基于自注意力机制和策略映射重组的多智能体强化学习算法

李静晨[1)] 史豪斌[1)] 黄国胜[1),2)]

[1)](西北工业大学计算机学院 西安 710072)

[2)](“高雄中山大学”电机系 高雄 000800)

摘 要 多智能体强化学习(Multi-Agent Reinforcement Learning,MARL)在群体控制领域中被广泛应用,但由于单个智能体的马尔可夫决策模型被破坏,现有的 MARL 算法难以学习到最优策略,且训练中智能体的随机性会导致策略不稳定.本文从状态空间到行为空间的映射出发,研究同构多智能体系统的耦合转换,以提高策略的先进性及稳定性.首先,我们调查了同构智能体行为空间的重组,打破智能体与策略对应的固定思维,通过构建抽象智能体将智能体之间的耦合转换为不同智能体行为空间同一维度的耦合,以提高策略网络的训练效率和稳定.随后,在重组策略映射的基础上,我们从序列决策的角度出发,为抽象智能体的策略网络和评估网络分别设计自注意力模块,编码并稀疏化智能体的状态信息.重组后的状态信息经过自注意力编码后,能显示地解释智能体的决策行为.本文在三个常用的多智能体任务上对所提出方法的有效性进行了全面的验证和分析,实验结果表明,在集中奖励的情况下,本文所提出的方法能够学到比基线方法更为先进的策略,平均回报提高了 20%,且训练过程与训练结果的稳定性提高了 50%以上.多个对应的消融实验也分别验证了抽象智能体与自注意力模块的有效性,进一步为我们的结论提供支持.

关键词 多智能体系统;多智能体强化学习;深度强化学习;注意力机制

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2022.01842

A Multi-Agent Reinforcement Learning Method Based on Self-Attention Mechanism and Policy Mapping Recombination

LI Jing-Chen[1)] SHI Hao-Bin[1)] HWANG Kao-Shing[1),2)]

[1)](*School of Computer Science, Northwestern Polytechnical University, Xi'an* 710072)

[2)](*Department of Electrical Engineering, "Kaohsiung Sun Yat-sen University", Kaohsiung* 000800)

Abstract Multi-Agent Reinforcement Learning (MARL) has been widely applied in the group control field. Due to the Markov decision process for an agent is broken in MARL, the existing MARL methods are hard to learn optimal policies, and policies are unstable because the random behaviors of agents in MARL. From the viewpoint of the mapping between state spaces and behavior spaces, this work studies the coupling among agents in homogeneous MARL, aiming at enhancing the policy effectiveness and training stability. We first investigate the recombination of the joint behavior space for homogeneous agents, breaking the one-to-one correspondence between agents and policies. Then the abstract agents are proposed to transform the coupling among agents into that among the actions in the behavior space, by which the training efficiency and stabilization are improved. Based on the former, inspiring by sequential decisions, we design

收稿日期:2021-10-14;在线发布日期:2022-04-10.本课题得到国家自然科学基金(61976178,62076202)、之江实验室开放课题(2022NB0AB07)、中国陕西省重点研发计划项目(2022GY-090)、中国人工智能学会-华为 MindSpore 学术奖励基金(CAAIXSJLJJ-2021-041A)、西北工业大学博士生创新基金(CX2022016)资助.**李静晨**,博士研究生,主要研究方向为智能控制.E-mail:staubs1212@mail.nwpu.edu.cn.**史豪斌**(通信作者),博士,教授,主要研究领域为智能控制、机器学习.E-mail:shihaobin@nwpu.edu.cn.**黄国胜**,博士,教授,主要研究领域为智能控制、机器学习.

self-attention modules for the abstract agents' policy networks and evaluation networks respectively, encoding and thinning the states of agents. The learned policies can be explicitly explained through the self-attention module and the recombination. The proposed method is validated in three simulated MARL scenarios. The experimental results suggest that our method can outperform baseline methods in the case of centralized rewards, while the stability can be increased more than fifty percent by our method. Some ablation experiments are designed to validate the abstract agents and self-attention modules respectively, making our conclusion more convincing.

Keywords　Multi-Agent system; Multi-Agent reinforcement learning; deep reinforcement learning; attention mechanism

1 引　言

实现多个决策主体的自动化控制是人工智能所要解决的关键问题之一. 强化学习[1]作为机器学习中的一个代表性算法，通过使智能体不断地与环境交互来获得经验样本，以实现对智能体策略的更新. 强化学习被认为是发展具有思考、计划和解决问题的强人工智能[2](Artificial General Intelligence)不可或缺的关键性技术. 而面向数个控制对象的多智能体强化学习[3](Multi-Agent Reinforcement Learning, MARL)，则被寄希望于解决多智能体系统的决策任务. 相比起传统的控制理论，MARL 的策略学习过程更为普适，且在策略学习过程结束后能达到实时决策. 此外，深度神经网络的引入也使得强化学习技术能适应高维的连续状态空间，让 MARL 逐渐成了多智能体系统领域的主流，并在多无人机协同控制[4]、机器人足球[5]、计算资源调度[6]等应用中大放异彩.

学者们对 MARL 技术的研究过程可分为多个阶段. 在早先的研究中，联合动作学习[7]将所有智能体看作一个整体，利用强化学习去训练整体的联合策略. 而独立动作学习[8]则忽视将其他智能体对环境的影响，为每个智能体分配互不干扰的策略. 这些对 MARL 浅层的尝试让学者们逐渐意识到 MARL 所面临的挑战，开始了新一轮的探索和研究. 智能体数目变多意味着联合状态空间与动作空间维度的指数增大，让基于完整马尔科夫决策过程的联合动作学习需要非常多的计算资源才能保证策略的收敛. 针对这一问题，从多智能体训练过程的角度出发，Lowe 等人[9]引入了 Actor-Critic 架构来分离强化学习的执行和评估过程，提出了多智能体深度确定性策略梯度(Multi-Agent Deep Deterministic Policy Gradient, MADDPG)算法，同时将集中训练分散执行(Centralized Training and Decentralized Execution, CTDE)机制带到了 MARL 领域中. CTDE 机制允许智能体在独立执行自身策略时通过集中性的 Critic 网络获得对自身策略的无偏估计，保证了多智能体策略的收敛，也迅速成为了多智能体集中学习的基础. 随后，基于 CTDE 机制，其他研究者从训练过程出发，不断地对 MADDPG 进行改进. 例如，Rashid 等人[10]在集中性评估模块中构建了一个用于估计联合动作值函数的神经网络，提出了 QMIX 算法，并在《星际争霸》测试平台上验证了其有效性；Foerster 等人[11]提出了反事实基线，将单个智能体的行为边缘化，来优化 CTDE 机制中的集中性评估. 这一类从强化学习训练过程中进行优化的 MARL 方法的主要贡献在对于智能体的值函数计算上，使其不受限于智能体的种类和智能体间的关系，而不足之处在于，智能体的行为仅仅由观测决定，需要大量的探索才能靠集中性评估推进智能体间协同，这无疑增加了训练所需要的样本数量.

为了加速智能体间协同行为的学习，提高智能体的一致性共识，部分研究者从信息整合的角度出发，构建交流模型来为智能体传递信息. 这种思路在部分可观测的环境中更为常见. 智能体间的信息共享可以加强对环境的感知，且使单个智能体可以考虑其他智能体的行为来进行决策. CommNet(Communication Network)[12]是最早的多智能体交流网络模型之一，为多智能体设计了一个集中性的观测编码网络，将所有智能体的观测输入到该网络中，并为每个智能体分发对应的交流信息. 伴随着注意力机制在自然语言处理领域的成功[13]，一些研究者尝试用注意力网络来优化集中性通信模型，典型的有 ATOC (Attentional Communication)[14]模型和 TarMAC (Targeted Multi-Agent Communication)[15]网络，都

是利用注意力机制去处理不同智能体的观测序列，获得对观测差异的软掩码. 不同处在于，ATOC 是为了解决智能体何时去与他人交流以及如何获得有效交流的问题，而 TarMAC 则兼顾了对通信对象的选择，允许智能体向不同通信对象传递不同信息. 这种集中性的通信模型主要起到整合观测的作用，减少因观测不完整所造成类似随机过程的影响. 而在多智能体系统缺乏集中性设备时，由于每个智能体需要集中通信模块发布信息，这类算法又难以在分布式环境下部署. 考虑到对这种问题，以循环神经网络为基础的循环通信模型则更容易应用到分散的部分可观测马尔科夫决策过程(Decenteralized Partilly Observable Markov Decision Process，Dec-POMDP)[16]中. 依靠交流序列在循环神经网络[17]中的不断传递，单个智能体能够在缺乏集中性交流网络的情况下得到其他智能体的信息. 这种思路也诞生了许多分布式通信的 MARL 方法，如 BicNet(Bidirectionally-Coordinated Nets)[18]和 ARMI(Attentional and Recurrent Message Integration)[19].

当我们回顾 MARL 的发展历程，不难发现，很少有研究者从策略本身来优化 MARL. 大多数的工作都在于研究如何提高多智能体的协同能力，以及保证学习的收敛，忽视了智能体联合状态空间与动作空间的关系. 在本文中，我们提供了一种优化 MARL 的新思路，即策略映射转换(Policy Mapping Transformation，PMT). 首先，我们重新思考了强化学习中“智能体”所扮演的角色，并研究“智能体”在 MARL 中的不同表现形式. 随后将智能体的策略看作从状态空间到行为空间的非线性映射，并调查这种映射在深度神经网络中的表达. 面向同构的多智能体系统，本文首先研究多智能体的策略映射重组技术，将多个智能体间的协同转化为行为空间中各个维度的协同. 此外，本文利用自注意力机制来稀疏化状态空间到行为空间的编码. 在 Actor-Critic 架构的基础上，分别为智能体的决策和评估设计不同的自注意力模块，为多智能体学习到更为先进和平稳的策略，同时提高所学策略的可解释性[20]. 最后，我们在几个多智能体仿真环境中调查了所提出方法的表现，并与现有的 MARL 进行了实验对比. 实验结果表明本文所提出的方法优于现有的 MARL 技术，且通过消融实验进一步验证了算法的可行性.

本文的主要贡献有三个方面：

(1) 进行了对 MARL 中“智能体”的重新思考，打破了强化学习对多智能体联合动作的固有封装模式. 从联合状态空间与联合行为空间之间的映射关系出发，为 MARL 领域提供了新的研究角度.

(2) 本文同时考虑智能体间的协同与智能体行为空间中不同动作的关系，提出了多智能体策略映射重组技术，将多智能体间的耦合转化为不同智能体行为空间相同维度的耦合，并讨论了这种耦合转化的优势，以及相应的理论分析.

(3) 利用自注意力机制实现了多智能体策略映射的软稀疏编码，为多智能体提供更为先进且可解释的策略. 针对决策和评估分离的强化学习架构，根据马尔科夫决策过程分别为 Actor 网络和 Critic 网络设计自注意力网络，优化多智能体的策略学习.

