软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, 2022,33(4):1477−1500 [doi: 10.13328/j.cnki.jos.006472]

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

基于预测编码的样本自适应行动策略规划*

梁星星[1], 马 扬[1], 冯旸赫[1], 张驭龙[1,2], 张龙飞[1], 廖世江[1], 刘 忠[1]

[1](国防科技大学 系统工程学院, 湖南 长沙 410072)

[2](31002 部队)

通信作者: 冯旸赫, E-mail: fengyanghe@nudt.edu.cn; 刘忠, E-mail: liuzhong@nudt.edu.cn

摘 要: 军事行动、反恐突击等强对抗场景中, 实时信息的碎片化、不确定性对制定具有博弈优势的弹性行动方案提出了更高的要求, 研究具有自学习能力的智能行动策略规划方法已成为编队级强对抗任务的核心问题. 针对复杂场景下行动策略规划状态表征困难、数据效率低下等问题, 提出了基于预测编码的样本自适应行动策略规划方法. 利用自编码模型压缩表示任务的原始状态空间, 通过任务环境的状态转移样本, 在低维度状态空间中使用混合密度分布网络对任务环境的动态模型进行学习, 获得了表征环境动态性的预测编码; 基于预测编码展开行动策略规划研究, 利用时间差分敏感的样本自适应方法对状态评估值函数进行预测, 改善了数据效率, 加速了算法收敛. 为了验证算法的有效性, 基于全国兵棋推演大赛机机挑战赛的想定, 构建了包含大赛获奖选手操作策略的5种规则智能体, 利用消融实验验证编码方式、样本采样策略等不同因子组合对算法的影响, 并使用Elo评分机制对各个智能体进行排序; 实验结果表明: 基于预测编码的样本自适应算法——MDN-AF 得分排序最高,对战平均胜率为 71%, 其中大比分获胜局占比为 67.6%, 而且学习到了自主波次划分、补充侦察策略、“蛇形”打击策略、轰炸机靠后突袭等4种长时行动策略. 该算法框架应用于2020年全国兵棋推演大赛的智能体开发, 并获得了全国一等奖.

关键词: 行动规划; 强化学习; 兵棋推演; 预测编码; 样本自适应

中图法分类号: TP18

中文引用格式: 梁星星, 马扬, 冯旸赫, 张驭龙, 张龙飞, 廖世江, 刘忠. 基于预测编码的样本自适应行动策略规划. 软件学报, 2022, 33(4): 1477–1500. http://www.jos.org.cn/1000-9825/6472.htm

英文引用格式: Liang XX, Ma Y, Feng YH, Zhang YL, Zhang LF, Liao SJ, Liu Z. Sample Adaptive Policy Planning Based on Predictive Coding. Ruan Jian Xue Bao/Journal of Software, 2022, 33(4): 1477−1500 (in Chinese). http://www.jos.org.cn/1000-9825/6472.htm

Sample Adaptive Policy Planning Based on Predictive Coding

LIANG Xing-Xing[1], MA Yang[1], FENG Yang-He, ZHANG Yu-Long[1,2], ZHANG Long-Fei[1], LIAO Shi-Jiang[1], LIU Zhong[1]

[1](College of Systems Engineering, National University of Defense Technology, Changsha 410072, China)

[2](31002 Troops)

Abstract: With the development of intelligent warfare, the fragmentation and uncertainty of real-time information in highly competitive scenarios such as military operations and anti-terrorism assault put forward higher requirements for making flexible policy with game advantages. The research of intelligent policy learning method with self-learning ability has become the core issue of formation-level tasks. Faced with difficulties in state representation and low data utilization efficiency, a sample adaptive policy learning method is

