神经网络中, 使用正弦时间戳嵌入块 SinusoidalPosEmb 函数保留当前时间信息, 再经过两层线性神经网络, 一次 Mish 函数激活. 评论家网络包含 3 层隐藏的线性神经网络, 均采用 ReLU 作为激活函数. 策略网络和评论家网络都使用 Adam 作为优化器, 以梯度下降的方式更新网络参数. DMEM 算法的其他超参数设置, 如表 1 所示.

表 1 DMEM 超参数设置

超参数	取值	参数描述	超参数	取值	参数描述
$\mathcal{D}$	5 000	经验池大小	κ_s	1E–4	时间步截断参数下线
δ_1	10	价值函数权重	κ_e	2E–2	时间步截断参数上线
δ_2	10	去噪误差权重	l	3E–4	学习率
n_t	5	扩散模型时间步个数	τ	5E–3	目标平滑系数
ς	0.2	更新V网络的动作权重	ω	0.7	期望回归权重

4.3 实验结果与分析

将本文所提出的 DMEM 算法, 与其他离线强化学习算法 CQL 和 IQL 进行性能对比, 同时与基于扩散策略的 Diffusion-QL 算法进行性能对比. 其学习曲线如图 4 所示. 在任务 Antmaze-medium-diverse 和 Antmaze-large 的两个环境中, DMEM 算法的表现明显优于 CQL、IQL 和 Diffusion-QL. 在其他任务中, 性能提高效果虽不甚明显, 但仍优于其他算法. 在大多任务中, DMEM 算法的阴影部分都比 IQL 算法的面积大, 究其原因是扩散模型选择的时间步偏小. 尽管影响了策略的去噪效果, 但节省了训练的时间成本.

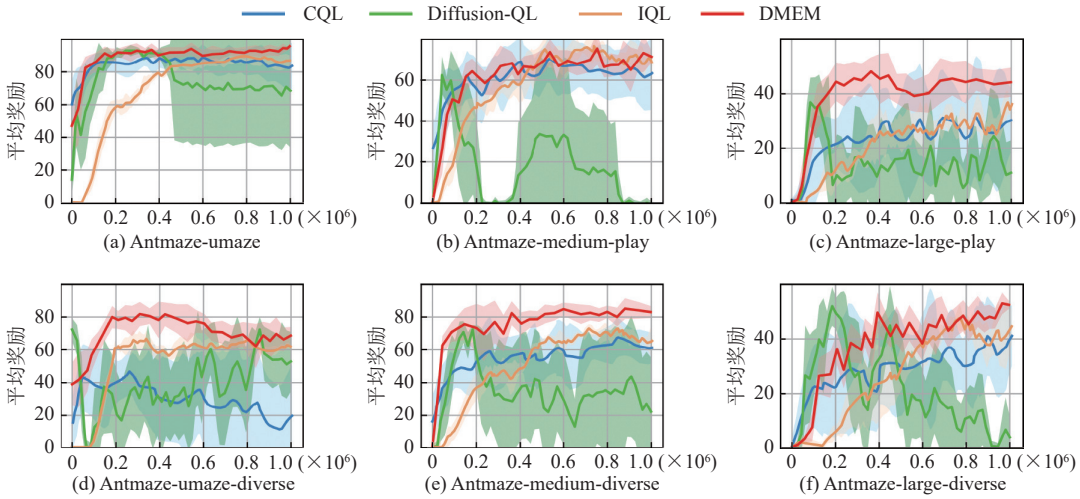


图 4 DMEM 和其他离线强化学习算法学习曲线图

表 2 给出了 4 个算法在 6 个任务中所获得的累计回报的均值和标准偏差. 其结果是在训练 5 个种子的基础上记录的. 其中均值和标准偏差都经过了归一化处理. 以均值作为最终评判标准, 实验图是根据均值和偏差共同绘制. 表中加粗了任务中表现优异的实验数据. 从平均值的角度, 观察到 DEME 方法在 6 个任务中的性能, 均优于其他算法. 但从标准偏差的角度观察, 在大部分任务中, DMEM 方法的训练过程的稳定性较好. 在 Antmaze-umaze-diverse 和 Antmaze-large-play 中 DMEM 稳定性比 IQL 算法差. 究其原因是: 在稀疏奖励的情况下, 为了提高 DMEM 算法的泛化能力, 而牺牲了训练过程中的一部分稳定性.