2 背景知识

2.1 多智能体深度强化学习

作为强化学习与多智能体系统的交叉领域，MARL 也是通过马尔科夫决策过程模型与贝尔曼方程来实现策略的搜索与更新[21]. 给定一个包含 N 个智能体的多智能体环境 $\langle S, A, P, R\rangle$，其中 $S=[S_1, S_2, \cdots, S_N]$ 表示联合状态空间，$A=[A_1, A_2, \cdots, A_N]$ 代表联合动作空间，$P: S \leftarrow (S, A)$ 为状态转移概率，R 为多智能体系统的集中奖励，在分散奖励的情况下 $R=[R_1, R_2, \cdots, R_N]$. 在 MARL 中，每个智能体需要被分配一个策略 $\pi(a|s)$，多智能体系统则根据他们的联合策略 $(\pi_1, \pi_1, \cdots, \pi_N)$ 不断地与环境交互，获得经验样本来优化联和策略[22]，而联合策略的优化目标则是最大化期望奖励

$$E(R)=\sum_{i=1}^{N}\left(\sum_{t=t_0}^{T}\gamma^{t-t_0}R^i(s_t^i, a_t^i\sim\pi_i(s_t^i))\right) \quad (1)$$

其中 R^i 是第 i 个智能体的奖惩函数，$s_t^i\in S_i$ 是第 i 个智能体在 t 时刻的状态，$a_t^i\in A_i$ 是该智能体根据自身策略所选择的动作，$\gamma\in[0,1]$ 为折扣因子，用来控制未来奖励的占比. 在合作型多智能体系统中，环境为多智能体反馈一个集中的奖励 R，此时期望奖励可写为

$$E(R)=\sum_{t=t_0}^{T}\gamma^{t-t_0}R(s_t^1, \cdots, s_t^N, a_t^1, \cdots, a_t^N) \quad (2)$$

Actor-Critic 架构常被用来分离策略执行和策略评估过程，同时也防止梯度方差过大的问题. 在 MARL 中，Actor-Critic 架构的形式与训练机制有关. 联合学习意味着所有智能体共享一个 Actor 网络，其输入是整体的联合观测，输出是联合动作，而联合学习的 Critic 网络也直接用来计算联合动作的

Q 值. 在分散学习中，每个智能体被指定一个 Actor 网络，根据自身观测、整体观测、或包含交流信息的观测来决定自身的行为. 与分布式学习不同，分散学习并不是指每个智能体独立完成整个训练过程，而是将智能体的决策过程建模成一个 Dec-POMDP. 分散学习中的集中性评估（即 CTDE 机制）是指每个智能体在计算 Q-值时，需要考虑其他智能体的观测与行为，从而获得对自身 Q-值的无偏估计. 相应的，在完全分散的 MARL 中，每个智能体的 Actor 网络和 Critic 网络都不考虑其他智能体的信息.

在 CTDE 机制中，Critic 网络计算在其他智能体行为已知的情况下，单个智能体的 Q 值 $Q(s_t^i,a_t^i,x_t^i)$，其中 $x_t^i=(s_t^1,\cdots,s_t^N,a_t^1,\cdots,a_t^N)/(s_t^i,a_t^i)$ 代表其他智能体的状态和动作. 第 i 个 Critic 网络的损失函数为

$$L(\theta_c^i)=\frac{1}{2}[Q(s_t^i,a_t^i,x_t^i)-r_t^i-\gamma Q(s_{t+1}^i,a,x_{t+1}^i)\big|_{a\sim\pi_i(s_{t+1}^i)}]^2 \tag{3}$$

其中 θ_c^i 是该 Critic 网络的参数. 而对应的 Actor 网络的梯度则计算为

$$\nabla_{\theta_a^i}J(\theta_a^i)=E_{a_i\sim\pi_i(s_t^i)}[\nabla_{\theta_a^i}\log\pi_i(a_i\mid s_t^i)Q(s_t^i,a_t^i,x_t^i)] \tag{4}$$

其中 θ_a^i 是该 Actor 网络的参数.

2.2　Actor-Attention-Critic

在自然语言处理领域[23]，研究者常利用自注意力机制[24]计算序列信息的内部关系度，并生成新向量所需的特征取值，利用生成权重与值向量的点乘将序列信息转换到新的特征空间[25]. 具体地说，自注意力机制所需求的 *key*、*query* 及 *value* 值都从序列信息本身得到. 构建三个转换矩阵 $\boldsymbol{W}^{\boldsymbol{Q}}$、$\boldsymbol{W}^{\boldsymbol{K}}$、$\boldsymbol{W}^{\boldsymbol{V}}$，其中 $\boldsymbol{W}^{\boldsymbol{Q}}$ 与序列输入 x 相乘得到 *query* 矩阵 $\boldsymbol{Q}=x\times\boldsymbol{W}^{\boldsymbol{Q}}$，$\boldsymbol{W}^{\boldsymbol{K}}$ 与所处理数据 y 相乘计算相似度，得到 *key* 矩阵 $\boldsymbol{K}=y\times\boldsymbol{W}^{\boldsymbol{K}}$，而 $\boldsymbol{W}^{\boldsymbol{V}}$ 则用来计算所处理数据 y 的 *value*：$\boldsymbol{V}=y\times\boldsymbol{W}^{\boldsymbol{V}}$. 随后，*query* 和 *key* 进行相似度计算，归一化后得到输入 x 对 y 的注意力 *score*：

$$score=\mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right) \tag{5}$$

其中 d_k 为 *key* 的维度. 最终 *score* 与 $\boldsymbol{V}$ 的点乘即为 y 在特征空间的表征结果.

在 MARL 领域，智能体本身的决策行为就可以看作一个动作序列[26]，同理，用自注意力机制来优化 MARL 便吸引了一些研究者的兴趣. Iqbal 等人[27]将自注意力机制引入到多智能体的集中性评估中，计算其他智能体信息的信用分配，抓住智能体交互中最有效的信息来评估策略. 在用于集中性评估的 Critic 网络中，第 i 个智能体的 Q 值是由所有智能体的状态和动作决定的，因此 Critic 网络所拟合的是 $Q(s^i,a^i,x^i)$. 在 Iqbal 等人所提出的 MAAC（Multi-Agent Actor-Attention-Critic）[27]中，x^i 被自注意力模块编码成注意力交互向量 $\bar{\boldsymbol{x}}^i$. 准确地说，第 i 个智能体的状态先被 $\boldsymbol{W}^{\boldsymbol{Q}}$ 转换成 *query* 矩阵

$$query_i=s^i\times\boldsymbol{W}^{\boldsymbol{Q}} \tag{6}$$

而 $\boldsymbol{W}^{\boldsymbol{K}}$ 和 $\boldsymbol{W}^{\boldsymbol{V}}$ 则用来编码其他智能体的状态-动作对. 第 j 个智能体的 *key* 值和 *value* 值为

$$key_j=(s^j,a^j)\times\boldsymbol{W}^{\boldsymbol{K}} \tag{7}$$

$$value_j=(s^j,a^j)\times\boldsymbol{W}^{\boldsymbol{V}} \tag{8}$$

计算第 i 个智能体在评估策略时对第 j 个的注意力

$$score_i^j=\frac{query_i\times\boldsymbol{key}_j^{\mathrm{T}}}{\sqrt{d_{key}}} \tag{9}$$

其中 d_{key} 是 key_j 的维度. $score_i^j$ 在归一化后乘上 $value_j$ 便为第 j 个智能体的交互信息 $\bar{x}_j^i$，而最后的注意力交互向量即为所有其他智能体交互信息的组合 $\bar{\boldsymbol{x}}^i=[\bar{x}_1^i,\cdots,\bar{x}_N^i]$.

MAAC 利用自注意力机制来显式地获取智能体间的交互信息，从而更加快速的优化智能体策略的学习. 这种将自注意力引入到 MARL 中的思路，很快使 MAAC 成为常见多智能体任务中的基线算法，也在大部分任务上取得了 State-of-the-art 效果.

3　多智能体策略映射转换

本工作面向同构的多智能体系统，研究智能体状态空间到行为空间的映射重组，利用自注意力机制实现策略映射稀疏化转换，为同构的 MARL 提供更为先进的策略，并加强所学策略的可解释性. 在本章节中，先讨论了“智能体”在 MARL 中的角色，随后解释了所提出的映射重组技术. 在状态空间到行为空间映射重组的基础上，分别提出了 Actor 网络和 Critic 网络的自注意力模型，并给出了所学策略的可解释性分析. 最后，给出了本文所提出的策略映射转换技术（Policy Mapping Transformation，PMT）的强化学习流程和神经网络参数更新方式.

3.1　智能体抽象化

智能体被定义为可以同环境交互的实体. 在强化学习中，这种交互是指智能体通过自身行为对环境产生影响，而环境又反馈给智能体一个奖励来更新智能体的策略. 因此，智能体的行为可以被定义为

一组动作的集合$\langle a_1,a_2,\cdots,a_{d_a}\rangle$，其中$d_a$是行为空间的维数.在一个包含$N$个智能体的同构多智能体系统中，所有智能体拥有相同的动作空间，第i个智能体的行为可以表示成$\langle a_1^i,a_2^i,\cdots,a_{d_a}^i\rangle$，而所有实体的联合行为则为

$$a=\begin{bmatrix} a_1^1 & a_2^1 & \cdots & a_{d_a}^1 \\ a_1^2 & a_2^2 & \cdots & a_{d_a}^2 \\ \vdots & \vdots & \ddots & \vdots \\ a_1^N & a_2^N & \cdots & a_{d_a}^N \end{bmatrix} \tag{10}$$

在现有的分散式MARL中，每个智能体被分配一个独立的策略，也就是说，每个实体自身被看作一个“智能体”，而实体的策略同时控制它行为空间所包含的动作.用深度神经网络来拟合策略函数，底层网络用来将实体的观测(或特征)转换到隐藏特征空间，称之为表征学习，而顶层的网络则负责根据表征学习结果决定各个动作的取值.在这种情况下，同一智能体行为空间的各个维度动作共享表征学习[28].

将智能体行为空间的各个动作通过策略统一控制，是因为研究人员假定智能体行为空间中各个维度动作具有强相关性，在共用表征学习模块时，能够产生符合强相关性的隐藏特征.事实上，在目前的多智能体应用中，大部分实体行为空间中的不同动作确实是相关的.例如，在自动驾驶中，驾驶器需要的方向与加速度往往需要配合来达到避障的目的；在群体仓储机器人中，机器人不同轮子的转速决定了机器人的前进方向.但是并非在所有多智能体系统中，实体的不同维度动作都具有这些强相关性，典型的如多机器人搬运，单个机器人的方向需要考虑搬运物偏移方向，而机器人的速度则需要与其他机器人一致.在这种情况下，单个实体的不同维度动作是弱相关的，而在使用一个策略网络来控制这些动作时，策略网络可以被看成多目标回归问题，而弱相关的动作对表征学习的要求不一，使得策略网络整体难以收敛，策略不够先进.