* 基金项目: 国家自然科学基金(71701205)
梁星星、马扬同为第一作者
本文由“面向开放场景的鲁棒机器学习”专刊特约编辑陈恩红教授、李宇峰副教授、邹权教授推荐.
收稿时间: 2021-05-23; 修改时间: 2021-07-16; 采用时间: 2021-08-27; jos 在线出版时间: 2021-10-26

proposed based on predictive coding. The auto-encoder model is applied to compress the original task state space, and the predictive coding of the dynamic environment is obtained through the state transition samples of the environment combined with the autoregressive model using the mixed density distribution network, which improves the capacity of the task state representation. Temporal difference error is utilized by the predictive-coding-based sample adaptive method to predict the value function, which improves the data efficiency and accelerates the convergence of the algorithm. To verify its effectiveness, a typical air combat scenario is constructed based on the previous national wargame competition platforms, where five specially designed rule-based agents are included by the contestants. The ablation experiments are implemented to verify the influence of different factors with regard to coding strategies and sampling policies while the Elo scoring mechanism is adopted to rank the agents. Experimental results confirm that MDN-AF, the sample adaptive algorithm based on predictive coding,reaches the highest score with an average winning rate of 71%, 67.6% of which are easy wins. Moreover, it has learned four kinds of interpretable long-term strategies including autonomous wave division, supplementary reconnaissance, “snake” strike and bomber-in-the-rear formation. In addition, the agent applying this algorithm framework has won the national first prize of 2020 National Wargame Competition.

Key words: action planning; reinforcement learning; wargame; predictive coding; sample adaptive

自战争出现以来, 作战任务规划活动就一直存在. 在传统的作战任务规划中, 指挥员根据自身的作战经验和对实时态势的理解进行作战行动即时规划, 指导作战行动. 然而, 随着装备技术的不断发展, 战争形态发生了深刻的变化, 使得战场信息更加不完全、环境变化更加剧烈、对抗边界更加不确定、作战响应更加迅捷, 不依赖任务规划工具则难以制定高效的作战行动方案. 行动策略规划的主要流程为: 认知当前任务状态; 在可选行动空间中, 选择满足预定目标的动作; 到达新的任务状态, 开始新的决策循环[1]. 当前行动策略规划的研究主要方法为:

1) 经典规划方法: 规划问题由状态、目标和行动等 3 个部分组成, 状态随着行动的选择进行转换, 规划的目标为寻找从初始状态到任务目标状态的行动序列. 典型的模型为 STRIPS 系统[2]及其变种[3,4]. 经典规划方法难以对作战任务各要素进行完整建模, 所得到的行动序列无法满足各个约束;
2) 基于层次任务网络的方法: 基于知识和任务分解, 采用自顶向下的层次分解方法, 将目标状态分解至可执行的子任务后进行规划. 典型的系统有 SIPE-2[5], O-Plan[6], PASSAT[7], I-X[8]. 层次任务网络方法的主要难点为知识的获取与表示, 对设计人员具有较高的专业要求; 此外, 任务的分解需要依靠已有的行动库和方法库, 对支撑系统的要求高, 且系统构建复杂[9];
3) 案例推理方法: 对比当前任务同已有案例的相似性, 采用近似任务的行动方案进行规划. 典型的系统包含 HICAP[10], JADE[11]. 基于案例推理的规划方法仅需寻找与当前任务相似的案例[12], 求解简单, 但是难以有效地进行任务间的相似性度量, 且构建全面、高质量的案例库需要大量资源, 成本高昂[13];
4) 过程推理方法: 由信念、愿望、意图和规划这 4 个部分组成, 信念是智能体对环境和自身的知识; 愿望是智能体计划达到的目标; 意图是智能体执行行动规划的驱动; 规划是在意图的驱动下, 基于智能体的信念, 制定行动序列来达到愿望[14]. 典型的系统有 SWARMM[15], ModSAF[16], MANA[17]. 对于给定的目标, 过程推理系统可在规划库中进行推理, 并执行规划出来的行动序列, 具有较高的执行效率[18].