表 2 DMEM 与 CQL, Diffusion-QL, IQL 离线强化学习算法最终性能对比

任务	CQL		Diffusion-QL		IQL		DMEM	
	平均值	标准偏差	平均值	标准偏差	平均值	标准偏差	平均值	标准偏差
Antmaze-umaze	86.651	14.342	68.454	34.62	86.842	3.145	**95.705**	1.660
Antmaze-umaze-diverse	68.454	34.620	52.827	18.112	63.092	1.165	**68.729**	5.254
Antmaze-medium-play	63.314	17.484	–0.73	1.476	68.628	3.861	**71.36**	2.923
Antmaze-medium-diverse	**86.842**	3.145	22.212	33.041	65.502	10.724	82.97	3.689
Antmaze-large-play	30.284	11.591	10.977	11.918	36.217	1.996	**44.184**	5.361
Antmaze-large-diverse	41.137	11.063	3.813	7.446	44.61	6.662	**52.385**	1.447

在 Antmaze 任务的 6 个环境上, 根据以上实验结果, 验证了 DMEM 算法优于传统的离线强化学习算法. 下面进一步研究该方法对重要超参数的敏感性. 价值函数是基于梯度算法更新的权重, 类似起到重要参数的作用, 其学习的方法影响策略网络的学习. 其学习的过程, 类似于调参的过程, 在很多算法中, 重要参数对算法的性能和稳定性都有很大的影响. 在价值函数更新中添加一个惩戒项, 提高 V 值网络的学习效率, 增加 Q 值网络的约束, 从而有效提高算法的性能.

实验对比如图 5 所示, 当 $\varsigma = 0.2$ 时, DMEM 的性能表现最为优异. 相比较而言, ς 较小时, 取 $\varsigma = 0.02$, 智能体在训练价值函数的过程中, 由于高维度空间的问题, 使得估计的偏差较大, 学习的 V 值函数与实际的 Q 值函数之间的差偏大. 导致在策略更新时, 高维度问题仍会影响策略学习的方向, 降低了学习的效率. 对于 ς 较大的情况, 取 $\varsigma = 0.5$ 和 $\varsigma = 0.995$ 时, DMEM 在蚂蚁迷宫的环境中, 价值评估函数几乎没有作用, 严重影响了 DEME 算法的性能. 因此在 $\varsigma = 0.2$ 的情况下, 发现曲线的误差区间显著变窄, 有效验证了适当的 ς 对算法的影响.

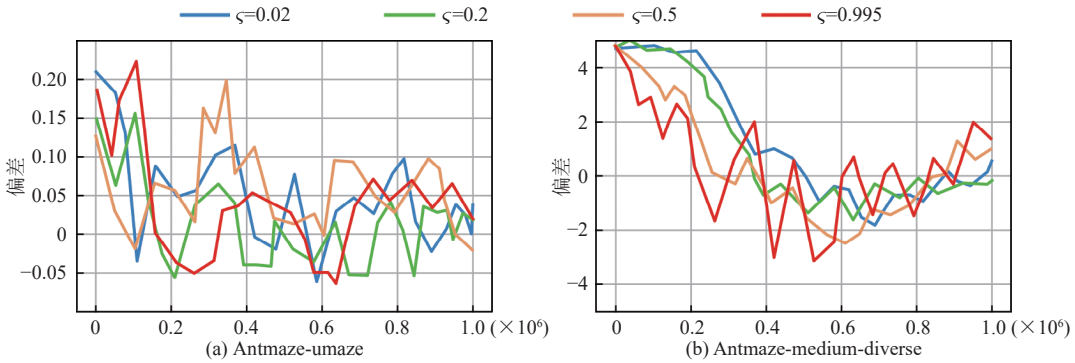


图 5 不同动作权重系数的 DMEM 学习曲线图

根据表 3 实验结果, 可验证 DMEM 方法在不同的 ς 上, 性能的差异. ς 过低, 没有惩戒作用, 两个价值网络间存在一定的差异, 高维度的动作空间依旧影响价值函数的学习. ς 过高, 过分强调动作维度对价值网络的作用, 导致价值网络无法得到有效学习, 失去了评估的作用, 从而影响策略网络的更新. 由此可见, 适当的 ς 选择对算法的性能起着重要的作用.

为了验证扩散模型对算法性能的影响, 选择了 Antmaze 的 3 个环境进行比较实验. 对比 DMEM 算法引入扩散模型和未使用扩散模型组件算法的性能. 实验结果如图 6 所示, 以平均奖励为指标, DMEM 算法的结果明显优

于未引入扩散模型的算法, 可见扩散模型有效地提高了算法的性能. 在训练的 20 万步前, DMEM 算法性能提升明显快于未引入扩散模型的算法, 由此可见扩散模型还能提高算法的收敛性, 加快模型的学习效率.