3.2 策略映射重组

考虑到上一小节所提到的情况，本工作旨在设计映射重组的方法，弱化不同维度动作之间的弱相关对策略网络表征学习的影响，稳定策略的收敛，并在加强所学策略先进性的同时加强可解释性.对于包含N个智能体的同构多智能体系统，智能体的状态空间维度为d_s，动作空间维度为d_a，用$(s_1^i,s_2^i,\cdots,s_{d_s}^i)$表示第$i$个智能体的状态，$(a_1^i,a_2^i,\cdots,a_{d_a}^i)$表示第$i$个智能体动作组成的行为，而智能体的联合状态为$S$.当实体的行为空间由离散动作组成时，$d_a$为单次行动所需要的离散动作数目.在传统的MARL算法中，策略分配给每个实体.从映射的角度看，分配给智能体的策略是从联合状态空间到智能体行为空间的映射$\pi_i:S\rightarrow(A^1,\cdots,A^{d_a})$.将所有智能体的所有动作视为$N\times d_a$个优化个体来看，其他工作中的方法将控制单个智能体行为的一组动作$(a_1^i,a_2^i,\cdots,a_{d_a}^N)$视为联合优化集群，用一个策略来推理它们.用$\boldsymbol{h}=f(S;\theta_r)$表示表征学习过程，$\boldsymbol{h}$为这个策略表征学习的结果，即为联合状态的特征向量，而$(a_1^i,a_2^i,\cdots,a_{d_a}^N)=g(\boldsymbol{h};\theta_d)$表示该策略的顶层推理网络，用于决定策略网络的输出.此时策略可以表示为$\pi=g(f(S;\theta_r);\theta_d)$，用多任务强化学习[29]的角度去理解该策略网络，每个动作被视为一个独立的优化任务，而这些任务之间存在一致的总体目标$E[R]$，它们通过共有的策略网络来实现协同行为.在这种情况下，不同策略需要学到的协同是智能体间的协同，而CTDE机制中的集中性评估则将协同过程放在Critic网络中，因此从动作集群上看，智能体间的协同要比智能体行为空间不同动作的协同更难训练，或者协同度更低.

而在本工作中，为了使这$N\times d_a$个动作的协同更容易被训练，我们并不为每个实体分配一个策略，而是构建与实体不一致的抽象智能体.抽象智能体的数目为d_a个，每个抽象智能体代表着不同实体同一维度动作的集合，对于第j个智能体来说，其行为空间为$\overline{A}_j=(A_1^j,A_2^j,\cdots,A_N^j)=(A^j)^N$，其中$A_i^j$是第$i$个智能体行为空间的第$j$个维度.此时，抽象智能体的联合动作则成了实体联合动作a的转置：

$$\overline{a}=\boldsymbol{a}^{\mathrm{T}}=\begin{bmatrix} a_1^1 & a_1^2 & \cdots & a_1^N \\ a_2^1 & a_2^2 & \cdots & a_2^N \\ \vdots & \vdots & \ddots & \vdots \\ a_{d_a}^1 & a_{d_a}^2 & \cdots & a_{d_a}^N \end{bmatrix} \tag{11}$$

随后，我们为每个抽象智能体分配一个策略网络，而在这个策略网络中，由于智能体是同构的，每个抽象智能体行为空间的各个动作也位于同一空间，拥有同样的边界与表征学习需求.这意味着在抽象智能体的策略网络中，无论这些动作相不相关，表征学习都不会对个别动作产生偏好，而此时策略网络可以不再是多目标回归问题，而是并行的单目标回归问题.

抽象智能体与实体的关系如图1所示.而本工作打破了实体与策略间的关系，将策略构建成从联

合状态空间到不同智能体行为空间同一动作维度的映射，此时对于所构建给抽象智能体的策略可以表示为 $\bar{\pi}_j: S \rightarrow (A^j)^N$. 此时的策略 $\bar{\pi}=g(f(S;\theta_r);\theta_d)$ 根据联合状态，从行为空间的某一维推理出一组动作序列，这组序列中的动作来自行为空间同一维度，拥有相同的上下界，对表征学习 $f(S;\theta_r)$ 有一致的要求. 因此我们的方法中，策略本身不再是一个以多目标联合优化为主的多任务学习模型，而是面向行为空间同一维度的动作序列输出问题. 由于此时策略网络输出中各个动作对表征学习的要求都是相同的，因此 $f(S;\theta_r)$ 对隐藏特征向量的编码不再担心会产生偏好，这从理论层面减轻了策略网络的负担. 再从协同的角度来看，我们用 Actor-Critic 网络来分离强化学习的策略执行和集中性评估，引入 CTDE 机制，第 i 个抽象智能体的 Actor 网络依据联合观测来推理自身的行为 $(a_i^1, a_i^2, \cdots, a_i^N)$，而集中性的 Critic 网络则根据联合观测与抽象智能体行为来计算每个行为的 Q 值. 使用 CTDE 机制来训练策略映射重组后的智能体，每个抽象实体收集多个智能体的信息，但策略网络仍只是这个抽象智能体自身的动作(即不同实体行为空间同一纬度)，不同抽象智能体的策略是分散的，对应 CTDE 中的 decentralized execution(分散执行). 而评估网络中则输入所有抽象智能体的动作，对应 CTDE 中的 centralized training(集中训练). 此时同一维度动作的协同在策略网络内部学习，而集中性 Critic 则去捕捉不同维度动作的协同. 相比起实体中各个动作的协同，在同构 MARL 任务中，所有实体同一维度动作的协同更加显式，且在策略网络中不会存在上一小节提到的弱关联性，这增加了训练过程中的稳定.

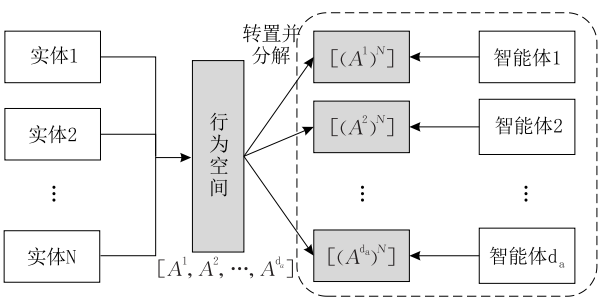


图 1　抽象智能体行为空间构建

我们所提出的策略映射重组方法，是一个“分-总-分”的过程. 首先对于环境中的多个实体，我们将其看作一个整体，每个实体都是整体中的一个部分. 随后，我们将每个实体行为空间的同一维度(动作)看作一个子系统，而分配的策略就是来控制这些子系统的. 在这个工程中，“抽象”智能体是对子系统的“抽象化”，而非对具体控制动作的抽象化. 换句话说，策略输出的动作仍为具体的动作. 而一个实体完整的行为，需要从不同的策略中整合得到. 总的来说，为抽象智能体分配策略，即为对多智能体系统中，由行为空间同一维度所组成的子系统分配策略，仍然具有物理意义.

3.3　自注意力模块

在上一小节中我们提到，所设计的策略映射重组方法能够将单个策略网络的多目标联合优化问题转化成了序列处理问题，这还存在了两个可预见的优势：可以使用面向序列本身的联合编码；编码-反编码的可解释性更强. 在本小节中，我们详细说明了本工作中的自注意力模块，并说明了在引入自注意力模块后产生的可解释性策略.

自注意力机制是从某种目标出发，去关注序列中的部分细节，实现对序列数据的转换. 抽象智能体的策略网络可以看作从联合状态空间到不同实体同一动作维度的映射，而自注意力模块用来稀疏化这种映射. 我们首先对联合状态空间进行与行为空间相似的编码操作. 对于所有实体在 t 时刻的联合状态信息 $S_t=[(s_1^1,\cdots,s_{d_s}^1),(s_1^2,\cdots,s_{d_s}^2),\cdots,(s_1^N,\cdots,s_{d_s}^N)]$，通过转置操作得到以不同实体同一维度状态所组成的集合 $\boldsymbol{S}_t^{\mathrm{T}}=[(s_1^1,\cdots,s_1^N),(s_2^1,\cdots,s_2^N),\cdots,(s_{d_s}^1,\cdots,s_{d_s}^N)]$，随后对行为空间每一个维度里不同实体的状态集合进行编码，如图 2 所示得到隐藏状态序列 $[h_t^1,h_t^2,\cdots,h_t^{d_s}]$，这组序列则作为自注意力模块的处理对象. 其中的编码器则由神经网络组成，通过可微的自注意力模块反向传播梯度进行更新.

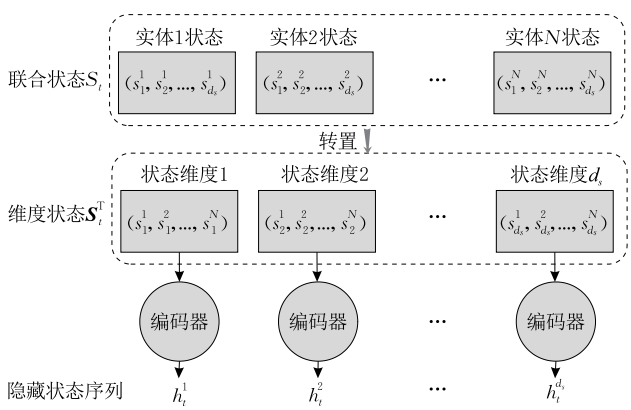


图 2　实体联合状态处理流程

在 Actor 网络中构建自注意力模块将隐藏状态序列 $[h_t^1,h_t^2,\cdots,h_t^{d_s}]$ 进一步编码. 对于每个抽象智能体的 Actor 网络，需考虑如何获得自注意力中的 key，$query$ 和 $value$. 其中 key 代表元素的键值，需

从序列数据自身编码得到. 构建 d_s 个 $\boldsymbol{W}^{\boldsymbol{K}}$ 矩阵，在时刻 t，分别将隐藏状态序列中的数据编码

$$key_j = h_t^j \times \boldsymbol{W}_j^{\boldsymbol{K}} \tag{12}$$

而 *query* 则是自注意力中的选择器. 对于第 i 个抽象智能体的策略来说，根据马尔科夫过程，t 时刻的状态只依赖于 $t-1$ 时刻的状态和行为，因此本工作中将 $t-1$ 时刻 Actor 网络的输出 $\bar{a}_{i,t-1}=[a_{i,t-1}^1, a_{i,t-1}^2, \cdots, a_{i,t-1}^N]$ 与 $\boldsymbol{W}^{\boldsymbol{Q}}$ 矩阵相乘得到第 i 个 Actor 网络注意力模块中的 *query* 值

$$query_i = \bar{a}_{i,t-1} \times \boldsymbol{W}^{\boldsymbol{Q}} \tag{13}$$

这样设计的另一个考虑是，Actor 网络中的自注意力模块需要智能体自主的从环境中选取值得关注的数据，而 $\bar{a}_{t-1}$ 包含了 $t-1$ 时实体行为的语义，在连续决策中，有益于获取到单步信息难以捕捉到的数据. 在强化学习中，智能体遵循马尔科夫决策过程，S_t 仅与 (S_{t-1}, a_{t-1}) 有关. 因此在策略网络不变时，$\bar{a}_{t-1}$ 包含了状态转移过程 $S_t \leftarrow (S_{t-1}, \bar{a}_{t-1})$ 的部分语义信息，我们利用这种语义信息来作为自注意力模块的选择器，能够隐式地捕获 S_t 在时间序列上的动态变化. 此外，$\bar{a}_{t-1}$ 本身是策略网络对于 S_{t-1} 的输出，因此其本身所包含的决策信息密度要远高于 S_{t-1}，作用于策略网络中的自注意力模块时，较小的数据维度和极高的数据密度能更快地收敛选择器网络 $\boldsymbol{W}^{\boldsymbol{Q}}$. 而 *value* 则是由 d_s 个 $\boldsymbol{W}^{\boldsymbol{V}}$ 矩阵将隐藏状态序列转换成语义信息得到，序列中第 j 个元素的 *value* 为

$$value_j = h_t^j \times \boldsymbol{W}_j^{\boldsymbol{V}} \tag{14}$$

随后计算隐藏状态序列中每个元素的 *score*，由选择器 *query* 与键值 *key* 相乘后归一化得到，用等式表示为

$$score = \mathrm{softmax}\left(\frac{query_i \times \boldsymbol{key}^{\mathrm{T}}}{\sqrt{d_{key}}}\right) \tag{15}$$

其中 $\boldsymbol{key}^{\mathrm{T}}$ 为 $[key_1, key_2, \cdots, key_{d_s}]$ 的转置，而 d_{key} 为 *key* 中元素的维度. 所得到的 *score* 即为该隐藏状态序列的注意力值，并与 *value* 进行点乘操作，得到注意力模块的输出 $(g_t^1, g_t^2, \cdots, g_t^{d_s})$，其中

$$g_t^j = score_j \cdot value_j \tag{16}$$

图 3 展示了 Actor 网络中的注意力模块，为了防止过拟合，本工作引入多头注意力机制，即构造多个注意力模块，作为集成模型来避免整体策略网络的过拟合. 如图 4 所示，每个注意力模块输出 $(g_t^1, g_t^2, \cdots, g_t^{d_s})$ 后，便将所有注意力模块的输出共同输入到用于决策的全连接层中，输出 t 时刻第 i 个抽象智能体的动作 $\bar{a}_{i,t}=[a_{i,t}^1, a_{i,t}^2, \cdots, a_{i,t}^N]$. 在所有 Actor 网络都输出抽象智能体的动作之后，实体的联合动作即为所有抽象智能体动作矩阵的转置：$[\bar{a}_{1,t}, \bar{a}_{2,t}, \cdots, \bar{a}_{d_a,t}]^{\mathrm{T}}$.