现有的作战行动策略规划方法在特定场景下具有较好的表现, 但以离线方式产生的行动策略难以适应任务的不确定性转移带来的波动; 其次, 在在线实时决策过程中, 注重即时的行动收益, 限制了行动规划的视野, 无法对长时回报进行有效利用; 最后, 传统规划方法严重依赖人类知识, 所获得的模型受限于设计者的水平, 限制了当前作战行动规划方法的能力上限, 难以生成超越设计者水平的行动策略.

将人工智能应用在军事领域, 旨在实现规划的高效率、动态环境下的高适应能力和行动策略的灵活性. 深度强化学习(deep reinforcement learning, DRL)结合了深度神经网络和强化学习的优势, 可以用于解决智能体在复杂高维状态空间中的感知决策问题, 在游戏、机器人、自动驾驶等领域, 深度强化学习已经取得了突破性进展[19]. 然而, 现有无模型的 DRL 仅仅利用环境奖赏, 忽略了固有的、能够提高学习效率的潜在环境信息,

无法有效地对当前状态进行高效表征, 学习效率低下, 限制了其在复杂现实生活问题中的应用, 尤其是实体数量众多、仿真推演困难的兵棋推演等问题; 其次, 在多智能体复杂环境下, 智能体的单步态势无法有效表达环境的当前状态, 使用循环神经网络能够有效表达当前态势下的真实状态; 最后, “端到端”的训练方式降低了模型构建的难度, 但提高了模型收敛的成本. 有效压缩高维态势空间, 使用压缩后的态势表征信息展开学习, 能够降低模型所需参数, 提升模型收敛速度[20].

针对这些问题, 本文提出了基于预测编码的样本自适应行动策略规划方法, 利用自回归模型对任务的原始状态空间进行压缩, 在低维度的状态空间中, 使用混合密度分布网络对任务环境的动态模型进行学习, 并将学习中的隐层信息定义为当前任务状态的预测编码, 基于预测编码展开行动策略的规划与评估, 并使用时间差分敏感的样本自适应方法动态调整训练数据对值函数更新的权重; 为了克服新增任务状态空间带来的预测编码模型漂移问题, 以软更新的参数更新方式, 固定时间间隔对预测编码模型进行更新. 在案例研究中, 本文基于全国兵棋推演大赛比赛平台[21], 针对典型空战想定, 根据大赛获奖选手的操作策略构建了 5 种规则智能体对手, 利用消融实验验证不同因子对算法的影响, 并通过 Elo 评分[22]对各个智能体的能力进行排序. 实验结果表明: 基于预测编码的样本自适应算法——MDN-AF 算法效果最好, 在 Elo 评分中排序第一, 可以击败 5 种规则智能体, 平均胜率为 71%, 成功击毁敌方核心单元、大比分获胜占比为 67.6%, 且学习到了自主波次划分、补充侦察策略、“蛇形”打击策略、轰炸机靠后突袭等 4 种长时行动策略. 在 2020 年全国兵棋推演大赛中, 基于本文所提算法开发的智能体在多轮智能体对抗中取得了全国一等奖的成绩.

代码地址: https://github.com/carrylj/fengyanghe.github.io/tree/master/resource

1 基于 DRL 的行动规划

1.1 基于MDP的行动过程建模

马尔可夫决策过程(Markov decision process, MDP)是强化学习问题的理想表达形式, 形式化地描述了 RL 中智能体同环境的交互过程[19,23], 如图 1 所示.