表 3　不同动作权重系数的 DMEM 算法最终性能

任务	$\varsigma = 0.02$	$\varsigma = 0.2$	$\varsigma = 0.5$	$\varsigma = 0.995$
Antmaze-umaze	90.870	95.705	86.149	50.572
Antmaze-umaze-diverse	65.624	68.729	60.527	34.173
Antmaze-large-diverse	48.447	52.385	43.711	19.481

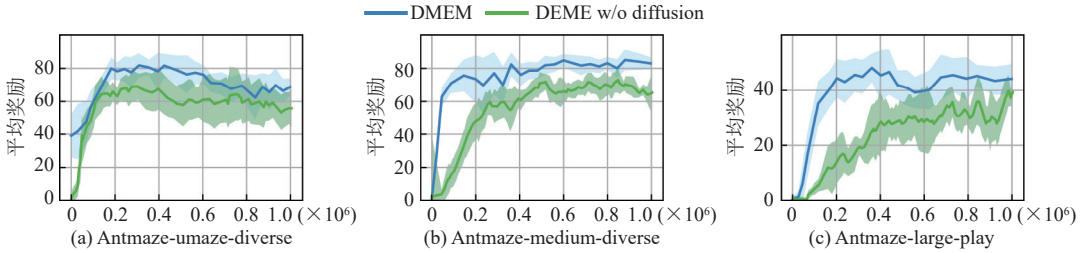


图 6　DMEM 和没有扩散模型的 DMEM 的算法对比学习曲线图

为了验证在 V 值网络更新中添加动作惩戒项对算法性能的影响, 选择了 Antmaze 的 3 个环境进行比较实验. 对比 DMEM 算法和未引入惩戒项组件的 DMEM 算法的性能. 实验结果如图 7 所示, 以平均奖励为指标, 相比于未引入惩戒项的 DMEM 算法, DMEM 算法结果更优, 可见惩戒项提高了策略的学习效果. 在消融实验环境中, DMEM 算法的阴影面积比 DMEM 未引入惩戒项的小, 由此可知动作惩戒项还能提高算法的稳定性.

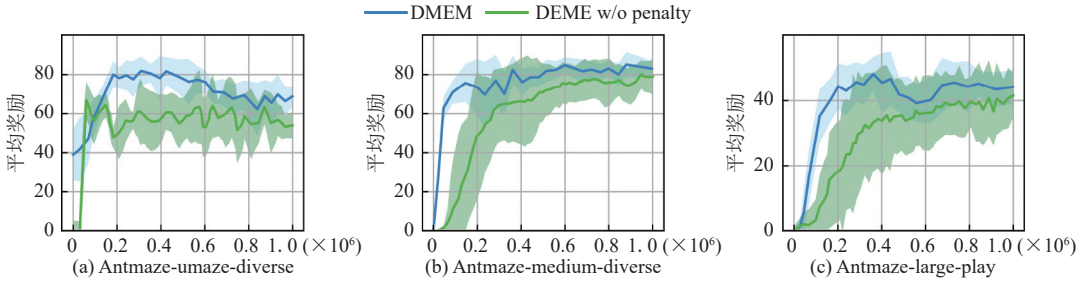


图 7　DMEM 和没有动作惩戒项的 DMEM 的算法对比学习曲线图

5　总　结

本文提出了一种扩散模型期望最大化的离线强化学习算法 DMEM. 通过去噪参数, 结合数据邻域信息, 生成数据样本, 增强了策略的泛化能力和选择的多样性. 同时使用期望回归和 SARSA 方法更新价值函数, 期望回归将二维价值拓展到高维空间, 有效减少了预测的误差, 提高了策略的学习效率. 实验选取蚂蚁迷宫环境中 6 个经典的稀疏奖励任务, 验证了算法的性能. 实验结果表明, 本文所提出的算法具有一定的优势. 在复杂离散的稀疏奖励环境下, 期望最大化方法和扩散模型使得泛化性能大幅提高.

从算法的性能和成本角度考虑, 未来的研究方向可能是利用轨迹来实现扩散和分层强化学习. 这意味着通过分层设计适当的子目标, 以减少扩散策略学习的次数. 虽然这可能会增加单次扩散加噪和去噪的计算成本, 但整体减少扩散策略学习的次数, 从而节约了计算成本. 从性能角度来看, 通过轨迹扩散学习, 可以加强当前节点和未来节点之间的联系. 在狭窄数据集的机械臂环境中, 延长扩散的视野距离, 增强未来结果对当前决策的影响, 从而提高智能体的学习和预测能力.

