一种用于两人零和博弈对手适应的元策略演化学习算法

吴 哲[1, 2] 李 凯[1, 2] 徐 航[1, 2] 兴军亮[1, 2, 3]

摘 要 围绕两人零和博弈所开展的一系列研究, 近年来在围棋、德州扑克等问题中取得了里程碑式的突破. 现有的两人零和博弈求解方案大多在理性对手的假设下围绕纳什均衡解开展, 是一种力求不败的保守型策略, 但在实际博弈中由于对手非理性等原因并不能保证收益最大化. 对手建模为最大化博弈收益提供了一种新途径, 但仍存在建模困难等问题. 结合元学习的思想提出了一种能够快速适应对手策略的元策略演化学习求解框架. 在训练阶段, 首先通过种群演化的方法不断生成风格多样化的博弈对手作为训练数据, 然后利用元策略更新方法来调整元模型的网络权重, 使其获得快速适应的能力. 在 Leduc 扑克、两人有限注德州扑克 (Heads-up limit Texas Hold′em, LHE) 和 RoboSumo 上的大量实验结果表明, 该算法能够有效克服现有方法的弊端, 实现针对未知风格对手的快速适应, 从而为两人零和博弈收益最大化求解提供了一种新思路.

关键词 两人零和博弈, 纳什均衡, 对手建模, 元学习, 种群演化

引用格式 吴哲, 李凯, 徐航, 兴军亮. 一种用于两人零和博弈对手适应的元策略演化学习算法. 自动化学报, 2022, **48**(10): 2462−2473

DOI 10.16383/j.aas.c211003

A Meta-evolutionary Learning Algorithm for Opponent Adaptation in Two-player Zero-sum Games

WU Zhe[1, 2] LI Kai[1, 2] XU Hang[1, 2] XING Jun-Liang[1, 2, 3]

Abstract Recently, two-player zero-sum games have made impressive breakthroughs in the Go and Texas Hold′em. Most of the existing two-player zero-sum game solutions are based on the assumption of rational opponents to approximate the Nash equilibrium solutions, which is a conservative strategy of trying to be undefeated but does not guarantee maximum payoffs in practice due to the opponents′ irrationality. The opponent modeling provides a new way to maximize the payoff, but modeling has difficulties. This paper proposes a meta-evolutionary learning framework that can quickly adapt to the opponents. In the training phase, we first generate opponents with different styles as training data through the population evolution method, and then use the meta-strategy update method to adjust the network weights of the meta-model so that it can gain the ability to adapt quickly. Extensive experiments on Leduc poker, heads-up limit Texas Hold′em (LHE), and RoboSumo have shown that the algorithm can effectively overcome the drawbacks of existing methods and achieve fast adaptation to unknown style of opponents, thus providing a new way of solving two-player zero-sum games with maximum payoff.

Key words Two-player zero-sum games, Nash equilibrium, opponent modeling, meta learning, population evolution

Citation Wu Zhe, Li Kai, Xu Hang, Xing Jun-Liang. A meta-evolutionary learning algorithm for opponent adaptation in two-player zero-sum games. *Acta Automatica Sinica*, 2022, **48**(10): 2462−2473

两人零和博弈作为博弈论中的一种基础模型, 由于其兼具理论性完备、适应性广泛的特点, 一直是人工智能领域所关注的重要问题. 近年来, 得益于以深度学习[1−2] 为代表的一系列新技术的发展与应用以及计算能力的飞速提升, 两人零和博弈中的一些比较困难的问题诸如围棋[3−7]、德州扑克[8−9] 等已经取得了突破性的进展[10].

围绕纳什均衡解概念[11] 进行求解是目前解决两人零和博弈问题主流的研究思路, 由此产生的一系列均衡求解算法也得到了广泛的研究与发展. 但

收稿日期 2021-10-22 录用日期 2022-03-13
Manuscript received October 22, 2021; accepted March 13, 2022
国家重点研发计划 (2020AAA0103401), 国家自然科学基金 (62076238, 61902402), 中国科学院战略性先导研究项目 (XDA27000000), CCF-腾讯犀牛鸟基金 (RAGR20200104) 资助
Supported by National Key Research and Development Program of China (2020AAA0103401), National Natural Science Foundation of China (62076238, 61902402), Strategic Priority Research Program of Chinese Academy of Sciences (XDA27000000), and CCF-Tencent Open Research Fund (RAGR20200104)
本文责任编委 穆朝絮
Recommended by Associate Editor MU Chao-Xu
1. 中国科学院自动化研究所智能系统与工程研究中心 北京 100190 2. 中国科学院大学人工智能学院 北京 100049 3. 清华大学计算机科学与技术系 北京 100084
1. Center for Research on Intelligent System and Engineering, Institute of Automation, Chinese Academy of Sciences, Beijing 100190 2. School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049 3. Department of Computer Science and Technology, Tsinghua University, Beijing 100084

是随着人们研究的进一步深入以及现实世界关于对抗问题的广泛关注[12], 这种均衡求解方法暴露出越来越多的局限性. 首先, 对于状态空间较大的复杂博弈, 寻找纳什均衡解的计算成本很高[13]. 例如求解德州扑克中的纳什均衡策略需要在整个博弈树上不断迭代求解, 该过程需要计算集群巨大的算力支持和 TB 级别的存储空间[14]. 同时, 近似纳什均衡解的质量并不理想[14], 在两人无限注德州扑克中的近似纳什均衡解可以被简单的局部最佳响应策略剥削. 其次, 求解纳什均衡解的前提条件是假设玩家双方是完全理性的, 但是现实世界中更多面对的是非完全理性的竞争对手. 这也就意味着采取均衡解要放弃剥削非理性对手带来的巨大潜在收益[15].

对手建模是均衡求解法之外的另一种在两人零和博弈中被广泛研究与使用的方法[16−18]. 与逼近纳什均衡解来保证最坏情况下的收益不同, 对手建模方法通常需要基于历史数据来为当前的对手策略拟合一个显式模型, 再根据该模型预测对手下一步的动作, 以此作出更有针对性的决策来获取超额收益. 但这种方法也有明显的局限性: 一方面, 对手建模类方法由于需要历史交互数据来刻画一个显式的对手模型, 因此交互数据的质量和规模都会影响模型刻画的准确性和时效性; 另一方面, 对手建模类方法所刻画的对手模型具有很强的针对性, 一旦对手策略发生了改变, 往往需要从零开始重新建立一个模型.

正是考虑到两人零和博弈领域内现有方法的局限性, 本文创新性地提出一种在博弈过程中可以快速适应未知风格对手的元策略演化学习算法. 本方法不以纳什均衡为求解目标, 因此可以获取远超均衡解的收益, 同时又避免了对手建模类方法对大量交互数据的需求. 图 1 展示了本方法的训练流程.

本文的研究重点和贡献可以被总结为三点. 1) 将元学习思想引入两人零和博弈过程, 训练一个元模型用于未知风格对手的快速适应. 这种快速适应的能力得益于元学习的策略更新方式, 经过训练后的元模型收敛于参数空间内的一类初始点. 该类初始点具有快速更新至目标参数空间的良好性质. 2) 提出了一种基于进化算法的种群训练方法用于提升元模型的泛化能力. 由于元模型需要在参数空间内充分探索以寻找到最佳的初始点, 因此本文提出一种基于进化算法的种群训练方式来为元模型的训练不断提供对手. 一方面, 该种群可以借助进化算法中的交叉 (Crossover)、变异 (Mutation) 算子来不断探索更大的参数空间. 另一方面, 进化算法中的选择 (Selection) 算子通过不断筛选出优质对手来进一步提升元模型的博弈水平. 3) 提高了种群演化算法的训练效率. 元模型在训练期间会周期性地补充到种群中去, 实现了进化算法多样性和梯度算法高效性的充分结合.