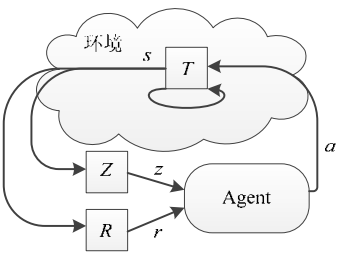


图 1 面向 DRL 的 MDP 交互过程

智能体在多个离散时间步同环境进行交互: 在时间步 t, 智能体从环境的状态空间 S 中获得状态 s_t, 根据策略 $\pi(a_t|s_t)$, 从可选动作空间 A 中选择动作 a_t, 环境根据内在动态性(收益函数 $R(s,a,s')$和状态转移函数 $T=P(s_{t+1}|s_t,a_t)$)转移到下一状态 s_{t+1}, 并返回一个标量的收益值 r_{t+1}(收益值是针对下一时刻的收益, 因而下标是 $t+1$). 当环境所处的状态为终止状态或交互达到最大时间步时, 交互结束. 在非完全信息的环境中, 智能体无法获得环境的真实状态, 决策所需要的信息 z 为真实状态的映射, 即 $z\sim Z(s)$.

在 MDP 中, 收益的累计和被称为回报, 记作 G_t, 数学表达为

$$G_t=R_{t+1}+\ldots+R_{t+i}+\ldots+R_T \tag{1}$$

其中, R_{t+i} 是 t 时刻后智能体同环境第 i 次交互获得的收益, R_T 表示终止时刻获得收益(在无终止状态的任务中, 为最大交互时刻). 在无终止状态的任务中, 使用折扣率 γ 对未来收益加权求和, 折扣累积回报表示为

$$G_t = R_{t+1} + \gamma R_{t+2} + ... + \gamma^{i-1} R_{t+i} + ... = \sum_{i=1}^{\infty} \gamma^{i-1} R_{t+i} \tag{2}$$

当$\gamma\to 0$, 则智能体注重当前收益; 当$\gamma\to 1$, 则智能体注重长远的累积收益[22]. 智能体的目标是, 最大化每个状态的累积奖赏期望值 $\max \bar{G}_t$.

1.2 基于PPO的行动策略规划

在深度强化学习中, 反向传播方法求解了策略的优化梯度, 但学习率需要随时间变化, 以确保更好的模型性能. Schulman 等人在信赖域策略优化(trust region policy optimization, TRPO)算法[24]的基础上给出了近端策略优化算法(proximal policy optimization, PPO)[25], 是当前深度强化学习最主流的算法之一. 相比 DQN[26], TRPO[24], SAC[27]等强化学习方法, PPO 方法能够有效处理稀疏奖赏下的学习任务, 对奖赏函数的设计要求较低, 计算复杂度相较 TRPO 具有明显优势, 且策略收敛能力稳定.

基于 PPO 的行动策略优化流程如图 2 所示. 蓝色数据流表示智能体同任务环境进行交互的过程, 用以生成策略学习所需数据. 新演说家(actor)网络 New_Actor 接收任务环境的状态 s, 经过正则化(normal)操作后, 获得可选行动的概率分布 $\pi(a|s)$, 并依概率采样行动 a 同环境进行交互, 按照时间关系对交互经验(s,a,r,s',d,p)进行存储. 黄色数据流为状态值函数学习过程, 根据经验存储中收集的经验$\{(s,a,r,s',d,p)\}$, 获得多步经验数据集$\{\tau_i\}$, 将状态 s 输入评论家(critic)网络——Critic 网络, 获得预测状态值 $v(s)$; 将 r,s',d 输入 Critic 网络, 获得目标状态值 $v^{-}(s)$; 之后, 将 $v^{-}(s)$和 $v(s)$做差, 获得优势值 *advantage*, 利用均方差损失函数(*mse_loss*)对 Critic 网络进行更新. 红色数据流为演说家网络——Actor 网络的学习过程, 将 s, a 分别输入新演说家网络 *New_Actor* 和旧演说家网络 *Old_Actor*, 获得新旧网络下的行动 a 的选择概率 *new_prob* 和 *old_prob*, 并计算新旧网络下的动作比率 *ratio*; 之后, 将该比率同得到的优势值 *advantage* 相乘, 获得未经裁剪 *surr* 和裁剪后 *clip surr* 的代理损失值, 通过取最小操作(min 操作符)获得最终的演说家网络损失值 *actor_loss*, 将新网络的参数备份给旧网络; 之后, 使用反向传播对新演说家网络的参数进行更新.