References:

[1] Sutton RS, Barto AG. Reinforcement Learning: An Introduction. 2nd ed., Cambridge: The MIT Press, 2018.

[2] Liu Q, Zhai JW, Zhang ZC, Zhong S, Zhou Q, Zhang P, Xu J. A survey on deep reinforcement learning. Chinese Journal of Computers, 2018, 41(1): 1–27 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2018.00001]

[3] Liu JW, Liu Y, Luo XL. Research and development on deep learning. Application Research of Computers, 2014, 31(7): 1921–1930, 1942 (in Chinese with English abstract). [doi: 10.3969/j.issn.1001-3695.2014.07.001]

[4] Levine S, Kumar A, Tucker G, Fu J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv:2005.01643, 2020.

[5] Peng ZY, Han CL, Liu YD, Zhou ZT. Weighted policy constraints for offline reinforcement learning. Proc. of the AAAI Conf. on Artificial Intelligence, 2023, 37(8): 9435–9443. [doi: 10.1609/AAAI.V37I8.26130]

[6] Mao YX, Zhang HC, Chen C, Xu Y, Ji XY. Supported value regularization for offline reinforcement learning. In: Proc. of the 37th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2024. 40587–40609.

[7] Zhang BL, Liu ZR. Adaptive uncertainty quantification for model-based offline reinforcement learning. Journal of Nanjing University of Posts and Telecommunications (Natural Science Edition), 2024, 44(4): 98–104 (in Chinese with English abstract). [doi: 10.14132/j.cnki.1673-5439.2024.04.009]

[8] Moerland TM, Broekens J, Plaat A, Jonker CM. Model-based reinforcement learning: A survey. Foundations and Trends® in Machine Learning, 2023, 16(1): 1–67. [doi: 10.1561/2200000086]

[9] Fujimoto S, Meger D, Precup D. Off-policy deep reinforcement learning without exploration. In: Proc. of the 36th Int'l Conf. on Machine Learning. Long Beach: PMLR, 2019. 2052–2062.

[10] Fujimoto S, Van Hoof H, Meger D. Addressing function approximation error in actor-critic methods. In: Proc. of the 35th Int'l Conf. on Machine Learning. Stockholm: PMLR, 2018. 1587–1596.

[11] Kumar A, Zhou A, Tucker G, Levine S. Conservative Q-learning for offline reinforcement learning. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 1179–1191.

[12] Yu TH, Thomas G, Yu LT, Ermon S, Zou J, Levine S, Finn C, Ma TY. MOPO: Model-based offline policy optimization. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 14129–14142.

[13] Kostrikov I, Nair A, Levine S. Offline reinforcement learning with implicit Q-learning. arXiv:2110.06169, 2021.

[14] Laskin M, Lee K, Stooke A, Pinto L, Abbeel P, Srinivas A. Reinforcement learning with augmented data. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 19884–19895.

[15] Zhu ZD, Lin KX, Jain AK, Zhou JY. Transfer learning in deep reinforcement learning: A survey. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2023, 45(11): 13344–13362. [doi: 10.1109/TPAMI.2023.3292075]

[16] Bhardwaj M, Xie TY, Boots B, Jiang N, Cheng CA. Adversarial model for offline reinforcement learning. In: Proc. of the 37th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2024. 1245–1269.

[17] Wang SY, Li XD, Qu H, Chen WY. State augmentation via self-supervision in offline multiagent reinforcement learning. IEEE Trans. on Cognitive and Developmental Systems, 2024, 16(3): 1051–1062. [doi: 10.1109/TCDS.2023.3326297]

[18] Qiao WD, Yang R. Soft Adversarial offline reinforcement learning via reducing the attack strength for generalization. In: Proc. of the 16th Int'l Conf. on Machine Learning and Computing. Shenzhen: ACM, 2024. 498–505. [doi: 10.1145/3651671.3651762]

[19] Rengarajan D, Vaidya G, Sarvesh A, Kalathil D, Shakkottai S. Reinforcement learning with sparse rewards using guidance from offline demonstration. arXiv:2202.04628, 2022.

[20] Liu SF, Sun SL. Safe offline reinforcement learning through hierarchical policies. In: Proc. of the 26th Pacific-Asia Conf. on Knowledge Discovery and Data Mining. Chengdu: Springer, 2022. 380–391. [doi: 10.1007/978-3-031-05936-0_30]

[21] Lin QJ, Liu H, Sengupta B. Switch trajectory Transformer with distributional value approximation for multi-task reinforcement learning. arXiv:2203.07413, 2022.