1　相关工作

1.1　均衡解求解

在两人零和博弈中, 若存在这样的一个策略组合: 每个策略都是其他策略的最佳响应, 那么这一策略组合便被称为纳什均衡策略组[19]. 在这个策略

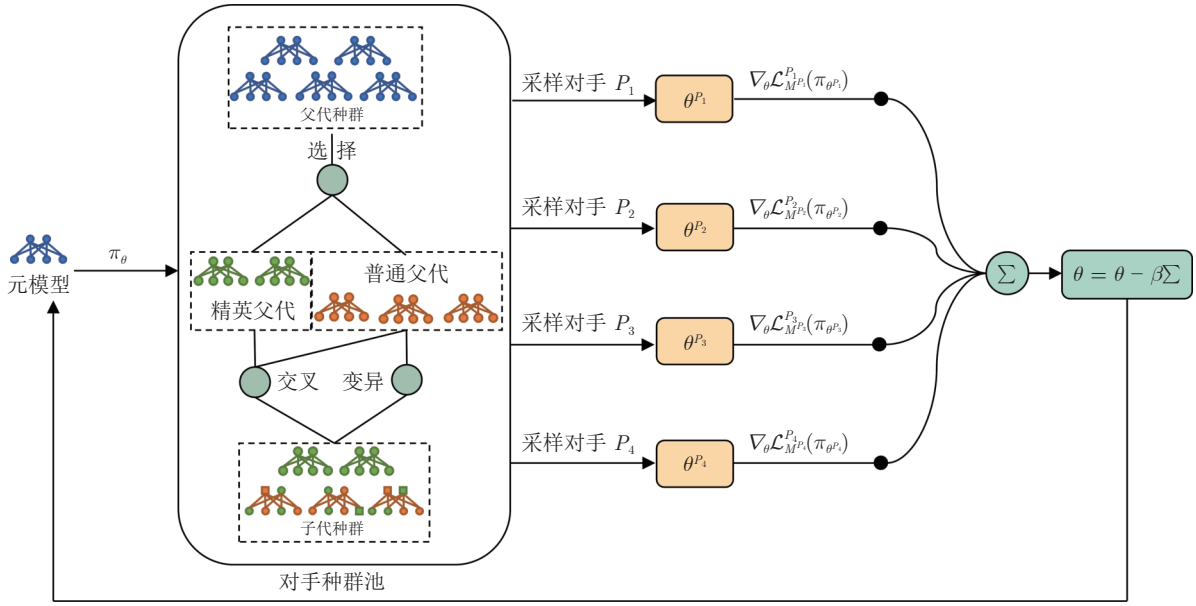

图 1　本文提出的元模型训练方法

Fig. 1　The meta-model′s training process

组中, 每一个玩家都不可能因为单方面改变自身策略而增加收益. 而且在任何一个有限两人零和博弈中都必然存在纳什均衡[19−20]. 因此, 求取近似纳什均衡解来保证最坏情况下的收益, 成为了目前两人零和博弈问题下主流的解决方案.

该类方法目前大多以自博弈 (Self-play) 框架为基础[11], 其中比较典型的做法是神经虚拟自博弈算法 (Neural fictitious self-play, NFSP)[21]. DeepMind 的 Lanctot 等[22] 基于 Double oracle 算法提出一种更加通用的均衡求解框架 PSRO (Policy-space response oracles), 该算法可以看作是 NFSP 思想的进一步拓展. Zinkevich 等[23] 则创新性地提出了一种同样具有良好理论保证的近似均衡求解算法: 反事实遗憾最小化算法 (Counterfactual regret minimization, CFR), 其成为此后构建德州扑克智能体的主流算法[24−26]. MCCFR[27] 和 DeepCFR[28] 则分别结合了蒙特卡洛采样和深度神经网络近似的方法来克服传统 CFR 方法中采样效率低、存储空间大的问题. 但目前这一系列方法仍然面临以求解均衡策略为目标所带来的计算成本高和策略保守无法最大化剥削非理性对手的固有缺点.

1.2 对手建模

与追求纳什均衡的目标不同, 对手建模方法旨在通过为对手建立模型来推测其意图, 从而作出针对性决策以获取更高收益. 常用的对手建模方法可分为显式建模、策略匹配、递归推理等几大类[16].

显式建模类方法[17] 通常基于与对手的大量交互数据针对性地构建一个模型, 以推测对手的决策意图. 这类方法的缺点是对交互数据的需求量大, 且需要不断更新当前的对手模型. 策略匹配类方法[29−30] 通常会建立一个离线数据库, 在与对手的交互过程中为其匹配策略库中的相似模型, 来预测对手的动作. 这类方法虽然省去了从头建立对手模型的步骤, 但是受限于离线数据库的规模, 只针对特定对手有效. 递归推理类方法[31] 考虑的是嵌套推断这类更复杂的决策过程 (即我让你以为我在思考什么). 这类方法的隐患是在诈唬对手的同时可能会被更狡猾的对手反利用. 与以上这些需要大量交互数据来建立对手模型的方法不同, 本文提出的方法并没有显式建立对手模型, 而是通过不断采样不同风格的对手进行训练, 以此获得快速适应的能力.

1.3 元学习

元学习目前已逐渐成为机器学习领域内一个新的研究热潮, 解决的是让智能体学会去学习 (Learning to learn) 的问题. 该方法在训练过程中充分利用各个采样任务中获得的经验, 将其泛化到新环境或新任务. 元学习目前已经在监督学习领域中的分类和回归任务以及强化学习领域的新任务适应问题中取得了一系列进展[32−33]. 本文借助元学习快速适应的特点来克服传统博弈求解方法中存在的策略适应性差、交互数据需求量大等弊端.

目前针对元学习的研究主要集中于以下三类: 1) 距离度量类方法[34], 通过学习不同任务之间的距离度量, 从而达到在距离空间内快速适应新任务的目标; 2) 基于循环神经网络的元学习类方法[35], 通过建立记忆系统来匹配面对新任务的决策模式; 3) 与模型无关的元学习类方法[32], 主要通过训练得到一个比较好的网络初始化参数来加速适应过程. 本文主要结合第三类元学习方法试图在决策空间内找到一组比较好的网络初始化权重, 从而在面对未知对手时可以进行快速适应.

1.4 演化算法

演化算法 (Evolutionary algorithm, EA)[36] 作为一类通用的搜索优化算法, 在人工智能领域得到了广泛应用. 该算法的一大优点是可以在不掌握目标函数及其梯度信息的情况下, 仅依靠迭代评估可行解的好坏来逼近全局最优解. 一些最新的群体智能算法通过引入演化算法的思想, 已经解决了多类博弈问题[37−39]. Jaderberg 等[38] 于 2019 年发表在 Science 上的工作就是借助种群训练 (Population-based training)[37] 的思想对博弈过程中的群体模型进行超参数优化. 它使用选择算子将当前模型权重替换为种群中表现最好的模型权重, 并使用变异算子对模型超参数进行随机扰动. DeepMind 的 Liu 等[39] 将演化算法应用到了合作类型的多智能体足球博弈场景中, 通过配合使用选择、交叉、变异算子, 使智能体在足球博弈场景中学会了复杂的合作策略.

2 两人零和博弈中的快速适应算法

本文所提出的适用于两人零和博弈的快速适应算法会在本节内进行详细介绍. 本算法的目标在于, 克服传统两人零和博弈中均衡求解法不具备策略适应性和对手建模方法对数据量要求大的固有缺陷. 本算法创新性地将元学习方法引入博弈求解过程, 并使用种群演化算法[40−41] 为元模型的训练提供对手模型, 克服了元学习方法依赖手工设计训练任务的弊端.

算法整体架构如图 1 所示. 本算法主要包含元模型训练器 (见第 2.2 节) 和对手种群演化池 (见第 2.3 节) 两个模块. 元模型训练器的目标是优化得到一个元模型, 该元模型在决策空间内拥有良好

的策略初始化. 在面对未知风格对手时, 能够通过少量交互来快速适应其策略, 使得元模型收益尽可能大. 为了提升元模型的泛化性, 对手种群演化池在训练过程中不断提供训练样本给元模型. 与传统元学习方法在图像分类、检测等任务中需要手工设计任务集不同, 对手种群通过进化算法为元模型的训练提供了一套自动的课程学习[42] 方案.