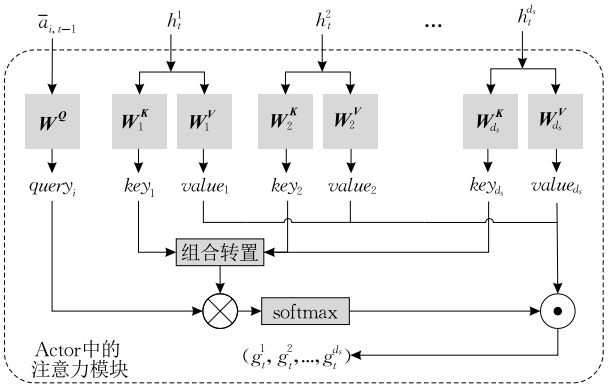


图 3 Actor 网络中的注意力模块

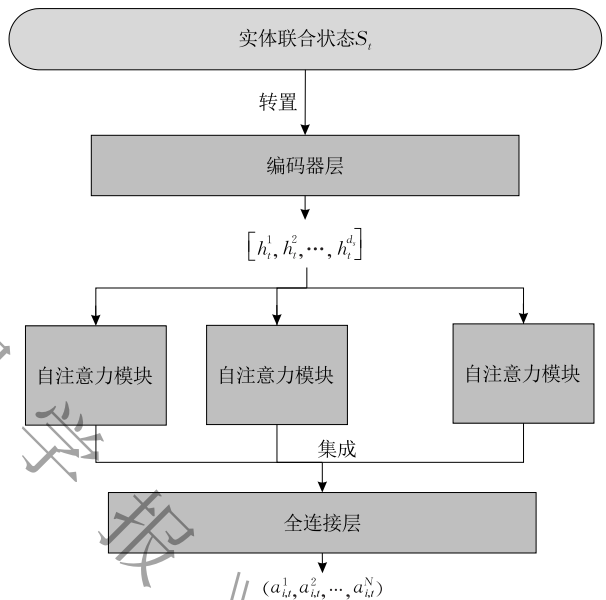


图 4 带有多头注意力的 Actor 网络

Critic 网络用来计算抽象智能体策略的 Q 值，本文利用 CTDE 机制来训练抽象智能体，因此 Critic 网络的输入是联合状态及每个抽象智能体的行为. 在集中式的 Critic 网络中，其他智能体的行为被视为环境的一部分，因此我们用自注意力模块来获取其它智能体对当前智能体在评估过程中的影响. 与 Actor 网络类似，引入多头注意力网络来编码输入信息. 在 Critic 中，注意力模块并不是用来编码联合状态，而是用来编码每个抽象智能体的行为. 这是由于在 Actor 网络中每个抽象智能体的行为都是根据联合状态决定的，因此在评估 Q 值时，其他抽象智能体的行为成了造成环境不平稳的因素，需要用自注意力机制去处理. Critic 网络的注意力模块如图 5 所示，其中第 j 个抽象智能体行为的键值 key_j 由一个转换矩阵和 $\bar{a}_{j,t}$ 相乘得到

$$key_j = \bar{a}_{j,t} \times \boldsymbol{W}_j^{\boldsymbol{K}} \tag{17}$$

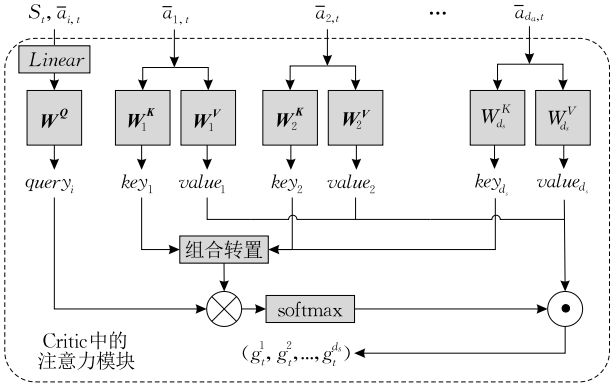


图 5　Critic 网络中的注意力模块

同样地，对应的 $value$ 值也由

$$value_j=\bar{a}_{j,t}\times\boldsymbol{W}_j^{\mathrm{V}} \tag{18}$$

计算. 而对于 Critic 网络中自注意力模块的选择器，则由联合状态以及当前智能体的行为共同编码得到

$$query_i=Linear(S_t,\bar{a}_{i,t})\times\boldsymbol{W}^{\boldsymbol{Q}} \tag{19}$$

其中 $Linear$ 是指用一个全连接层将 S_t 和 $\bar{a}_{i,t}$ 整合，方便后续处理. 第 i 个抽象智能体的 Critic 网络可以用图 6 表示，其输入是联合状态及抽象智能体的联合行为，输出是第 i 个智能体的 Q 值：$Q(\bar{a}_{i,t},S_t,x_{i,t})$，其中 $x_{i,t}=(\bar{a}_{1,t},\bar{a}_{2,t},\cdots,\bar{a}_{d_a,t})/\bar{a}_{i,t}$.

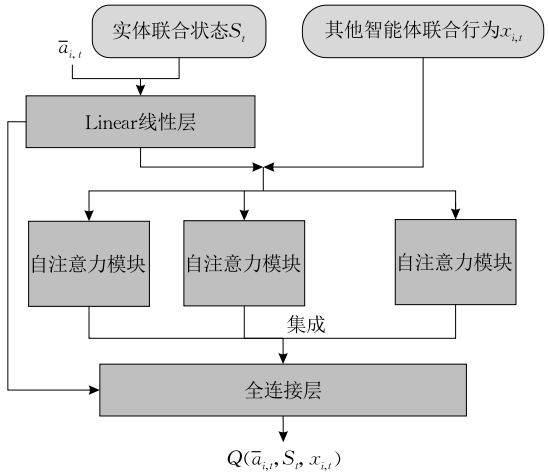


图 6　带有多头注意力的 Critic 网络

3.4　可解释性

在实体的策略映射重组得到抽象智能体后，每个 Actor 网络拟合的都是抽象智能体的策略. 在 Actor 网络中，输出是同构实体的行为空间同一维度动作的集合，因此对 Actor 网络底层的表征学习模块拥有相同的要求，Actor 网络输出空间每个维度都对表征学习的结果具有相同的置信度. 在这种情况下，自注意力模块中的 $score$ 可以视为所对应状态信息对该次决策的重要性. 对于联合状态 S_t，经过转置后得到状态维度 $(s_1^1,s_1^2,\cdots,s_1^N),\cdots,(s_{d_s}^1,s_{d_s}^2,\cdots,s_{d_s}^N)$，而自注意力模块中的 $(score_1,\cdots,score_{d_s})$ 则为对每个状态维度语义信息的掩码. 用 m 表示多头注意力网络中自注意力模块的个数，$(score_1^j,\cdots,score_{d_s}^j)$ 为单次决策中，第 j 个自注意力模块的掩码结果，则可以将所有自注意力模块的掩码结果整合后得到该次决策行为的解释，其中

$$\begin{aligned}w^i&=(w_1^i,w_2^i,\cdots,w_{d_s}^i)\\&=\mathrm{softmax}\Big(\sum_m score_1^m,\cdots,\sum_m score_{d_s}^m\Big)\end{aligned} \tag{20}$$

即可表示状态空间中各个维度对实体行为空间第 i 个维度在该次决策中的重要性.

此外，实体行为空间中各个动作的相关性也可以通过 $(w^1,w^2,\cdots,w^{d_a})$ 来显示的表达. 具体地说，我们计算 w^i 与 w^j 的 KL 散度来观察行为空间第 i 个维度和第 j 个维度在此次决策时的相关性：

$$D_{\mathrm{KL}}(w^i\|w^j)=\sum_{k=1}^{d_s}w_k^i\log\Big(\frac{w_k^i}{w_k^j}\Big) \tag{21}$$

其中，$D_{\mathrm{KL}}(w^i\|w^j)$ 越低，说明行为空间第 i 个维度和第 j 个维度具有在此次决策中有强耦合. $D_{\mathrm{KL}}(w^i\|w^j)$ 越高，则意味着这两个动作相关性越低.

本工作中通过自注意力机制计算行为空间各维度对于实体动作的重要性，是策略层面的解释. 这意味着，所学策略的解释可以与其他方法结合，以增强或验证对策略的说明.

3.5　训练和更新

在训练所构建的抽象智能体的策略时，由于 Actor 网络中的自注意力模块需要上一时刻的动作，因此在更新上与其他方法稍有区别. 在 t 时刻，Actor 网络根据联合状态 S_t 及抽象智能体的联合动作 $\bar{a}_{t-1}$ 计算此时抽象智能体的联合动作 $\bar{a}_t$，随后 $\bar{a}_t$ 转置得到实体的联合动作 a_t 并在环境中执行. 环境反馈一个奖励 r_t，并且联合状态变化为 S_{t+1}，此时用于更新网络参数的经验回放池存储经验 $\langle S_t,S_{t+1},\bar{a}_{t-1},\bar{a}_t,r_t\rangle$. 与经典的多智能体强化学习方法相比，所提出的 PMT 方法在存储经验是需要一个额外的元素 $\bar{a}_{t-1}$，这是由于在 3.3 小节，Actor 网络中的注意力模块需要根据上一时间步的行为 $\bar{a}_{t-1}$ 来得到语义选择器 $query$. 当经验回放池拥有足够的经验后便开始更新抽象智能体的 Actor 网络和 Critic 网络. 对于 Critic 网络，其目的是拟合抽象智能体的 Q 值. 对于第 i 个抽象智能体，其 Q 值可以由当前的奖励加上后续期望奖励近似得到，而其后续期望奖励则可以通过 Critic 网络输出拟合，因此 Critic 网络的损失函数为

$$L(\theta_c^i)=\frac{1}{2}[Q(\bar{a}_t^i,S_t,x_{i,t})-r_t^i-\gamma Q(\bar{a}_{t+1}^i,S_{t+1},x_{i,t+1})|_{\bar{a}_{t+1}^i\sim\pi_i(S_{t+1},\bar{a}_t^i)}]^2 \quad (22)$$