[22] Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 6840–6851.

[23] Hong ZW, Kumar A, Karnik S, Bhandwaldar A, Srivastava A, Pajarinen J, Laroche R, Gupta A, Agrawal P. Beyond uniform sampling: Offline reinforcement learning with imbalanced datasets. In: Proc. of the 37th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2024. 4985–5009.

[24] Xu HR, Jiang L, Li JX, Yang ZR, Wang ZR, Chan VWK, Zhan XY. Offline RL with no OOD actions: In-sample learning via implicit value regularization. arXiv:2303.15810, 2023.

[25] Garg D, Hejna J, Geist M, Ermon S. Extreme Q-learning: MaxEnt RL without entropy. arXiv:2301.02328, 2023.

[26] Omura M, Osa T, Mukuta Y, Harada T. Symmetric Q-learning: Reducing skewness of bellman error in online reinforcement learning. Proc. of the AAAI Conf. on Artificial Intelligence, 2024, 38(13): 14474–14481. [doi: 10.1609/AAAI.V38I13.29362]

[27] Jin C, Krishnamurthy A, Simchowitz M, Yu TC. Reward-free exploration for reinforcement learning. In: Proc. of the 37th Int'l Conf. on Machine Learning. JMLR.org, 2020. 4870–4879.

[28] Racaniere S, Lampinen AK, Santoro A, Reichert DP, Firoiu V, Lillicrap TP. Automated curricula through setter-solver interactions. arXiv:1909.12892, 2020.

[29] Yin HL, Lin YJ, Yan J, Meng Q, Festl K, Schichler L, Watzenig D. AGV path planning using curiosity-driven deep reinforcement learning. In: Proc. of the 19th IEEE Int'l Conf. on Automation Science and Engineering. Auckland: IEEE, 2023. 1–6. [doi: 10.1109/CASE56687.2023.10260579]

[30] Li JN, Tang C, Tomizuka M, Zhan W. Hierarchical planning through goal-conditioned offline reinforcement learning. IEEE Robotics and Automation Letters, 2022, 7(4): 10216–10223. [doi: 10.1109/LRA.2022.3190100]

[31] Isele D, Rahimi R, Cosgun A, Subramanian K, Fujimura K. Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. In: Proc. of the 2018 IEEE Int'l Conf. on Robotics and Automation. Brisbane: IEEE, 2018. 2034–2039. [doi: 10.1109/ICRA.2018.8461233]

[32] Wang ZD, Hunt JJ, Zhou MY. Diffusion policies as an expressive policy class for offline reinforcement learning. arXiv:2208.06193, 2023.

[33] Kang BY, Ma X, Du C, Pang TY, Yan SC. Efficient diffusion policies for offline reinforcement learning. In: Proc. of the 37th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2024. 67195–67212.

[34] Chen HY, Lu C, Ying CY, Su H, Zhu J. Offline reinforcement learning via high-fidelity generative behavior modeling. arXiv:2209.14548, 2023.

[35] Jiang CX, Jiang M, Xu QF, Huang X. Expectile regression neural network model with applications. Neurocomputing, 2017, 247: 73–86. [doi: 10.1016/J.NEUCOM.2017.03.040]

[36] Fu J, Kumar A, Nachum O, Tucker G, Levine S. D4RL: Datasets for deep data-driven reinforcement learning. arXiv:2004.07219, 2021.

附中文参考文献:

[2] 刘全, 翟建伟, 章宗长, 钟珊, 周倩, 章鹏, 徐进. 深度强化学习综述. 计算机学报, 2018, 41(1): 1–27. [doi: 10.11897/SP.J.1016.2018.00001]

[3] 刘建伟, 刘媛, 罗雄麟. 深度学习研究进展. 计算机应用研究, 2014, 31(7): 1921–1930, 1942. [doi: 10.3969/j.issn.1001-3695.2014.07.001]

[7] 张伯雷, 刘哲闰. 基于自适应不确定性度量的离线强化学习算法. 南京邮电大学学报 (自然科学版), 2024, 44(4): 98–104. [doi: 10.14132/j.cnki.1673-5439.2024.04.009]

刘全(1969－), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究领域为强化学习, 深度强化学习, 自动推理.

乌兰(1999－), 女, 博士生, 主要研究领域为分层强化学习, 离线强化学习.

颜洁(2000－), 女, 硕士生, 主要研究领域为离线强化学习.