2.1　问题建模

为确保整个算法的通用性和可拓展性, 元模型 M 使用深度神经网络 π_θ 进行建模, 用于训练的对手 P 被建模为 π_ϕ. 由此元模型与对手交互的过程可以建模为两人马尔科夫博弈[5] 过程:

$$G = (S, (A_M, A_P), T, (R_M, R_P)) \tag{1}$$

其中, S 为状态空间, A_M 和 A_P 分别代表双方玩家的动作空间, 系统的状态转移方程定义为 $T: S \times A_M \times A_P \to \Delta(S')$, 玩家的奖励函数 $R_i: S \times A_M \times A_P \times S' \to \mathbf{R}$ 取决于当前状态 S、下一个状态 S' 以及双方玩家的动作. 在给定一个策略已知且固定的对手 P 的情况下, 上述定义的两人马尔科夫博弈 G 就会退化为单人马尔科夫决策过程 (Markov decision process, MDP)[43]:

$$G_M^P = \left(S, A_M, T_M^P, R_M^P\right) \tag{2}$$

G_M^P 的状态与动作空间的定义与 G 相同, 区别在于状态转移函数和奖励函数中嵌入了对手策略. G_M^P 的状态转移函数 T_M^P 和奖励函数 R_M^P 分别定义为:

$$\begin{cases} T_M^P(s, a_M) = T(s, a_M, a_P) \\ R_M^P(s, a_M, s') = R_M(s, a_M, a_P, s') \end{cases} \tag{3}$$

2.2　元模型训练算法

假设目前已经有了训练用的对手策略集合 $\{P_i\}_{i=1}^N$ (对手生成见第 2.3 节), 那么对于每一个训练用的对手 P_i 都可以按照上述定义来构建一个 MDP $G_M^{P_i}$. 元模型的优化目标是通过采样少量的交互轨迹就可以适应对手. 具体来说, 元模型 M (即 π_θ) 首先采样一个固定对手 P_i 与之交互, 收集得到轨迹 $\tau = \{s^{(1)}, a_M^{(1)}, s^{(2)}, \cdots, s^{(t)}, a_M^{(t)}, s^{(t+1)}, \cdots\}$, 记作 $\tau \sim G_M^{P_i}$. 其中 $s^{(t+1)} \sim T_M^{P_i}(s^{(t)}, a_M^{(t)})$, 以及 $a_M^{(t)} \sim \pi_\theta(\cdot|s^{(t)})$. 元模型通过该轨迹 τ 进行一步或多步梯度更新[44] 后, 得到特定于当前对手的适应性参数 θ^{P_i}. 一步梯度更新过程为:

$$\begin{cases} \theta^{P_i} = \theta - \alpha \nabla_\theta \mathcal{L}_M^{P_i}(\pi_\theta) \\ \mathcal{L}_M^{P_i}(\pi_\theta) = -\mathrm{E}_{\tau \sim G_M^{P_i}} \left[\sum\limits_t \gamma^t R_M^{P_i}(s^{(t)}, a_M^{(t)}, s^{(t+1)}) \right] \end{cases} \tag{4}$$

其中, α 代表学习率, $\mathcal{L}_M^{P_i}(\pi_\theta)$ 代表元模型 π_θ 面对固定对手 P_i 的损失函数, γ 代表折扣因子, M^{P_i} 用于表示更新后的元模型. 为了提升元模型的泛化能力, $\{P_i\}_{i=1}^N$ 中的对手策略不断被采样并完成上述梯度更新过程. 算法最终的目标是使得元模型可以快速适应每一个当前对手 P_i 并最大化平均奖励, 即目标函数为:

$$\begin{aligned} J(\theta) = \min_\theta \sum_{i=1}^N \mathcal{L}_{M^{P_i}}^{P_i}(\pi_{\theta^{P_i}}) = \\ \min_\theta \sum_{i=1}^N \mathcal{L}_{M^{P_i}}^{P_i}(\pi_{\theta - \alpha \nabla_\theta \mathcal{L}_M^{P_i}(\pi_\theta)}) \end{aligned} \tag{5}$$

该目标函数是通过针对特定对手适应后的参数 θ^{P_i} 计算的, 但算法的优化目标是元模型的参数 θ, 因此需要对 θ 进行梯度更新:

$$\theta = \theta - \beta \nabla_\theta \sum_{i=1}^N \mathcal{L}_{M^{P_i}}^{P_i}(\pi_{\theta^{P_i}}) \tag{6}$$

其中, β 代表元模型参数更新的步长.

针对该优化目标进行求导后得到:

$$\begin{cases} \nabla_\theta J(\theta) = \mathrm{E}_{\tau' \sim G_M^P(\tau'|\theta')} \left[\nabla_{\theta'} \log P_\tau(\tau' \mid \theta') R(\tau') U \right] \\ U = (I + \alpha \nabla_\theta^2 \mathrm{E}_{\tau \sim G_M^P(\tau|\theta)}[R(\tau)]) \end{cases} \tag{7}$$

其中, τ' 代表适应性参数 θ' 采集的轨迹. 该优化形式与 MAML (Model-agnostic meta-learning) 类元学习算法[32] 的优化目标一致. 结合 Finn 给出的理论保证[45], 只需选用任意一个可进行自动微分的机器学习框架 (例如 PyTorch[46]) 对上述优化形式进行实现, 训练后的元模型理论上就可以获得快速适应新对手的能力. 由于式 (7) 中二阶梯度项的存在, 算法整体的计算复杂度约为 $\mathrm{O}(d^2)$, d 代表博弈空间的问题维度. 为了降低算法复杂度, 本算法可采取 MAML类元学习算法的通用实现方式, 将二阶梯度项忽略, 此时算法复杂度降低为 $\mathrm{O}(d)$, 且整体效果不会受到较大影响[32].

总结来看, 使用上述目标函数进行梯度更新的训练算法旨在找到这样一个元模型: 仅需与对手进行少量交互 (即只进行几次梯度更新) 就可以学会如何适应对手策略并剥削 (奖励最大化). 元模型训练算法的具体步骤详见算法 1. 种群演化对手策略生成 (Opponent strategy generation, OSG) 模块将在第 2.3 节进行详细阐述.

算法 1. 元模型训练算法

输入. α, β, $M(\pi_\theta)$, $P(\pi_\emptyset)$

输出. 具有快速适应性的元模型 M

1) 随机初始化 θ, $\emptyset$
2) 初始化对手策略池 $\mathcal{B}=\{P\}$
3) 对 $t=0,\cdots,T$ 执行:
4) $\quad\mathcal{D}=\text{Evolution}-\text{OSG}(M)$; 见第 2.3 节
5) $\quad$更新对手策略池 $\mathcal{B}=\mathcal{B}\cup\mathcal{D}$
6) $\quad$采样一批对手 $P_i\sim\mathcal{B}$
7) $\quad$对每一个对手 P_i 执行:
8) $\quad\quad$构建单人 MDP $G_M^{P_i}$
9) $\quad\quad M$ 与 P_i 交互, 采样得到轨迹 τ
10) $\quad\quad$根据式 (4) 更新 M, 得到 M^{P_i}
11) 根据式 (6) 更新元模型 M

该算法主要包含两个阶段, 首先通过种群演化算法得到训练用的对手策略集合, 然后再通过元模型训练算法得到快速适应性. 第 2.3 节对种群演化算法进行描述.

2.3 种群演化对手生成

在第 2.2 节中假设已经给出了训练用的对手集合 $\{P_i\}_{i=1}^N$, 元模型 M 通过不断采样对手池中的策略进行训练来提高自身的快速适应性. 本节将对如何自动演化生成对手策略池进行详细阐述. 其主要思想为使用种群演化算法来构建一个课程学习范式, 该算法借助了演化算法的种群优势, 并进一步使用交叉、变异和选择算子来渐进式地提升对手策略的难度与多样性. 元模型通过不断与这些兼顾难度与多样性的对手进行训练, 来逐步提升自身策略的快速适应性与泛化能力. 接下来对种群演化算法的流程进行详细阐述.