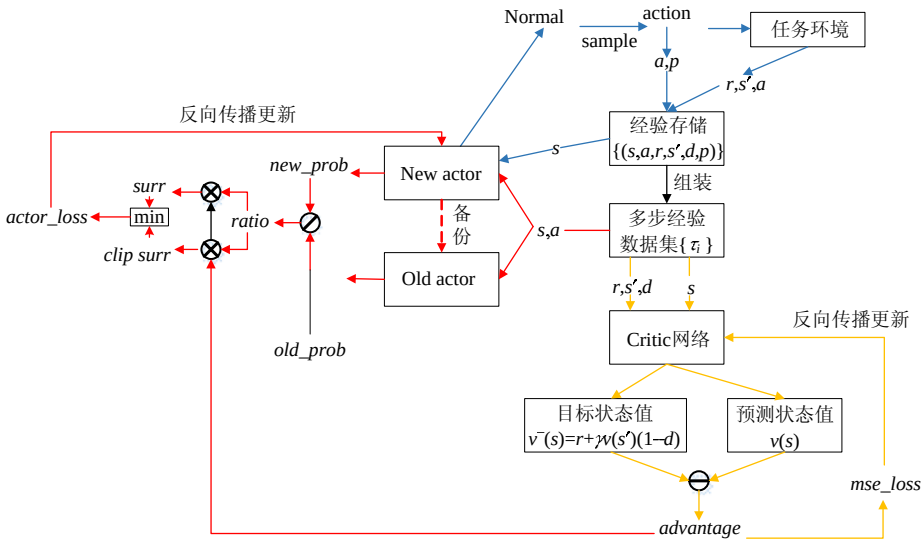


图 2 基于 PPO 的行动策略优化流程图

2 基于预测编码的样本自适应行动策略规划

针对兵棋推演等复杂环境下, 状态表征难、数据利用率低、策略收敛慢等问题, 本文提出了基于预测编码的样本自适应行动策略规划方法, 结合基于模型的深度强化学习方法, 显式地学习环境的动态性, 并将学习过程中的隐层状态编码视作当前状态的预测编码; 基于所获得的预测编码进行行动策略规划, 并使用样本自适应的方法加快状态值函数的收敛, 更好地指导行动策略学习.

2.1 基于自回归模型的状态预测编码学习

相比于其他视频类强化学习环境, 兵棋推演下的环境时空广、维度多样, 返回的数据为多源复合类型数据, 数据类型包含离散和连续值、枚举值等. 本文利用变分自编码器(variational auto-encoder, VAE)模型[28]对多源复合类型数据进行压缩, 算法如图 3 所示.

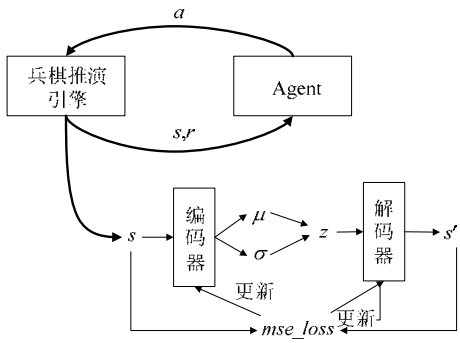


图 3 面向兵棋推演状态的 VAE 模型

智能体使用随机或者学习策略同兵棋推演引擎进行交互, 将生成的状态进行存储; 之后, 利用 VAE 模型对状态进行压缩编码. 训练中模型的损失函数设计见公式(3):

$$loss=mse\_loss(s,s')-D_{KL}(q(z|s)||p_{model}(z)) \tag{3}$$

损失函数包含 s 到 s'的复制损失 $mse\_loss(s,s')$, 以及以 KL 散度衡量的近似后验分布 $q(z|s)$和模型先验 $p_{model}(z)$间的接近损失 $D_{KL}(q(z|s)||p_{model}(z))$.