对于 Actor 网络,其策略梯度通过

$$\nabla_{\theta_a^i}J(\theta_a^i)=E_{\bar{a}_i\sim\pi_i(S_t,\bar{a}_{i,t-1})}[\nabla_{\theta_a^i}\log\pi_i(\bar{a}_i|S_t,\bar{a}_{i,t-1})\cdot Q(\bar{a}_i,S_t,x_{i,t})] \quad (23)$$

得到,引入确定性策略梯度[9],梯度计算等式可以简化为

$$\nabla_{\theta_a^i}J(\theta_a^i)=E_{S,\bar{a}\sim D}[\nabla_{\theta_a^i}\log\pi_i(\bar{a}_{i,t}|S_t,\bar{a}_{i,t-1})\cdot Q(\bar{a}_i,S_t,x_{i,t})|_{\bar{a}_i=\pi_i(S_t,\bar{a}_{i,t-1})}] \quad (24)$$

其中 D 为经验池. 利用优势函数稳定训练,可以通过计算该动作对其他动作的优势来代替该动作的 Q 值[30]. 首先计算状态值

$$V(S_t,x_{i,t})=E_{\bar{a}_i\sim\pi_i(S_t,\bar{a}_{i,t-1})}[Q(\bar{a}_i,S_t,x_{i,t})] \quad (25)$$

则 $\bar{a}_{i,t}$ 对于其他动作的优势 δ 为

$$\delta=Q(\bar{a}_{i,t},S_t,x_{i,t})-V(S_t,x_{i,t}) \quad (26)$$

此时 Actor 网络的梯度可以改写成

$$\nabla_{\theta_a^i}J(\theta_a^i)=E_{S,\bar{a}\sim D}[\nabla_{\theta_a^i}\log\pi_i(\bar{a}_{i,t}|S_t,\bar{a}_{i,t-1})\delta] \quad (27)$$

在主流的 CTDE 机制中,现有的方法都是通过集中性评估来捕获实体间的耦合,而实体行为空间不同动作维度的耦合则是在策略网络中隐式生成. 在这种情况下,单个实体动作维度的耦合更容易学习,而不同实体间的协同行为则难以得到. 尽管诸如 MAAC 和 COMA 通过对集中性评估网络(Critic)的一些改进去优化实体间的协同,所带来的提升仍旧有限. 而在本文提出的 PMT 中,抽象智能体的策略网络将不同实体行为空间同一维度的耦合隐式生成,同时集中性评估网络去学习不同动作维度的协同. 算法 1 为我们所提出的 PMT 模型的伪代码. 这一思路与其他工作不同的是,不同实体间的协同更容易学习(策略网络),而评估网络将实体之间的耦合转为行为空间内各个维度之间的耦合. 其优点在于:策略网络输出个维度对于策略网络的表征模块需求是一致的,使得策略网络更容易收敛且稳定,因此我们的 PMT 总是能学习到更为先进的策略(集中回报情况下);不同动作维度之间的耦合不在局限于单个实体,而是作用于所有实体,能够学习到传统方法无法获取的整体信息.

算法 1. 策略映射转换 PMT 强化学习算法.

1. MARL 任务包含 N 个同构实体,实体行为空间维度 d_a,状态空间维度 d_s,构建 d_a 个抽象智能体
2. 构建 d_a 个 Actor 网络与 d_a 个 Critic 网络,并在网络中加入多头自注意力模块,$t=0$,初始化抽象智能体先前联合动作 $\bar{a}_{t-1}$
3. While Training:
4. 获取实体联合状态 S_t
5. For $i=1:d_a$ Do
6. $\bar{a}_{i,t}=\pi_i(S_t,\bar{a}_{i,t-1})$ //π_i为第 i 个 Actor
7. 获取实体联合动作 $a_t=[\bar{a}_{1,t},\cdots,\bar{a}_{d_a,t}]^{\mathrm{T}}$
8. 执行 a_t,环境反馈 r_t
9. 存储经验$\langle S_t,S_{t+1},\bar{a}_{t-1},\bar{a}_t,r_t\rangle$
10. If update Do
11. For $i=1:d_a$ Do
12. 根据等式(22)计算 $L(\theta_c^i)$
13. 通过最小化损失 $L(\theta_c^i)$更新 θ_c^i
14. $V(S_t,x_{i,t})=E_{\bar{a}_i\sim\pi_i(S_t,\bar{a}_{i,t-1})}[Q(\bar{a}_i,S_t,x_{i,t})]$
15. $\delta=Q(\bar{a}_{i,t},S_t,x_{i,t})-V(S_t,x_{i,t})$
16. 根据等式(27)计算梯度 $\nabla_{\theta_a^i}J(\theta_a^i)$
17. $\theta_a^i\leftarrow\theta_a^i+\alpha\nabla_{\theta_a^i}J(\theta_a^i)$
18. $t=t+1$

4 实验结果与分析

在本节中,我们通过几个不同场景下的仿真实验来验证了所提出的 PMT 方法的可行性,并与其他的 MARL 方法进行了比较. 首先我们给出了实验所用到的仿真场景以及相关设置,随后详细介绍了用于对比的基线算法. 我们对实验的结果进行了详细的分析,并通过两个消融实验进一步证实了该方法设计的合理性.

4.1 实验设置

在本文中,我们在 MPE(Multi-Agent Particle Environment)仿真平台[9]上测试了所提出的 PMT 方法,并与几个常见的 MARL 基线进行比较. 实验所选择的多智能体任务有 Simple Spread、Collect Treasure 和 Multi-Push. 相关代码已开源至 https://github.com/LiJingchen1212/PolicyMappingTransformerMARL.git.

Simple Spread. 这个场景中包含多个实体和相同数目的地标,实体需要尽快移动到地标且避免相互之间的碰撞. 由于一个目的地同时只能容纳一个实体,因此实体间必须形成协同行为才能分配好地标,在避免碰撞的同时尽快赶往地标. 在这个环境中,环境的整体奖励与所有实体与距其最近的地标距离之和负相关,并且每次发生碰撞都会受到 -1 的惩罚.

Collect Treasure. 在这个场景中,多个实体需要不断地收集地图上的资源. 资源在地图中每隔一段时间随机的产生,实体需要在规定时间内收集尽可能多的资源. 在这个环境中实体同样不允许彼此碰撞,每收集到一个资源,实体的奖励增加 10,否则奖

励减去 0.1. 每次碰撞都会使实体的奖励减 5. 在这个场景中，我们为每个智能体设立独立的奖惩函数. 在 PMT 算法中，抽象智能体的奖励为所有智能体奖励的平均值.

Multi-push. 在这个环境中，所有实体需要将一个障碍物移动到地标位置. 障碍物的体积远大于实体，且并不会自己移动. 障碍物会因为与实体的碰撞而发生移动，移动方向为碰撞位置的反方向，每次碰撞产生的移动距离为 1 个单位. 在这个环境中，实体可以彼此碰撞，且整体奖励与障碍物到地标的距离负相关.

在所有三个任务中，实体的状态空间为自身的位置与速度，以及其他实体（包含障碍物和地标）的位置，而实体的动作空间则包含移动角度和速度. 我们使用 8 个独立的线程同时训练，一共训练 50 000 个回合，每个回合包含 25steps. 经验回放池的最大容量为 10^6，每次更新采集 1024 个经验，更新间隔为 100steps. 此外，学习率 α 设为 0.001，折扣因子 γ 为 0.99，Actor 和 Critic 网络里的多头注意力中各包含 4 个自注意力模块. 在 Actor 网络中，隐藏层包含 128 个节点，策略以高斯分布的形式输出. Critic 网络中包含一层对输入进行编码的隐藏层，拥有 128 个节点.

4.2　基线方法

本文所提出的 PMT 方法主要针对于同构的 MARL 任务，采用 CTDE 机制来训练多个策略网络，因此 MADDPG、MAAC（Multi-Agent Actor-Attention-Critic）、COMA（COunterfactual Multi-Agent）、DDPG（Deep Deterministic Policy Gradient）以及 VDN（Value-Decomposition Networks）算法被选作基线方法，用来评判 PMT 算法的性能.

MADDPG. 作为首次提出 CTDE 机制的方法，MADDPG 已经成为了 MARL 领域中最为著名的基线之一[9]. 在本文中所提出的 PMT 算法也采用了 CTDE 机制，因此与 MADDPG 进行比较，可以直观地判断 PMT 算法的优劣.

MAAC. MAAC 算法将所有智能体的状态-动作对视为一个序列，在集中性的 Critic 网络中构建一个自注意力模块，捕获智能体在不同情况下的协同关系[26]. 作为 MADDPG 的改进，MAAC 已经在多个 MARL 场景中实现了最好的结果. 与 MAAC 进行对比能够体现 PMT 算法的先进性.

COMA. COMA 算法使用反事实基线来解决智能体的信用分配问题，用中心化的 Critic 网络来评估智能体的策略[11]. COMA 算法拥有和 MAAC 相同的目的，但采取的解决方法不同. 作为最适合集中性奖惩的 MARL 方法，COMA 算法能够形成先进协同行为，因此也作为本文的基线之一.

DDPG. 在本文中，我们用集中学习的 DDPG 算法作为基线方法之一. 在 DDPG 算法中，所有实体被看作一个整体，它们的行为通过一个策略网络共同输出. 同样的，一个集中性的 Critic 网络被用来输出联合动作的 Q 值. 用集中性 DDPG 算法作为基线的目的是调查 PMT 算法与集中学习算法的表现差异.

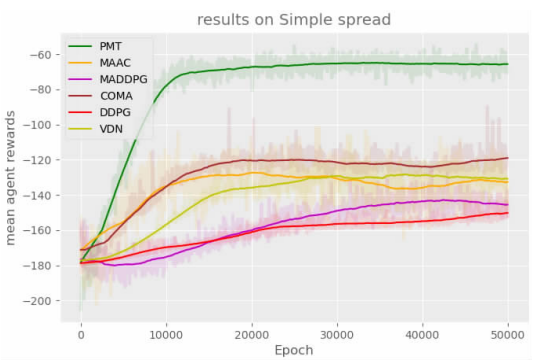
VDN. VDN 算法采用值函数分解方法来解决多智能体协作时的信用分配问题. 将所有智能体的独立 Q 值之和作为全局 Q 值，并在评估后反向传播给每个智能体，VDN 算法能够隐式地学习各个智能体独立的评估网络[31]. 在本文中，由于所提出的 PMT 算法基于集中性评估，我们使用完全中心化学习的 VDN 模型作为基线来比较 PMT 算法的性能.

在本文的实验中，我们引入目标网络机制[32]来加速智能体的策略学习. 在基线方法和 PMT 算法的训练过程中，都由目标网络来计算下一时刻的 Q 值，而目标网络每 100 步更新一次.

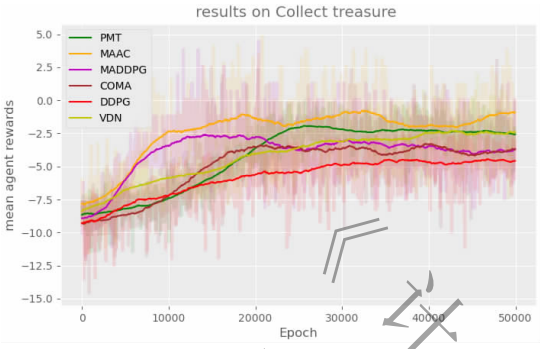
4.3　结果与分析

首先我们在场景中实体数目较少的情况下进行实验对比，其中 Simple Spread 和 Collect Treasure 场景中各含有 3 个智能体，Multi-Push 场景中包含 2 个智能体. 在智能体数目较小的情况下进行实验旨在调查 PMT 方法的可行性，并观察是否能学到比基线算法更为先进的策略.