在迭代开始前, 首先随机初始化一个对手策略的种群 $\mathcal{B}=\{P_1,P_2,\cdots,P_N\}$, 此时元模型策略 M 也会被补充到对手种群中, 用于引导对手策略在演化初期快速提升博弈水平, 提高训练效率.

种群策略的初始化完成以后, 使用元模型来评估当前种群中每一个对手策略的适应度. 适应度 F 被形式化为对手策略 P_i 与元模型在一个对局步长内获得的累计奖励总和:

$$F=\mathrm{E}_{\tau\sim G_{P_i}^M}\left[\sum_t\gamma^t R_{P_i}^M(s^{(t)},a_{P_i}^{(t)},s^{(t+1)})\right]=\frac{1}{\delta}\sum_\tau\left[\sum_t\gamma^t R_{P_i}^M(s^{(t)},a_{P_i}^{(t)},s^{(t+1)})\right]\tag{8}$$

其中, $G_{P_i}^M$ 为当前迭代轮次的元模型和对手策略构建而成的单人 MDP, $R_{P_i}^M$ 为 P_i 面对本轮固定元模型 M 的收益函数, δ 为用于近似期望值的采样次数.

评估结束后, 选择 (Selection) 算子基于当前种群策略的适应度, 以 ϑ 的比例挑选出适应度靠前的部分个体组成精英团体 E. 对手种群池中剩余部分的策略 $(\mathcal{B}-E)$ 通过与精英团体 E 内的策略进行交叉 (Crossover) 以及自身策略的变异 (Mutation), 来获得提升策略适应度的机会. 精英团体 E 内策略的基因会完整保留到下一代种群中, 不会受到交叉和变异算子的影响, 这使得种群内部表现优异的基因 (网络权重) 可以不断延续下去.

本节所提出的种群演化算法通过不断自动生成对手策略来为元模型提供训练数据. 选择算子的应用使得对手种群的博弈水平逐步提升, 从而为元模型的训练构建了一个课程学习的范式. 同时, 交叉和变异算子的应用丰富了种群策略的多样性. 用于训练的对手策略风格越多样, 元模型的泛化能力就越强. 算法 2 详细描述了种群演化对手生成的具体步骤.

算法 2. 种群演化对手生成算法

输入. 最新版本的元模型 $M(\pi_\theta)$
输出. 更新后的种群策略池 $\mathcal{B}$
1) 补充元模型到对手种群 $\mathcal{B}=\mathcal{B}\cup M$
2) 对 $P_i\in\mathcal{B}$ 执行:
3) $\quad F=\text{evaluate}(P_i)$; 见式 (8)
4) $E=\text{select}(\mathcal{B},\vartheta,F)$ 选出精英
5) 对 $P_i\in(\mathcal{B}-E)$ 执行:
6) $\quad\text{crossover}(P_i,P_j\in E)$
7) $\quad$如果 $\text{random}_{\text{seed}}<\text{mut}_{\text{prob}}$:
8) $\quad\quad\text{mutation}(P_i,\text{mut}_{\text{strength}})$

本文中种群策略池的规模设定主要考虑了环境复杂度和算法收敛速度两个方面, 统一设置为 10. 精英团体 E 占种群策略的比例 ϑ 根据不同问题在 $0.2\sim0.5$ 内取值. 变异率 mut_{prob} 根据不同问题在 $0.1\sim0.3$ 内取值. 变异强度 $\text{mut}_{\text{strength}}$ 设置为 0.1, 即对应 10% 的高斯噪声.

3 实验评估

本节通过构建一系列对比实验, 来验证本文所提出的元模型训练算法在两人零和博弈中的有效性. 第 3.1 节给出了用于实验评估的仿真环境和基线算法介绍. 第 3.2 节给出了算法训练及测试过程中的详细参数设置. 第 3.3 节对实验结果进行对比分析.

3.1 评估环境与基线算法

3.1.1 评估环境

本文所提出的算法在 Leduc 扑克、两人有限注德州扑克 (Heads-up limit Texas Hold′em, LHE) 以及复杂连续空间下的仿真器 RoboSumo[33] 上都进行了实验验证. 这些环境是目前两人零和博弈研

究中被广泛使用的验证平台, 其中两人有限注德州扑克和 RoboSumo 均具有较高的环境复杂度, 同时又分别属于离散和连续动作问题, 因此能够充分评估算法的适应性和可拓展性.

环境具体介绍如下: Leduc 扑克通常包含两种花色 (红桃、黑桃), 每种花色有三个牌型 (J, Q, K), 共计 6 张牌组成. 整个游戏分为两轮, 每个玩家在第一轮分别会得到一张私有牌, 第二轮则只发一张公共牌. 当有一方玩家的私有牌与公共牌组成对子时, 则获得胜利; 若无人组成对子, 牌力高的一方获得胜利. 胜利的一方赢取所有筹码, 牌力相同时平分场上所有筹码. 在发牌前, 每个玩家会被强制下注 1 个筹码, 接下来的两轮下注中, 每轮最多允许有两次加注, 筹码量被分别固定为 2 和 4.

两人有限注德州扑克是现实世界德扑玩家常用的一种扑克玩法. LHE 共包含 52 张牌, 游戏总共有 4 个阶段, 每个玩家在翻牌前阶段 (Pre-flop) 会得到 2 张私有牌, 在后续的翻牌阶段 (Flop)、转牌阶段 (Turn)、河牌阶段 (River) 会分别发出 3 张、1 张、1 张, 总共 5 张公共牌. 玩家在每个阶段的可选动作包括 “过牌 (Check)”、“加注 (Raise)”、“跟牌 (Call)”、“弃牌 (Fold)”. 若无人弃牌则牌力高的一方获得游戏胜利.

RoboSumo 是一个高维连续状态动作空间下的仿真机器人环境. 本文中所用的 RoboSumo-ants 仿真了两只蚂蚁在圆形擂台上角力的竞争型博弈过程. 游戏获得胜利的条件是, 将另一方推出擂台或掀翻. 如果达到时间限制, 该局比赛则以平局结束. 该环境中每个玩家的状态空间由自身的可观测信息和对手的部分可观测信息组成. 玩家自己的可观测信息包括自身关节的位置、速度和接触力, 对手的部分可观测信息主要包括对手关节的位置.

3.1.2　基线算法

为验证元模型训练算法的优越性, 本文与前面所介绍的求解两人零和博弈中的经典方法进行了实验对比. 具体使用的基线算法如下.

1) CFR[23]. 反事实遗憾最小化算法 CFR 被广泛应用于求解两人零和博弈中的近似纳什均衡解. 2) DRON (Deep reinforcement opponent network)[17]. DRON 方法将对手建模算法与深度强化学习相结合, 在交互过程中推断对手类型并进行剥削. 3) EOM (Explicit opponent modeling)[16]. EOM 是一种显式对手建模方法, 通过收集大量对手交互数据来拟合对手模型并进行针对性求解. 4) NFSP[21]. NFSP 通过结合虚拟自博弈与深度强化学习算法, 来求取两人零和博弈中的近似均衡解. 5) MAML[32]. 元学习算法 MAML 已经被成功应用于各种回归、分类以及单智能体强化学习任务. 6) Oracle[17]. Oracle 代表了在两人零和博弈中针对当前对手的近似最佳响应. 7) 本文算法 +PPO. “本文算法 +PPO” 代表将元模型的求解器由 TRPO (Trust region policy optimization)[47] 替换为计算效率更高的近端策略优化算法 (Proximal policy optimization, PPO)[48]. 8) 本文算法 −EA. “本文算法 −EA”代表在本文所提出算法中移除掉种群演化模块, 仅使用元模型更新算法. 9) EA. EA 算法仅使用种群演化算法[36], 不使用元模型的更新方式.