同简单的强化学习任务相比, 兵棋推演环境下的行动规划面临着奖赏稀疏性更大、决策步长更长、动作更加多样可变等问题, 因而预测编码模型相对复杂. 如图 4 所示, 以循环神经网络(rerrent neural network, RNN)[29]作为状态预测编码模型的主体架构, 将上一时刻的 RNN 隐层状态 h_{t-1}和当前时刻的 VAE 模型压缩编码 z_t作为网络的第 1 部分输入, 经过 RNN 后将获得隐层状态 h_t, 即状态预测编码, 向后传播作为下一时刻的网络输入, 向上传播进行未来状态编码预测.

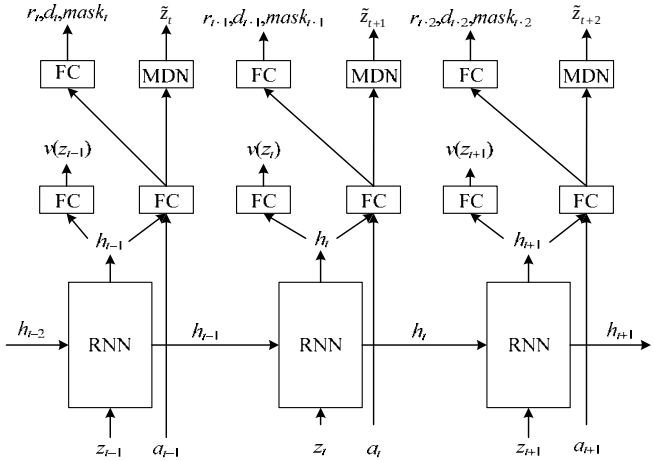


图 4 面向兵棋推演行动规划的 MDN-RNN 结构

向上传播分为两个头, 其中一个头用于预测当前压缩编码状态 z_t 下的状态值 $v(z_t)$; 另一个头同当前步的动作 a_t进行联结, 对下一时刻的收益 r_{t+1}、结束标志 d_{t+1}、可用动作 $mask_{t+1}$以及下一时刻的状态压缩编码 z_{t+1}进行预测, 其中, 使用混合密度网络(mixture density networks, MDN)[30]模型对 z_{t+1}进行预测. 根据预测值类型的不同, 这一模型包含多种不同的损失值: 状态值函数的损失 $loss_v$、奖赏值函数 r_{t+1} 的损失 $loss_r$、可用动作

$mask_{t+1}$ 的损失 $loss_{mask}$、编码 $\tilde{z}_{t+1}$ 的预测损失 $loss_z$.

基于兵棋推演行动规划的复杂强化学习环境的预测编码学习损失函数可表示为公式(4):

$$L=\beta_1 loss_v+\beta_2 loss_r+\beta_3 loss_{done}+\beta_4 loss_{mask}+\beta_5 loss_z \tag{4}$$

其中, β_1~β_5 为参数, 对各损失函数项权重进行调整, 各损失函数项的具体设计见附录 A.

2.2 基于预测编码的样本自适应行动策略规划

本节开展基于预测编码的样本自适应算法研究, 利用预测编码生成智能体的行动策略, 并通过样本自适应方法对值网络(评论家网络)进行训练, 指导策略网络(演说家网络)地学习.

(1) 基于预测编码的控制器模型

兵棋推演中的智能体可执行机动、打击等多类动作, 在每个决策时刻的可用动作空间不同, 不同类型的实体单元的动作空间也存在差异. 为了采用统一的动作空间进行全体实体单元动作表示, 本文使用 *mask* 标记对单个实体的在各状态下的可用动作空间进行标记.