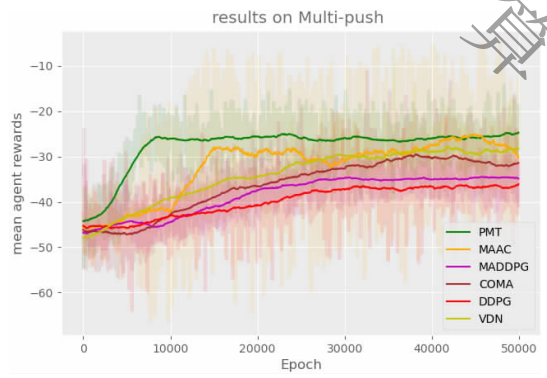
我们在每个实验中进行了 50 000 回合的训练，由 8 个线程一同完成，每 50 回合记录一次智能体的平均回报. 实验结果如图 7 所示，从 Simple Spread 上的结果我们可以得知本文所提出的 PMT 算法能够比其他基线方法学到更为先进的策略，智能体的平均回报大于 -70，而 MAAC、COMA 及 VDN 在五万回合后的平均回报不超过 -110. MADDPG 则表现更差，平均回报仅为 -150. 而集中性 DDPG 则在 50 000 回合后仍处于策略的摸索阶段. 这是由于集中学习算法中，策略网络根据联合状态输出联合行为，过多的输入/输出维度使得策略网络需要极多的经验才能到达梯度曲面的平坦区. 从训练曲线来看，PMT 所学得到更为先进的策略并不需求过多的



(a)



(b)



(c)

图 7 在三个场景中的训练结果

训练次数. 在训练初始阶段(0～10 000 回合),PMT 算法便能使智能体迅速达到策略曲面的平坦区,而其他算法不仅无法学习到能够匹敌 PMT 的策略,并且需要更多的训练次数才能收敛. 然而,从图 7(b)中可以看到,当环境所给予的奖励不再集中,每个实体都有自己独立的奖惩函数时,PMT 算法的表现并不如 MAAC. 在五万次训练回合结束后,PMT 在 Collect Treasure 场景中只取得了－2.3 左右的得分,而 MAAC 的策略更为先进,达到了－1.0. 这是由于在我们对实体的策略映射进行重组后,分散的奖励无法直观地评价抽象智能体策略的好坏,当使用平均奖励来做为整体奖励时,策略网络丢失了奖惩函数自身固有的部分信息,因此阻碍了 PMT 的策略学习过程. 尽管如此,PMT 算法仍优于 COMA、MADDPG 和 DDPG,与 VDN 相差无几,但对应的策略搜寻过程则稍微漫长. 而在 Multi-Push 场景中,PMT 算法再一次展现出了在集中奖励场景下的优异能力. 不仅比 MAAC 更快达到策略曲面的平坦区,且学习到的策略更为平稳和先进,所得到的结果与 Simple Spread 中的一致.

随后我们提高智能体的数量,在不同设置下调查了三个场景的训练结果,为我们的分析提供更有力的支持. 在所有三个场景,分别在智能体数目为 5,7 的情况下通过 PMT 和基线算法进行实验,并记录十次实验结果的平均值与标准差. 实验结果如表 1 所示,在集中奖励的两个场景中(Simple Spread 和 Multi-Push),我们所提出的 PMT 算法均实现了最好的结果. 而在分散奖励的场景中(Collect Treasure),PMT 算法的表现比 MAAC 稍差,但结果却更为稳定. 在三个智能体数目不同的任务上所获得的标准差都远低于 MAAC. 而与以值分解为核心的 VDN 相比,尽管训练结果方差较大,但所学到的策略仍优于 VDN. 这与图 7(b)中的结果一致. 在分散奖励的场景中,每个实体都持有自己的奖励函数. 而在 PMT 算法中,每个策略网络不再与实体一一对应,而抽象智能体的奖励又为实体奖励的平均值,因此分散的奖励会使得 Critic 网络输出的 Q 值带有偏差. VDN 为每个智能体都分配独立的 Critic 网络,尽管这些网络通过总体的 Q 值评估去隐式更新,但

表 1 不同智能体数目下的实验结果

场景	平均回报						标准差					
	PMT	MAAC	MADDPG	COMA	DDPG	VDN	PMT	MAAC	MADDPG	COMA	DDPG	VDN
Simple Spread(3)	**－67.1**	－132.5	－145.9	－119.3	－153.8	－128.4	**±1.40**	±5.80	±5.70	±4.50	±3.60	±4.20
Simple Spread(5)	**－57.1**	－92.4	－128.4	－93.5	－142.6	－114.3	**±1.80**	±3.30	±3.40	±3.10	±2.80	±2.40
Simple Spread(7)	**－55.6**	－127.9	－108.3	－112.6	－133.0	－97.2	**±2.50**	±4.70	±4.30	±5.20	±3.20	±3.10
Collect Treasure(3)	－2.2	**－1.1**	－3.8	－3.7	－5.1	－2.3	±0.13	±0.24	±0.17	±0.16	±0.21	**±0.12**
Collect Treasure(5)	－2.4	**－1.7**	－3.5	－3.3	－4.7	－2.6	±0.25	±0.31	±0.23	±0.29	±0.25	**±0.21**
Collect Treasure(7)	**－2.8**	**－2.8**	－4.1	－3.9	－4.5	－3.1	**±0.15**	±0.23	±0.17	±0.27	±0.19	±0.17
Multi-Push(2)	**－24.6**	－29.9	－35.7	－32.5	－37.3	－28.2	**±1.20**	±1.86	±3.17	±2.93	±2.83	±2.67
Multi-Push(5)	**－26.7**	－31.3	－38.6	－35.9	－35.6	－33.5	±2.58	±2.76	±3.99	±2.74	±2.96	**±2.52**
Multi-Push(7)	**－25.5**	－29.3	－37.0	－34.1	－34.3	－28.4	**±1.66**	±2.09	±3.44	±2.21	±3.07	±2.21

仍允许分散的奖励在每个智能体的策略中实现无偏估计，因此 VDN 在 Collect Treasure 中得到了最小的结果方差. 这个现象更进一步地验证了我们的结论：PMT 算法在集中奖励的多智能体环境中能帮助智能体学习到更为先进的策略，且依靠自注意力机制对策略映射的稀疏化能使得训练更为稳定.

PMT 对于策略的优化更多体现在从实体状态空间到行为空间的映射上. 抛开传统的“实体即为智能体”这一思路，PMT 算法能将实体之间的耦合转为行为空间内各个维度之间的耦合. 在同构的 MARL 任务中，这种耦合更为稳定，并不会因为实体的独立性而丢失，因此容易学习到更为先进的策略. 此外，在 PMT 的策略网络中，网络输出的是同构多智能体行为空间的同一维度，策略网络的表征模块对每个输出都是统一且公平的. 在强化学习过程中，经验的获取具有随机性，因此传统 MARL 方法在行为空间各个动作关联性不强的时候，表征学习容易对个别动作产生偏好，导致较大的训练波动和结果的方差. 而 PMT 算法则通过策略映射重组避免了这一点，稳定了策略的训练. 从训练曲线上来看，所提出的自注意力模块进一步将状态空间到行为空间的映射稀疏化，减少决策无关状态对于智能体行为的影响，因此 PMT 能够使策略网络更快地收敛，这是其他 MARL 方法所缺失的. 在得到这部分的仿真结果后，我们再一次思考 MAAC、MADDPG、COMA 和 VDN 等其他算法的缺点. 不难发现，这些算法大都将研究重点放在对策略的无偏估计及其优化上，忽视了深度强化学习本身固有的一些特性：策略网络本质是状态空间到行为空间的非线性映射. 而本工作是首次从策略映射层面对 MARL 进行研究的工作. 仿真实验结果也说明，从策略映射层面对 MARL 重新思考是很有必要的.

表 2 为 PMT 算法和基线算法在不同场景下训练五万个回合所需要的训练时间. 在我们的实验中，由 8 个线程同时进行训练. 可以看到，所提出的 PMT 算法单次训练所需要的时间更长，这是由于引入注意力机制后，策略网络和评估网络所需要的浮点运算量增加，单次策略执行或评估所需的时间变长. MAAC 算法在评估网络中加入了注意力机制，因此训练时间同样较长. 但随着智能体数目增多，MAAC 所需的执行时间反超 PMT. 这是因为智能体数目增多时，PMT 算法中抽象智能体的数目并未增加，仅仅只增加了抽象智能体抽象动作的维度，因此训练时间增加并不明显. 其他算法如 MADDPG、COMA、DDPG 和 VDN 则会随着智能体数目增加需要更多的训练时间. 尽管引入了自注意力机制使得 Actor 网络和 Critic 网络变得复杂，我们所提出的 PMT 算法并不会造成过多的训练代价. 在智能体数目较少时，PMT 算法所需的训练时间不超过基线算法平均训练时间的 20%，在智能体数目较多时，训练时间同比不超过 15%.

表 2　三种场景下多线程训练所需时间

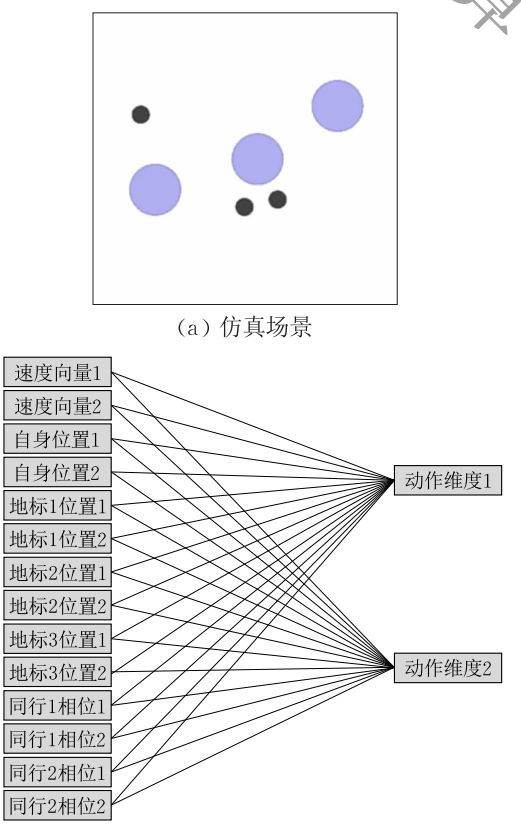
场景	训练时间					
	PMT	MAAC	MADDPG	COMA	DDPG	VDN
Simple Spread(3)	**13h34m**	12h52m	9h43m	10h15m	9h30m	11h31m
Simple Spread(5)	**14h16m**	13h44m	9h57m	11h06m	10h32m	12h57m
Simple Spread(7)	14h22m	**14h27m**	10h28m	13h19m	12h02m	13h39m
Collect Treasure(3)	**15h07m**	13h21m	11h33m	12h13m	10h38m	12h33m
Collect Treasure(5)	**15h29m**	14h28m	11h59m	12h52m	11h05m	13h46m
Collect Treasure(7)	15h44m	**15h49m**	12h46m	13h16m	11h17m	14h11m
Multi-Push(2)	**10h33m**	10h00m	8h11m	8h50m	7h49m	9h05m
Multi-Push(5)	**11h14m**	10h52m	8h32m	9h07m	8h06m	9h28m
Multi-Push(7)	**11h27m**	11h26m	8h49m	9h23m	8h21m	10h01m

不仅如此，所提出的 PMT 算法能够通过自注意力模块的输出对所学到的策略做出说明，为智能体行为赋予一定的可解释性. 在联合状态空间重组之后，自注意力模块为每个状态维度计算一个权值，这个权值即表示了该状态维度对于这次决策的重要性. 因此，可以通过统计了各个状态维度的平均注意力值，来调查不同状态维度对于 PMT 算法所学策略的重要性.