上述基线算法中的 1) 和 4) 是两人零和博弈中求取近似纳什均衡解法的代表性方法, 基线算法 2) 和 3) 是两人零和博弈对手建模解法中隐式建模和显式建模的代表性方法, 基线算法 5) 是单智能体领域元学习解法的代表性方法. 本文将 MAML 训练所需的任务分布替换为对手策略分布, 在每次迭代中, 从中采样一批对手进行交互并完成元模型的更新. 由于真实博弈过程中的对手策略分布并不会被玩家预先知晓, 因此 MAML 类算法仅能依靠随机生成的对手进行训练. 基线算法 6) 代表了理想情况下的最佳响应. 因此, 使用上述基线算法与本文所提出的元模型训练算法进行对比, 可以充分验证算法的有效性.

3.1.3　实验参数设置

本文使用带有 GAE (Generalized advantage estimate)[49] 的 TRPO 算法[47] 作为元模型的求解器, 也可选用 PPO[48] 或其他任意梯度优化算法. 模型架构为两层全连接网络, 激活函数选择 ReLU[50]. 本文所有的训练与评估都在一块 NVIDIA TITAN Xp GPU 上完成, 算法通过 PyTorch 框架[46] 进行部署. 详细实验参数见表 1. 本文中报告的所有实验结果都是在设置 3 个随机种子运行后得到的平均值. 表 2 和表 3 中扑克类环境的结果在每个随机种子下重复对打 10 000 局后统计得出, 代表的是该统计结果下的标准差. 图 2 中 RoboSumo 的实验结果在每个随机种子下重复对打 500 局后统计得出. 图 3 内曲线部分的阴影代表了 3 个随机种子下统计结果 95% 的置信区间. 图 4 的实验结果为对打 10 000 局后统计得出. 图 5 内曲线部分的阴影含义与图 3 相同.

3.2　实验结果与分析

3.2.1　元模型有效性验证

本节用于验证元模型训练算法的有效性. 首先, 对训练得到的元模型的快速适应性进行验证. 模型的适应过程被限制为与当前对手只进行三步梯度更新, 通过统计每步梯度更新后的平均收益来观测当

表 1 不同环境下的实验参数设置
Table 1 Hyperparameters settings

参数	Leduc	LHE	RoboSumo
状态空间	36	72	120
动作空间	4	4	8
网络尺寸	[64, 64]	[128, 128]	[128, 128]
训练步长 α	0.100	0.050	0.003
训练步长 β	0.0100	0.0100	0.0006
折扣系数 γ	0.990	0.995	0.995
梯度更新步数	1	1	1
种群规模	10	10	10
精英比例 ϑ	0.2	0.4	0.4
变异率	0.3	0.1	0.2
变异强度	0.1	0.1	0.1
测试更新步长	0.100	0.050	0.001
测试更新步数	3	3	3
评估局数	50	100	50
存储资源 (GB)	~ 0.3	~ 2.0	~ 1.5
迭代次数 (T)	100	400	300

前元模型的适应能力. 其次, 快速适应后的元模型被进一步用于与各类基线算法进行对比, 通过统计与各类对手 10 000 局对战的平均收益来观测各类算法的表现情况. 在不同任务上的具体实验参数设置见表 1.

需要注意的是, 适应过程中的各类对手都没有用于元模型的训练. 同时, 本文还提供了一系列涵盖不同风格以及不同实力的博弈对手, 用于元模型和各类基线算法的公平对比. Leduc 扑克中各类对手具体包括: 1) Random 对手会随机采取各类动作, 博弈水平相对较弱; 2) Call 对手总是采取跟牌动作, 具有明显的博弈风格; 3) Bluff 对手会根据手牌强弱进行动作并在一定概率内进行诈唬, 博弈水平接近人类玩家; 4) CFR 对手是通过基线方法中的 CFR 训练得到的, 属于近似纳什均衡策略, 很难被剥削; 5) NFSP 对手是通过基线方法中的 NFSP 算法训练得到的. 在规模更大的两人有限注德州扑克 LHE 中, 本文提供了三类更加符合人类玩家特征的对手模型, 分别为比较激进的 LA 型对手, 相对激进并有一定概率诈唬的 TA 型对手, 以及相对保守的 LP 型对手. 这三类对手策略基于专业牌手常用的德州扑克手牌计算器 PokerStove 的缓存矩阵所计算出的获胜概率进行决策. 因此这三类对手策略

表 2 本文算法与基线算法在 Leduc 环境中的对比
Table 2 The average return of our method and baseline methods in Leduc

方法	Random 对手	Call 对手	Bluff 对手	CFR 对手	NFSP 对手
本文算法	1.359 ± 0.023	0.646 ± 0.069	**0.576 ± 0.043**	− 0.162 ± 0.032	**0.325 ± 0.096**
CFR 算法	0.749 ± 0.014	0.364 ± 0.010	0.283 ± 0.028	**0.010 ± 0.024**	0.144 ± 0.007
DRON 算法	1.323 ± 0.014	0.418 ± 0.011	0.409 ± 0.052	− 0.347 ± 0.031	0.212 ± 0.080
EOM 算法	1.348 ± 0.015	0.635 ± 0.007	0.444 ± 0.024	− 0.270 ± 0.042	− 0.012 ± 0.023
NFSP 算法	0.780 ± 0.019	0.132 ± 0.024	0.029 ± 0.022	− 0.412 ± 0.040	0.011 ± 0.027
MAML 算法	**1.372 ± 0.028**	0.328 ± 0.013	0.323 ± 0.044	− 0.409 ± 0.010	0.089 ± 0.051
本文算法 +PPO	1.353 ± 0.011	**0.658 ± 0.005**	0.555 ± 0.017	− 0.159 ± 0.041	0.314 ± 0.012
本文算法 −EA	0.994 ± 0.042	0.611 ± 0.021	0.472 ± 0.038	− 0.224 ± 0.016	0.203 ± 0.029
EA 算法	0.535 ± 0.164	0.422 ± 0.108	0.366 ± 0.113	− 0.365 ± 0.094	0.189 ± 0.102
Oracle	1.373 ± 0.007	0.662 ± 0.014	0.727 ± 0.012	− 0.089 ± 0.016	0.338 ± 0.041

表 3 本文算法与基线算法在 LHE 环境中的对比
Table 3 The average return of our method and baseline methods in LHE

方法	Random 对手	LA 对手	TA 对手	LP 对手
本文算法	2.594 ± 0.089	**0.335 ± 0.012**	**0.514 ± 0.031**	0.243 ± 0.102
DRON 算法	2.131 ± 0.672	− 0.609 ± 0.176	0.294 ± 0.057	0.022 ± 0.028
EOM 算法	2.555 ± 0.020	− 0.014 ± 0.013	0.237 ± 0.023	0.144 ± 0.128
NFSP 算法	1.342 ± 0.033	− 0.947 ± 0.012	− 0.352 ± 0.094	0.203 ± 0.089
MAML 算法	**2.633 ± 0.035**	0.037 ± 0.047	0.231 ± 0.057	0.089 ± 0.051
本文算法 +PPO	2.612 ± 0.058	0.327 ± 0.011	0.478 ± 0.042	**0.246 ± 0.070**
本文算法 −EA	2.362 ± 0.023	0.185 ± 0.049	0.388 ± 0.012	0.119 ± 0.015
EA 算法	2.193 ± 0.158	0.096 ± 0.087	0.232 ± 0.097	0.091 ± 0.009
Oracle	2.682 ± 0.033	0.513 ± 0.009	0.624 ± 0.011	0.270 ± 0.026