为了消除不可用动作对行动概率分布的影响, 将行动概率值向量同 *mask* 向量进行对位乘法, 将不可用的动作的概率重置为 0, 对概率值非 0 的动作进行归一化后采样. 图 5 描述了基于预测编码的控制器模型. 控制器模型基于当前状态的预测编码 h_t 生成原始的动作表示 $h(a)$; 之后, 经过 *softmax* 函数操作, 生成动作的初步概率分布 $[\tilde{a}_0,...,\tilde{a}_n]$, 使用 $mask_t$ 对初步动作概率进行⊗操作, 将不可用的动作概率置为 0; 在训练中, 将概率值非 0 的动作概率求和后归一化, 获得有效动作的选择概率$[a_0,\ldots,a_n]$.

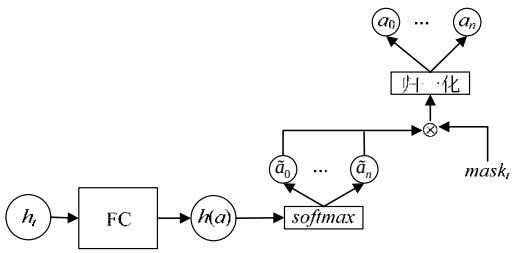


图 5 基于预测编码的控制器模型

(2) 样本自适应采样策略

为了克服稀疏奖赏导致的模型收敛困难, 每次采集一个完整的推演局作为训练样本. 本文采用离策略的方式对 AC 算法中的评论家函数进行更新, 采用经验回放[31]对已生成的样本进行存储并再次训练. 在优先经验重放机制[32]中, 使用样本的时序差分误差(TD-error, δ)衡量样本的采样权重, 采用一局样本 j 中的偏差最大的时刻 i 的 TD-error, 即 $\max(|\max\{\delta_i\}_j|,|\min\{\delta_i\}_j|)$作为样本的采样权重, 用以计算 PPO 中的经验回放选择概率.

图 6 展示了基于预测编码的自适应算法流程, 其中, 实线表示了转移数据生成机制, 虚线表示了样本自适应的行动策略规划流程.

(3) 行动策略学习损失

图 5 的控制器模型设计中, 使用 *mask* 对初步动作概率进行了硬编码. 为了使模型在动作输出概率中显式地降低对不可用动作的选择概率值, 在 PPO 算法的损失项基础上加入对不可用动作输出概率的惩罚, 使用公式(5)的计算方式对不可用的动作概率输出进行约束, 该损失项为

$$loss_{\overline{mask}}=-\frac{\sum_{i=1}^{n}(1-y_i)[y_i\log\hat{y}_i+(1-y_i)\log(1-\hat{y}_i)]}{\sum_{i=1}^{n}(1-y_i)} \tag{5}$$

其中, y_i 是真实标签, $\hat{y}_i$ 是预测标签.

综上可得, 控制器的损失函数为

$$L=\beta_6\mathbb{E}[\min(r_t(\theta)\hat{A}_t,clip(r_t(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_t]+\beta_7 H(a_t\mid s_t)+\beta_8 loss_{\overline{mask}} \tag{6}$$

其中, 第 1 项为 PPO 算法中策略更新的损失项, $r_t(\theta)$表示新旧策略的比值, $\hat{A}_t$ 表示优势函数, 本文使用 GAE 方法计算 $\hat{A}_t$; 第 2 项为策略熵正则化项, 鼓励智能体具有更强的探索性; 第 3 项为不可用动作的惩罚项, 约束智能体对不可用动作的概率估计.