如图 8 所示，我们以 Simple Spread 场景中的结果为例，分析 PMT 所学策略的可解释性. 图 8(a)为仿真场景，(b)为对应的注意力权重图. 我们用表示映射关系的连接线表示平均注意力值，即式(20)中的 ω_i，其中深色的线表示高注意力值，浅色的线表示低注意力值. 可以看到，两个动作维度对于实体的自身位置都有着较高的关注度. 不同的是，动作维度 1 更加关注状态空间的实体自身速度 2，而动作维度 2

则对自身速度 1 有更高的注意力. 这是由于实体的动作直接决定自身的速度,来改变实体的行为,且动作维度 1 控制速度向量 2,动作维度 2 控制速度向量 1,因此在决策时,动作维度 1 则需要根据速度向量 2 做出合适的反应,同样动作维度 2 的值则更多依赖于速度向量 1. 三个地标的相对位置则拥有较为相似的关注度,这是因为在决策过程中,一个 Actor 网络控制所有实体行为空间同一维度的动作. 当每个实体有了明确的目标之后,三个地标都会被充分的考虑. 而同行的相对位置则对于两个动作都具有极低的注意力,这是由于在以联合状态作为输入的策略网络中,实体的自身位置已经给予了策略网络足够多的位置信息来决定动作,而相对位置作为冗余的状态信息,则在训练过程中逐渐被自注意力模块稀疏化,以减少对决策过程的干扰. 随后,我们用训练后的策略分别在三类场景中执行整个回合,记录 w^i 在一次任务完成过程中的值,并计算对应的平均 KL 散度:

$$\overline{D}=\frac{1}{T}\sum_{t=0}^{T}D_{\mathrm{KL}}(w^i\|w^j) \tag{28}$$



(a) 仿真场景

(b) 注意力权重图

图 8 Simple spread 场景中的平均注意力示意图

两个动作维度在三种场景下的结果如表 3 所示. 可以观察到,当环境中存在实体较少时,平均 KL 散度 $\overline{D}$ 较小,行为空间中两个维度关联性较高. 而当实体数目增多,在三个场景中,平均 KL 散度都有所增加. 例如,在只有三个实体的 Simple Spread 中,两个动作间的平均 KL 散度只有 0.33,而在包含 7 个实体的环境中,这个值增加到了 0.42. 这是由于我们所提出的 PMT 算法,将实体间的耦合转化为不同实体行为空间统一维度的耦合. 在智能体数目增多时,联合状态空间维度增加,各个动作对状态空间的依赖关系也随之发生变化,因此平均 KL 散度增加. 从三种场景的不同结果来看,Multi-Push 中两个动作的相关性最高. 这是因为在 Multi-Push 中,各个实体的任务都是将目标推到指定位置,不需要深层次的协同,反而对两个动作之间的耦合要求较大,因此它们的平均 KL 散度较小.

表 3 各个场景下两个动作的平均 KL 散度

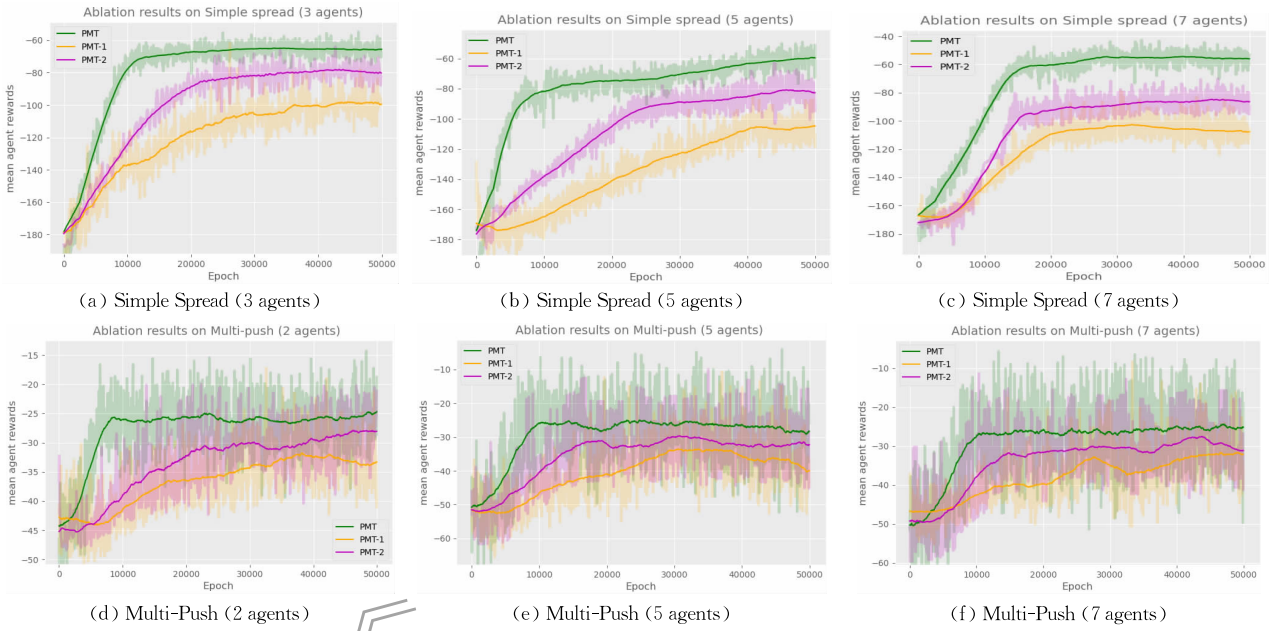
场景	平均 KL 散度 $\overline{D}$
Simple Spread(3)	0.33092
Simple Spread(5)	0.37740
Simple Spread(7)	0.42591
Collect Treasure(3)	0.33426
Collect Treasure(5)	0.36266
Collect Treasure(7)	0.38592
Multi-Push(2)	0.28439
Multi-Push(5)	0.31056
Multi-Push(7)	0.32280

4.4 消融实验

在前面的实验中,我们通过比较 PMT 算法和算法能够学习到更为先进的策略,且具有更加稳定的训练过程. 在这一小节中,我们通过几个消融实验,来进一步探究 PMT 算法,讨论策略先进性与训练稳定性的原因.

本文所提出的 PMT 算法主要包含两个关键步骤. 首先是对从状态空间到行为空间的策略映射进行重组,将不同智能体行为空间的同一动作维度组成的集合看作一个抽象智能体. 其次是对 actor 网络与 critic 网络的自注意力模块,用来编码且稀疏化重组后的策略映射. 因此,在本小节中,我们设置两个控制组,第一个控制组不使用抽象智能体,直接将自注意力模块引入到具体智能体的 actor 和 critic 网络中,用联合状态来分别控制每个智能体的行为. 第二个控制组则不使用自注意力模块,仅仅在策略映射重组后,直接对抽象智能体进行训练. 我们用 PMT-1 表示第一个控制组,第二个控制组用 PMT-2 表示.

在本文的消融实验中,我们只选取了 Simple Spread 和 Multi-Push 两个拥有集中奖励的环境,分别使用 PMT 算法和两个控制组训练. 训练结果如图 9 所示.



(a) Simple Spread (3 agents)　(b) Simple Spread (5 agents)　(c) Simple Spread (7 agents)

(d) Multi-Push (2 agents)　(e) Multi-Push (5 agents)　(f) Multi-Push (7 agents)

图 9　消融实验结果((a),(b),(c)为 Simple Spread 场景实验结果;(d),(e),(f)为 Multi-Push 场景实验结果)

从图 9 中可以得知,当只引入本文所提出的抽象智能体时(PMT-1),所得到的结果远远低于 PMT 算法,且训练中的平均回报波动较大. 当只引入本文所提出的自注意力模块时(PMT-2),智能体虽然能够学习到比基线算法和 PMT-1 更为先进的策略,但最终的实验结果仍与 PMT 算法有较大的差距. 在我们的设计中,抽象智能体将一个多目标回归问题近似成一个序列化的单目标回归问题,这种设计旨在弱化智能体间的耦合,加强行为空间中不同动作的耦合. 因此 PMT-1 能够表现得比大多数基线算法(MADDP, COMA, VDN)好,但稍弱于 MAAC. 而本工作中为 actor 网络和 critic 网络分别提出的自注意力模块则进一步加强了这种序列化单目标回归问题的编码,因此两者的结合,PMT 算法,能够实现极其优秀的表现. 但仅仅引入了自注意力模块的 PMT-2 则提升并不显著.

从消融实验的结果还能得知,当智能体数目增多时,PMT 算法的波动逐渐变大. 这是由于智能体数目的变多使得抽象智能体的 actor 网络输出维度变大,而抽象智能体的数目却没有变化. 因此对于单个网络的训练则变得困难,训练曲线时产生的波动增多. 尽管如此,当 PMT 算法达到策略曲面的平坦区时,仍能保持所学策略的稳定. 从 PMT-2 的表现可以看到,这种优势来自于自注意力模块对策略映射的稀疏化. 在进入到策略曲面的平坦区时,PMT-2 在大多数场景中都比 PMT-1 更为稳定.

表 4 为消融实验重复十次后的结果. 可以观察到,PMT-1 的结果标准差远远大于 PMT 和 PMT-2. PMT-1 没有使用策略映射重组来构建抽象智能体,只是在策略网络和评估网络中加入了本文提到的自注意力模块. PMT-2 则不实用自注意力模块,仅重组了多智能体系统的策略映射. PMT-1 的标准差与表 1 中的其他 MARL 算法近似,而 PMT-2 的标准差与 PMT 相差甚微. 这种现象说明了在 PMT 算法中,策略映射重组是使得训练结果稳定的直接因素. 在策略映射重组后,策略网络的表征模块对下游动作输出产生的偏好较低,且受强化学习训练过程随机性影响较小,因此训练结果更加稳定,这也与我们在 4.3 小节中的分析一致.

表 4　不同智能体数目下的消融实验结果

场景	平均回报			标准差		
	PMT	PMT-1	PMT-2	PMT	PMT-1	PMT-2
Simple Spread(3)	**−67.1**	−100.5	−79.8	**±1.40**	±4.90	±1.80
Simple Spread(5)	**−57.1**	−107.4	−83.2	**±1.80**	±4.60	±2.10
Simple Spread(7)	**−55.6**	−108.5	−86.6	±2.50	±4.70	**±2.40**
Multi-Push(2)	**−24.6**	−33.4	−28.1	**±1.20**	±2.10	±1.53
Multi-Push(5)	**−26.7**	−40.0	−31.7	±2.58	±2.75	**±2.39**
Multi-Push(7)	**−25.5**	−32.6	−31.9	**±1.66**	±3.42	±1.84

4.5　分析总结

从对比实验和消融实验的结果来看,PMT 算法有很明显的优点. 首先,PMT 算法能帮到同构多智能体系统学习到更为先进的策略. 尽管这些策略不与实体一一对应,但通过策略映射重组,实体之间的耦合被转化成行为空间中不同动作的组合,而这种耦合更容易通过自注意力模块学习. 其次,PMT 算

法所学到的策略更容易收敛，训练所需经验较少.这是由于在策略映射重组后的策略网络中，输出的是不同实体行为空间的同一维度，策略网络的表征学习对这些输出不带有偏好，且输出拥有同样的边界.因此策略网络所代表的策略曲面更为平滑，且受强化学习训练过程随机性的影响较小，从而使得训练曲线稳定，训练结果更可靠.最后，PMT 算法拥有一定的可解释性，能够通过状态空间每个维度对各个动作的注意力值来解释所学到的策略.此外，对于不同动作所得到的注意力分布之间的差异，也能够解释行为空间各个动作之间的关联性.