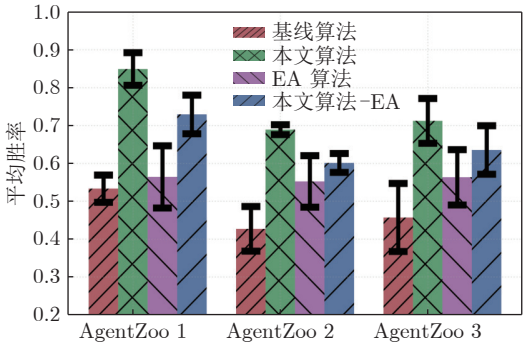

图 2　本文算法与基线算法在 RoboSumo 中的对比

Fig. 2　Comparison of our method with the baseline algorithm in RoboSumo

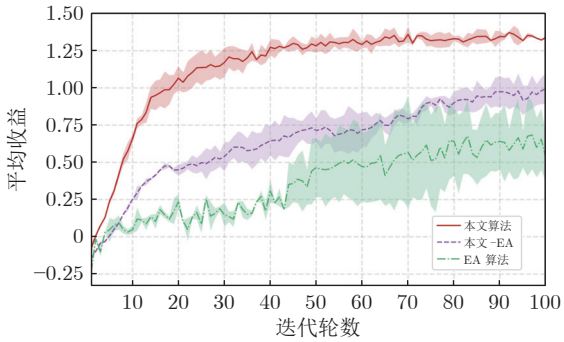

图 3　消融实验

Fig. 3　Ablation study

的博弈水平更加贴近真实人类玩家. RoboSumo 中的对手策略 AgentZoo N ($N = 1, 2, 3$) 为 Bansal 等[51] 使用 PPO 算法训练后开源的预训练模型, 其中 N 的不同取值代表了训练时采用的不同随机种子. 这三类开源的预训练对手模型的博弈水平较高且呈现出了不同博弈风格, 已经被作为基线模型广泛使用[33].

如表 2 所示, 元模型在面对 Leduc 扑克中各类风格的对手时, 都表现出了快速的适应性. 在面对实力较弱或者表现出明显博弈风格的对手 (比如 Random、Call 等类型) 时, 元模型可以快速更新到该类对手的近似最佳响应. 在面对风格多变或者近似均衡这类很难去发现弱点进行剥削的对手时, 元模型也可以快速提高对局中的平均收益. 因此实验表明, 通过结合元学习的策略更新方式, 元模型在策略空间内更新到了拥有快速适应性的初始化区域, 从而在面对未知风格对手时可以快速提升自身博弈水平. 表 3 展示了元模型在 LHE 中也具有类似的快速适应性. 图 2 展示了适应后的元模型相比 RoboSumo 中内置的基线算法的平均胜率, 可以看出适应后的元模型相比于基线模型的胜率得到了大幅提升.

表 2、表 3 还展示了元模型与各类基线算法的实验对比情况. 可以看出, 元模型相比于对手建模类算法 DRON 和 EOM, 在同样的交互数据量以及更少的适应步骤 (三步梯度更新) 内获取了更高的平均收益. 需要注意的是, 本实验中的 DRON 和 EOM 方法使用了与元模型相同的交互数据对对手进行建模, 但是并没有对梯度更新步骤进行限制. 这也与本文前面提到的对手建模类算法需要大量交互数据拟合相对准确的对手模型的特点相一致. 此外, 元模型相比于近似均衡类求解方法 (NFSP 和 CFR), 在面对实力较弱的对手时能够大幅提升平均收益. 而传统的元学习类 MAML 算法仅在面对 Random 类对手时具有快速适应性, 这是由于 MAML 类算法在训练过程中缺乏高质量对手, 仅依靠随机生成的对手进行训练, 无法在参数空间内进行高效探索.

本文所提出的种群演化对手生成算法, 正是通过选择、交叉、变异算子实现了兼顾难度与风格的对手自动生成. 元模型通过不断与这些自动生成的对手进行训练, 逐步提升了自身策略的适应性. 第 3.2.2 节对该对手生成模块的作用机制展开了详细实验验证. 最后, 与基线算法 “本文算法 +PPO” 的实验结果对比也说明了本算法具有与模型无关 (Model-agnostic) 的性质[32], 可以与任何梯度优化算法兼容.

3.2.2　种群演化模块验证

本节主要针对第 2.3 节中提出的种群演化对手生成算法的作用机制展开实验验证. 主要包括: 1) 算法消融性实验; 2) 演化自动生成的对手策略是否具有多样性; 3) 超参数设置对实验性能的影响.

图 3 展示了种群演化模块对元模型实验性能的影响, 三条曲线代表了不同模型在 Leduc 扑克中面对 Random 对手的表现. “本文算法” 代表本文所提算法; “本文算法 −EA” 代表移除种群演化模块后, 仅使用元模型更新算法; “EA 算法” 代表仅使用种群演化算法来生成对手, 不使用元模型的更新方式.

由图 3 可以看出, 元模型的更新方式和种群演化模块对于实验性能的提升都有显著作用. 仅使用种群演化算法的 EA 模型性能提升缓慢, 而且方差比较大, 这是因为进化算法更新过程中并不使用梯度信息, 采样效率较低. 当不使用种群演化模块时, 元模型只能通过自博弈的方式进行策略更新, 会导致元模型在参数空间内探索受限, 泛化性较差. 当同时使用元模型更新方式和种群演化算法时, 模型的性能大幅提升, 而且表现更加稳定. 这是因为种群演化算法通过选择、交叉、变异算子的共同作用,

自动生成了兼顾难度与风格的对手数据, 从而提升了元模型的泛化性与适应性. 同时, 元模型通过与这些对手进行不断交互, 提升了自身快速适应性. 在 Leduc 扑克、两人有限注德州扑克和 RoboSumo 中面对不同种类对手进行的消融实验, 更加充分地验证了本文所提出的种群演化模块和元模型更新算法的有效性, 具体实验结果见表 2、表 3 和图 2.

图 4 可视化了种群演化算法自动生成的对手策略. 这 4 幅图分别对应 Leduc 扑克环境下对手策略种群里 4 个对手模型的动作概率分布. 每幅图中的动作概率分布均由元模型与相应对手模型对打 10 000 局后统计得出. 横坐标代表了 Leduc 中的不同牌型, 纵坐标代表了持有该牌型时各个动作的概率. 图 4 下方的 4 个图例分别代表了 Leduc 中 4 个不同的可选动作. 从图 4 中可以看出, 自动生成的对手策略覆盖了更大的策略空间, 而且由于选择、交叉、变异算子的共同作用, 自动生成的对手策略并不仅仅关注策略上的差异性, 还兼顾了模型的性能. 例如, 在拿到较强牌力的手牌时, 弃牌的概率相应减小.

超参数的选择对实验性能也会造成一定影响, 表 1 中详细展示了通过网格搜索确定的一组超参数. 其中, “评估局数”的设置会对实验性能产生较大影响. 图 5 显示了评估局数的不同取值对元模型性能的影响. 当评估局数较少时, 模型性能出现了剧烈震荡; 随着评估局数的增加, 模型的方差逐渐变小, 性能也随之提升. 这是因为该参数代表了对手生成模块中选择算子评估对手模型性能的对局数量, 这直接影响了子代种群的模型性能. 通过实验发现, 评估局数较多时挑选出的子代种群有利于模型性能的提升, 在本文的三类实验场景中, 50 ~ 100 是一个比较合理的范围, 能够兼顾评估速度与质量.

4 结论

本文提出了一种针对两人零和博弈的快速适应算法, 该算法克服了均衡求解方法中策略过于保守以及对手建模方法泛化性差的弊端. 本算法主要分为对手自动生成和元模型更新两个阶段. 在对手生成阶段, 通过种群演化算法中的选择、交叉和变异算子来自动生成兼具不同风格与博弈水平的对手策略. 在元模型更新阶段, 模型通过不断与这些对手

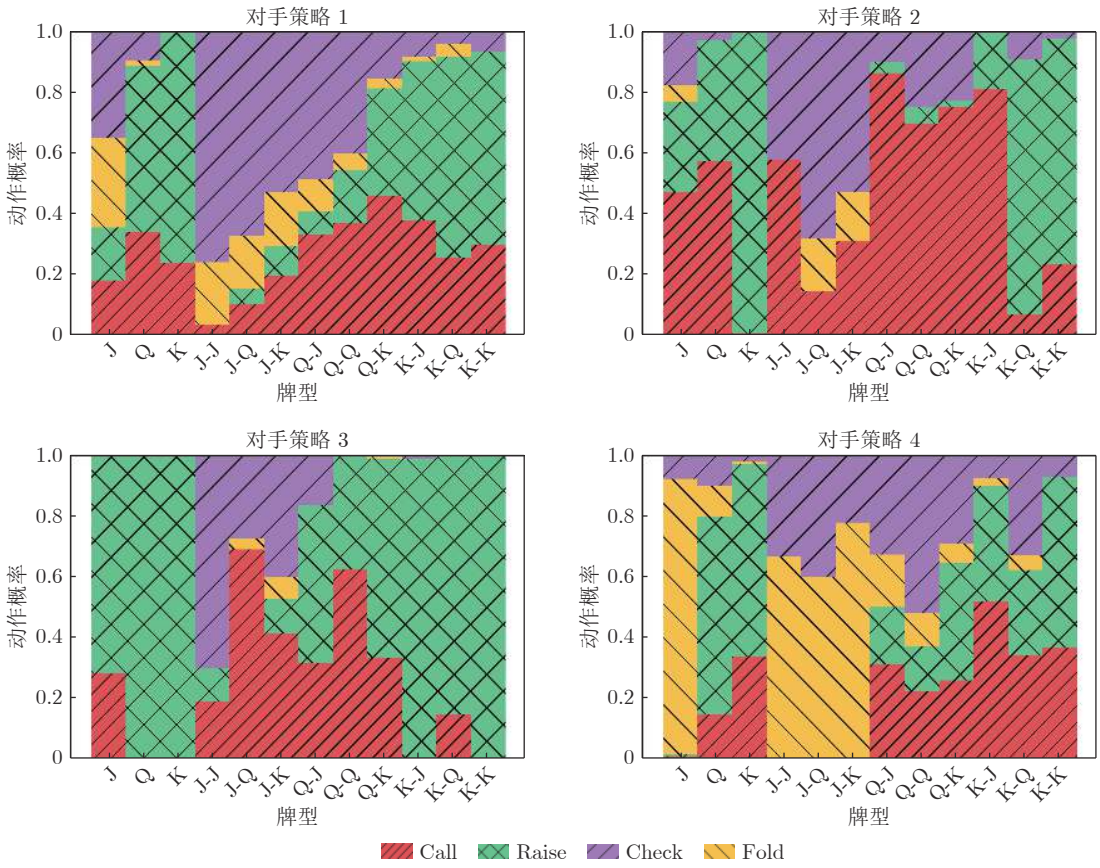

图 4 种群演化模块生成的对手策略

Fig. 4 Visualization of the styles of the strategies

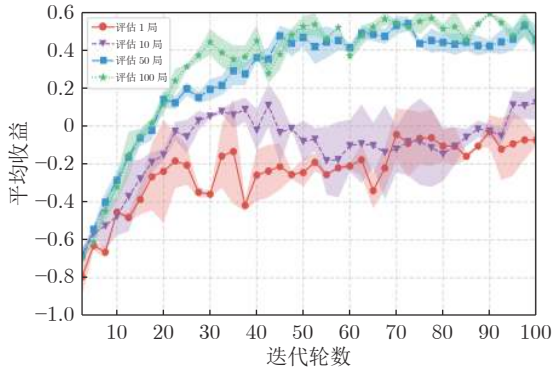

图 5　超参数设置对模型性能影响

Fig. 5　Effect of hyperparameter settings

进行交互并结合元学习的参数更新方式, 来不断提升模型面对不同博弈对手的泛化能力.

在 Leduc 扑克、LHE 以及 RoboSumo 中的实验结果表明, 本文算法在与对手进行少量交互的情况下, 使用同一个元模型针对不同风格的对手都实现了快速适应. 相比于均衡类算法, 本文算法能够实现对次优对手的剥削; 相比于对手建模类算法, 本文算法避免了显式建模过程, 提高了模型的泛化能力. 消融实验与可视化实验的结果显示了本文算法所使用的对手演化生成模块与元模型更新算法的有效性. 如何将本文算法应用到更大规模的游戏 (例如, 无限注德州扑克) 以及如何结合均衡类算法的优点来保证元模型在适应过程中的安全性, 是本文未来可以继续研究的方向.

References

1　LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*, 2015, **521**(7553): 436−444

2　Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks. *Science*, 2006, **313**(5786): 504−507

3　Silver D, Huang A, Maddison C J, Guez A, Sifre L, Van Den Driessche, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 2016, **529**(7587): 484−489

4　Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, et al. Mastering the game of go without human knowledge. *Nature*, 2017, **550**(7676): 354−359

5　Silver D, Hubert T, Schrittwieser J, Antonoglou I, Lai M, Guez A, et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 2018, **362**(6419): 1140−1144

6　Zhao Dong-Bin, Shao Kun, Zhu Yuan-Heng, Li Dong, Chen Ya-Ran, Wang Hai-Tao, et al. Review of deep reinforcement learning and discussions on the development of computer Go. *Control Theory and Applications*, 2016, **33**(6): 701−717
(赵冬斌, 邵坤, 朱圆恒, 李栋, 陈亚冉, 王海涛, 等. 深度强化学习综述: 兼论计算机围棋的发展. 控制理论与应用, 2016, **33**(6): 701−717)

7　Zhou Zhi-Hua. AlphaGo special session: An introduction. *Acta Automatica Sinica*, 2016, **42**(5): 670
(周志华. AlphaGo 专题介绍. 自动化学报, 2016, **42**(5): 670)

8　Sandholm T. Solving imperfect-information games. *Science*, 2015, **347**(6218): 122−123

9　Bowling M, Burch N, Johanson M, Tammelin O. Heads-up limit Hold′em poker is solved. *Science*, 2015, **347**(6218): 145−149

10　Guo Xiao-Xiao, Li Cheng, Mei Qiao-Zhu. Deep learning applied to games. *Acta Automatica Sinica*, 2016, **42**(5): 676−684
(郭潇逍, 李程, 梅俏竹. 深度学习在游戏中的应用. 自动化学报, 2016, **42**(5): 676−684)

11　Brown G W. Iterative solution of games by fictitious play. *Activity Analysis of Production and Allocation*, 1951, **13**(1): 374−376

12　Shen Yu, Han Jin-Peng, Li Ling-Xi, Wang Fei-Yue. AI in game intelligence —— From multi-role game to parallel game. *Chinese Journal of Intelligent Science and Technology*, 2020, **2**(3): 205−213
(沈宇, 韩金朋, 李灵犀, 王飞跃. 游戏智能中的 AI —— 从多角色博弈到平行博弈. 智能科学与技术学报, 2020, **2**(3): 205−213)

13　Tammelin O. Solving large imperfect information games using CFR+. arXiv preprint arXiv: 1407.5042, 2014.

14　Moravcík M, Schmid M, Burch N, Lisý V, Morrill D, Bard N, et al. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 2017, **356**(6337): 508−513

15　Brown N, Sandholm T. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 2018, **359**(6374): 418−424

16　Albrecht S V, Stone P. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 2018, **258**: 66−95

17　He H, Boyd-Graber J, Kwok K, DauméIII H. Opponent modeling in deep reinforcement learning. In: Proceedings of the 33rd International Conference on Machine Learning. New York, USA: PMLR, 2016. 1804−1813

18　Foerster J N, Chen R Y, Al-Shedivat M, Whiteson S, Abbeel P, Mordatch I. Learning with opponent-learning awareness. In: Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems. Richland, USA: ACM, 2018. 122−130

19　Nash J. Non-cooperative games. *Annals of Mathematics*, 1951, **54**: 286−295

20　Neumann J V, Morgenstern O. *The Theory of Games and Economic Behaviour*. New Jersey: Princeton University Press, 1944.

21　Heinrich J, Silver D. Deep reinforcement learning from self-play in imperfect-information games. arXiv preprint arXiv: 1603.01121, 2016.