PMT 算法仍有一些不足.在分散奖励情况下，由于评估网络无法取得对策略的无偏估计，PMT 算法的表现并不如 MAAC，但仍然优于 MADDPG、COMA、DDPG 及 VDN.总的来说，PMT 算法很适合于集中奖励的同构多智能体强化学习任务.

5 结 论

本文探讨了多智能体强化学习中智能体的策略映射，以重组智能体动作集群的方式将智能体行为空间重新划分，构建抽象智能体来把智能体之间的耦合转化为智能体行为空间中不同动作的耦合，并在此基础上为 actor 网络和 critic 网络分别设计对应的自注意力模块，编码并且稀疏化重组后的策略映射，为同构的多智能体系统学习到更为先进且稳定的策略.

所提出的 PMT 算法为多智能体强化学习领域提供了一种新思路，既强化学习模型中的智能体并不一定是被控实体本身.这种思路打破了实体与策略之间的一一对应关系，并且将抽象智能体的策略近似成一个序列输出网络，利用注意力机制进一步优化.我们通过与基线算法的对比实验，证明了所提出的 PMT 算法的优越性.在 CTDE 机制下，PMT 算法能在集中奖励时更快到达策略曲面的平坦区，且所学到的策略无论在先进性还是稳定性上都远优于基线算法，在 Simple Spread 和 Multi-Push 任务上领先约 20%.多次重复实验得到的结果显示，PMT 算法的鲁棒性也远高于基线算法，十次重复实验的标准差比基线算法平均低 50%.

下一步，我们将把现有的 PMT 算法拓展到异构的多智能体强化学习中，尝试通过引入隐藏状态与隐藏行为特征来重组异构多智能体的策略映射.

致 谢 感谢各位评审老师给出的宝贵意见！

参 考 文 献

[1] Xu Jin, Liu Quan, Zhang Zong-Zhang, et al. Asynchronous deep reinforcement learning with multiple gating mechanisms. Chinese Journal of Computers, 2019, 42(3): 636-653 (in Chinese)
(徐进，刘全，章宗长等. 基于多重门限机制的异步深度强化学习. 计算机学报，2019，42(3)：636-653)

[2] Rocha F M, Costa V S, Reis L P. From reinforcement learning towards artificial general intelligence//Proceedings of the 2020 World Conference on Information Systems and Technologies. Springer, Cham, 2020: 401-413

[3] Chai Lai, Zhang Ting-Ting, Dong Hui, et al. Multi-agent deep reinforcement learning algorithm based on partitioned buffer replay and multiple process interaction. Chinese Journal of Computers, 2021, 44(6): 1140-1152(in Chinese)
(柴来，张婷婷，董会等. 基于分区缓存区重放与多线程交互的多智能体深度强化学习算法. 计算机学报，2021，44(6)：1140-1152)

[4] Cui J, Liu Y, Nallanathan A. Multi-agent reinforcement learning-based resource allocation for UAV networks. IEEE Transactions on Wireless Communications, 2019, 19(2): 729-743

[5] Catacora Ocana J M, Riccio F, Capobianco R, et al. Cooperative multi-agent deep reinforcement learning in soccer domains//Proceedings of the 18th International Conference on Autonomous Agents and Multi Agent Systems. Montreal, Canada, 2019: 1865-1867

[6] Liu X, Yu J, Feng Z, et al. Multi-agent reinforcement learning for resource allocation in IoT networks with edge computing. China Communications, 2020, 17(9): 220-236

[7] Posor J E, Belzner L, Knapp A. Joint action learning for multi-agent cooperation using recurrent reinforcement learning. Digitale Welt, 2020, 4(1): 79-84

[8] Schöllig A, Alonso-Mora J, D'Andrea R. Independent vs. joint estimation in multi-agent iterative learning control//Proceedings of the 49th IEEE Conference on Decision and Control (CDC). Atlanta, USA, 2010: 6949-6954

[9] Lowe R, Wu Y, Tamar A, et al. Multi-agent actor-critic for mixed cooperative-competitive environments//Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, USA, 2017: 6382-6393

[10] Rashid T, Samvelyan M, Schroeder C, et al. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning//Proceedings of the 2018 International Conference on Machine Learning. Vienna, Austria, 2018: 4295-4304

[11] Foerster J, Farquhar G, Afouras T, et al. Counterfactual multi-agent policy gradients//Proceedings of the 2018 AAAI Conference on Artificial Intelligence. Louisiana, USA, 2018, 32(1): 2974-2982

[12] Foerster J N, Assael Y M, de Freitas N, et al. Learning to communicate with deep multi-agent reinforcement learning// Proceedings of the 30th International Conference on Neural Information Processing Systems. Barcelona, Spain, 2016: 2145-2153

[13] Devlin J, Chang M W, Lee K, et al. BERT: Pre-training of deep bidirectional transformers for language understanding// Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Minneapolis, USA, 2019: 4171-4186

[14] Jiang J, Lu Z. Learning attentional communication for multi-agent cooperation//Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal, Canada, 2018: 7265-7275

[15] Das A, Gervet T, Romoff J, et al. TarMAC: Targeted multi-agent communication//Proceedings of the International Conference on Machine Learning. Long Beach, USA, 2019: 1538-1546

[16] Omidshafiei S, Agha-Mohammadi A A, Amato C, et al. Decentralized control of partially observable Markov decision processes using belief space macro-actions//Proceedings of the 2015 IEEE International Conference on Robotics and Automation. Washington, USA, 2015: 5962-5969

[17] Zaremba W, Sutskever I, Vinyals O. Recurrent neural network regularization. arXiv preprint arXiv:1409.2329, 2014

[18] Peng P, Wen Y, Yang Y, et al. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play StarCraft combat games. arXiv preprint arXiv: 1703.10069, 2017

[19] Peng Z, Zhang L, Luo T. Multi-agent communication with attentional and recurrent message integration//Proceedings of the 2018 IEEE Symposium on Computers and Communications. Natal, Brazil, 2018: 198-203

[20] Li J, Shi H, Hwang K S. An explainable ensemble feedforward method with Gaussian convolutional filter. Knowledge-Based Systems, 2021, 225: 107103

[21] Zhang K, Yang Z, Başar T. Multi-agent reinforcement learning: A selective overview of theories and algorithms. Handbook of Reinforcement Learning and Control, 2021: 321-384

[22] Kapoor S. Multi-agent reinforcement learning: A report on challenges and approaches. arXiv preprint arXiv:1807.09427, 2018

[23] Nadkarni P M, Ohno-Machado L, Chapman W W. Natural language processing: An introduction. Journal of the American Medical Informatics Association, 2011, 18(5): 544-551

[24] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need//Proceedings of the 31st Advances in Neural Information Processing Systems. Long Beach, USA, 2017: 5998-6008

[25] Yang H, Kim J Y, Kim H, et al. Guided soft attention network for classification of breast cancer histopathology images. IEEE Transactions on Medical Imaging, 2019, 39(5): 1306-1315

[26] Liang Xing-Xing, Feng Yang-He, Ma Yang, et al. Deep multi-agent reinforcement learning: A survey. Acta Automatica Sinica, 2020, 46(12): 2537-2557(in Chinese)
(梁星星，冯旸赫，马扬等. 多 Agent 深度强化学习综述. 自动化学报，2020, 46(12): 2537-2557)

[27] Iqbal S, Sha F. Actor-attention-critic for multi-agent reinforcement learning//Proceedings of the 36th International Conference on Machine Learning. Long Beach, USA, 2019: 2961-2970

[28] Shi H, Li J, Mao J, et al. Lateral transfer learning for multiagent reinforcement learning. IEEE Transactions on Cybernetics, 2021. doi: 10.1109/TCYB.2021.3108237

[29] Hessel M, Soyer H, Espeholt L, et al. Multi-task deep reinforcement learning with PopArt//Proceedings of the 2019 AAAI Conference on Artificial Intelligence. Hawaii, USA, 2019, 33(1): 3796-3803

[30] Liu G, Li X, Sun M, et al. An advantage actor-critic algorithm with confidence exploration for open information extraction//Proceedings of the 2020 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2020: 217-225

[31] Sunehag P, Lever G, Gruslys A, et al. Value-decomposition networks for cooperative multi-agent learning based on team reward//Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems. Sao Paulo, Brazil, 2018: 2085-2087

[32] Kobayashi T, Ilboudo W E L. t-soft update of target network for deep reinforcement learning. Neural Networks, 2021, 136: 63-71

LI Jing-Chen, Ph. D. candidate. His research interest is intelligent control.

SHI Hao-Bin, Ph. D., professor. His research interests include intelligent robots and machine learning.

HWANG Kao-Shing, Ph. D., professor. His research interests include intelligent robots and machine learning.

Background

As an important field in Artificial Intelligence, Multi-Agent reinforcement learning is expected to intelligently control Multi-Agent systems. However, the existence of other agents make the Markov Decision Process for an agent incomplete. Although researchers have developed several mechanisms such as Centralized Training and Decentralized Execution to enhance MARL, the existing methods are hard to gain the optimal policies due to the difficulty of capturing the coupling among agents. Several works attempted to extract these coupling in different ways, such as communication channel, attention encoding, and centralized evaluation. However, the learned policy tends to be suboptimal, and the training process is unstable.

For these reasons, we aim at transforming the coupling among agents, using abstract agents as the mediator in the interaction between agents and environments. The policy mapping from the joint state space and joint behavior space are recombined, and the abstract agents are assigned with policies, by which the coupling among agents are transformed into that among the behavior space. This transformation makes the policy networks easy to learn coordinate, while two self-attention modules are proposed for policy networks and evaluation networks, resulting in a more stable training. Moreover, the self-attention module can reflect the importance for each state vector on the decision, by which the learned policy can be explained explicitly. The proposed method provides a new viewpoint of multi-agent reinforcement learning, and homogeneous Multi-Agent systems can learn more advanced, explainable, and stable policies by our method.

Based on the method above, we have developed several Multi-Agent reinforcement learning methods and researched the coordination of multi-agent systems. Such as Multi-Agent transfer learning, Multi-Agent reinforcement learning based on hard-attention model, and explainable feedforward neural network. Our existing achievements lay a solid foundation for this work.

This work is supported by the National Natural Science Foundation of China under Grant Nos. 61976178 and 62076202, the Open Research Projects of Zhejiang Lab (No. 2022NB0AB07), the Shaanxi Province Key Research and Development Program of China under Grant No. 2022GY-090, the CAAI-Huawei MindSpore Open Fund (No. CAAIXSJLJJ-2021-041A), and the Doctor's Scientific Research and Innovation Foundation of Northwestern Polytechnical University (No. CX2022016).