22　Lanctot M, Zambaldi V, Gruslys A, Lazaridou A, Tuyls K, Pérolat J, et al. A unified game-theoretic approach to multiagent reinforcement learning. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, USA: Curran Associates Inc, 2017. 4193−4206

23　Zinkevich M, Johanson M, Bowling M, Piccione C. Regret minimization in games with incomplete information. In: Proceedings of the 21st International Conference on Neural Information Processing Systems. Vancouver, Canada: Curran Associates Inc, 2007.

24 Brown N, Sandholm T. Solving imperfect-information games via discounted regret minimization. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence. Honolulu, USA: AAAI Press, 2019. 1829−1836

25 Johanson M, Bard N, Burch N, Bowling M. Finding optimal abstract strategies in extensive form games. In: Proceedings of the 26th AAAI Conference on Artificial Intelligence. Toronto, Canada: AAAI Press, 2012. 1371−1379

26 Heinrich J, Lanctot M, Silver D. Fictitious self-play in extensive-form games. In: Proceedings of the 32nd International Conference on Machine Learning. Lille, France: PMLR, 2015. 805−813

27 Lanctot M, Waugh K, Zinkevich M, Bowling M. Monte Carlo sampling for regret minimization in extensive games. In: Proceedings of the 23rd Advances in Neural Information Processing Systems. Vancouver, Canada: Curran Associates Inc, 2009. 1078−1086

28 Brown N, Lerer A, Gross S, Sandholm T. Deep counterfactual regret minimization. In: Proceedings of the 36th International Conference on Machine Learning. Long Beach, USA: PMLR, 2019. 793−802

29 Hernandez-Leal P, Rosman B, Taylor M E, Sucar L E, Munoz De Cote E. A Bayesian approach for learning and tracking switching, non-stationary opponents. In: Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems. Singapore: ACM, 2016. 1315−131

30 Rosman B, Hawasly M, Ramamoorthy S. Bayesian policy reuse. *Machine Learning*, 2016, **104**(1): 99−127

31 Letcher A, Foerster J, Balduzzi D, Rocktäschel T, Whiteson S. Stable opponent shaping in differentiable games. In: Proceedings of the 7th International Conference on Learning Representations. New Orleans, USA: OpenReview.net, 2019.

32 Finn C, Abbeel P, Levine S. Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia: PMLR, 2017. 1126−1135

33 Al-Shedivat M, Bansal T, Burda Y, Sutskever I, Mordatch I, Abbeel P. Continuous adaptation via meta-learning in nonstationary and competitive environments. In: Proceedings of the 6th International Conference on Learning Representations. Vancouver, Canada: OpenReview.net, 2018.

34 Vinyals O, Blundell C, Lillicrap T, Wierstra D. Matching networks for one shot learning. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. Barcelona, Spain: Curran Associates Inc, 2016. 3630−3638

35 Ravi S, Larochelle H. Optimization as a model for few-shot learning. In: Proceedings of the 5th International Conference on Learning Representations. Toulon, France: OpenReview.net, 2017.

36 Drugan M M. Reinforcement learning versus evolutionary computation: A survey on hybrid algorithms. *Swarm and Evolutionary Computation*, 2019, **44**: 228−246

37 Jaderberg M, Dalibard V, Osindero S, Czarnecki W M, Donahue J, Razavi A, et al. Population based training of neural networks. arXiv preprint arXiv: 1711.09846, 2017.

38 Jaderberg M, Czarnecki W M, Dunning I, Marris L, Lever G, Castaneda A G, et al. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 2019, **364**(6443): 859−865

39 Liu S, Lever G, Merel J, Tunyasuvunakool S, Heess N, Graepel T. Emergent coordination through competition. In: Proceedings of the 7th International Conference on Learning Representations. New Orleans, USA: OpenReview.net, 2019.

40 Khadka S, Tumer K. Evolution-guided policy gradient in reinforcement learning. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal, Canada: Curran Associates Inc, 2018. 1196−1208

41 Conti E, Madhavan V, Petroski Such F, Lehman J, Stanley K, Clune J. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal, Canada: Curran Associates Inc, 2018. 5032−5043

42 Elman J L. Learning and development in neural networks: The importance of starting small. *Cognition*, 1993, **48**(1): 71−99

43 Liang Xing-Xing, Feng Yang-He, Ma Yang, Cheng Guang-Quan, Huang Jin-Cai, Wang Qi, et al. Deep multi-agent reinforcement learning: A survey. *Acta Automatica Sinica*, 2020, **46**(12): 2537−2557
(梁星星, 冯旸赫, 马扬, 程光权, 黄金才, 王琦, 等. 多 Agent 深度强化学习综述. 自动化学报, 2020, **46**(12): 2537−2557)

44 Sun Chang-Yin, Mu Chao-Xu. Important scientific problems of multi-agent deep reinforcement learning. *Acta Automatica Sinica*, 2020, **46**(7): 1301−1312
(孙长银, 穆朝絮. 多智能体深度强化学习的若干关键科学问题. 自动化学报, 2020, **46**(7): 1301−1312)

45 Finn C B. Learning to Learn With Gradients [Ph.D. dissertation], University of California, Berkeley, 2018

46 Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. Pytorch: An imperative style, high-performance deep learning library. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. Vancouver, Canada: Curran Associates Inc, 2019. 8024−8035

47 Schulman J, Levine S, Abbeel P, Jordan M, Moritz P. Trust region policy optimization. In: Proceedings of the 32nd International Conference on Machine Learning. Lille, France: PMLR, 2015. 1889−1897

48 Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. arXiv preprint arXiv: 1707.06347, 2017.

49 Schulman J, Moritz P, Levine S, Jordan M, Abbeel P. High-dimensional continuous control using generalized advantage estimation. In: Proceedings of the 4th International Conference on Learning Representations. San Juan, Puerto Rico: OpenReview.net, 2016.

50 Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of the 4th International Conference on Artificial Intelligence and Statistics. Fort Lauderdale, USA: JMLR, 2011. 315−323

51 Bansal T, Pachocki J, Sidor S, Sutskever I, Mordatch I. Emergent complexity via multi-agent competition. In: Proceedings of the 6th International Conference on Learning Representations. Vancouver, Canada: OpenReview.net, 2018.

吴 哲 中国科学院自动化研究所智能系统与工程研究中心硕士研究生. 2019 年获得山东大学工学学士学位. 主要研究方向为计算机博弈, 强化学习. E-mail: wuzhe2019@ia.ac.cn

(**WU Zhe** Master student at the Center for Research on Intelligent System and Engineering, Institute of Automation, Chinese Academy of Sciences. He received his bachelor degree in engineering from Shandong University in 2019. His research interest covers computer game and reinforcement learning.)

李 凯 中国科学院自动化研究所智能系统与工程研究中心副研究员. 2018 年获得中国科学院自动化研究所模式识别与智能系统博士学位. 主要研究方向为计算机博弈, 强化学习. E-mail: kai.li@ia.ac.cn

(**LI Kai** Associate professor at the Center for Research on Intelligent System and Engineering, Institute of Automation, Chinese Academy of Sciences. He received his Ph.D. degree in pattern recognition and intelligent system from Institute of Automation, Chinese Academy of Sciences in 2018. His research interest covers computer game and reinforcement learning.)

徐 航 中国科学院自动化研究所智能系统与工程研究中心硕士研究生. 2020 年获得武汉大学工学学士学位. 主要研究方向为计算机博弈, 强化学习. E-mail: xuhang2020@ia.ac.cn

(**XU Hang** Master student at the Center for Research on Intelligent System and Engineering, Institute of Automation, Chinese Academy of Sciences. He received his bachelor degree in engineering from Wuhan University in 2020. His research interest covers computer game and reinforcement learning.)

兴军亮 清华大学计算机科学与技术系研究员. 2012 年获得清华大学计算机科学与技术系博士学位. 主要研究方向为计算机博弈. 本文通信作者. E-mail: jlxing@nlpr.ia.ac.cn

(**XING Jun-Liang** Professor in the Department of Computer Science and Technology, Tsinghua University. He received his Ph.D. degree in Department of Computer Science and Technology from Tsinghua University in 2012. His main research interest is computer game. Corresponding author of this paper.)