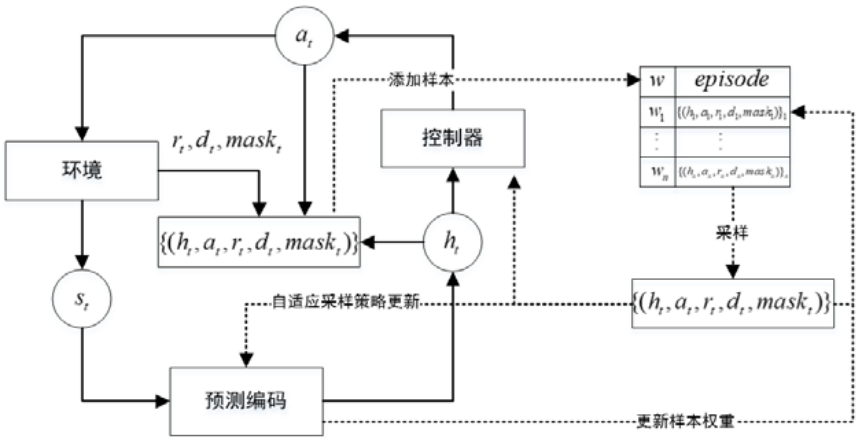


图 6 基于预测编码的自适应算法流程图

3 算法对比与分析

为了验证算法性能, 本文基于 2020 年全国兵棋推演大赛机机对抗赛想定[21]进行案例研究(见附录 B), 对状态及动作空间进行了整理(见附录 C). 为了降低智能体的无效行动探索空间, 加快智能体行动策略的收敛, 针对空战任务的具体特征以及想定的胜负判定标准, 对智能体的起始规划与有效作战区域进行有效界定(见附录 D). 本节首先对智能体学习的对比算法以及对抗智能体进行描述, 之后对训练结果进行分析.

3.1 对比算法设计

基于大赛获奖选手的操作策略, 设计了 5 种基于规则和运筹学的基线智能体. 由于存在复数个对抗对手智能体, 每次训练时, 需要从中选择一个对手. 本文通过 5 个对手智能体的两两对抗, 获得智能体 Elo 评分[22], 并对各智能体进行排序. Elo 得分是通过参与者的互相对抗, 确定各参与者的能力得分的评分方式, 计算方式如下:

$$\left.\begin{aligned} E_A &= \frac{1}{1+10^{(R_B-R_A)/400}},\\ E_B &= \frac{1}{1+10^{(R_A-R_B)/400}},\\ R'_A &= R_A + K(S_A - E_A),\\ R'_B &= R_B + K(S_B - E_B) \end{aligned}\right\} \tag{7}$$

其中, 初始分值 R_A 和 R_B 设定为 1 200 分, E_A 和 E_B 初始值为 0.5, S_A 和 S_B 为 A 和 B 的胜平负关系. 每进行完一局对战后, 更新得分 R_A 和 R_B, 继而对 E_A 和 E_B 进行更新. K 值随 R 的变化取不同值(R>1250, K=6; 1250>R>1230, K=10; 1230>R, K=16). 每一对智能体重复对抗 20 局, 5 种基线智能体的得分见表 1.

表 1 5 种基线智能体的 Elo 得分与排序

Agent 名称	得分	序值
Agent_1	1 292.49	1
Agent_2	1 213.21	2
Agent_3	1 189.13	3
Agent_4	1 114.77	5
Agent_5	1 126.10	4

在训练过程中, 将 5 种基线智能体的蓝方放入对手训练池中, 以智能体的得分序值倒数$\left[1,\frac{1}{2},\frac{1}{3},\frac{1}{5},\frac{1}{4}\right]$作为采样概率, 每次训练和测试时, 随机选择一个智能体同强化学习智能体进行对抗.

为了验证算法的有效性, 本文根据状态编码形式和有无采用样本自适应策略等设计了 6 种对比算法.

- 全连接状态表示、无样本自适应策略模型 FC-nAF: 该模型中, 利用全连接层对状态进行表示, 并利用该表示输出动作的概率和当前状态的评估值, 在训练过程中, 不使用样本自适应策略对模型进行训练;
- 全连接状态表示、有样本自适应策略模型 FC-AF: 该模型中, 利用全连接层对状态进行表示, 并利用该表示输出动作的概率和当前状态的评估值, 使用样本自适应策略对状态的评估值进行训练;
- VAE 状态表示、无样本自适应策略模型 VAE-nAF: 该模型中, 利用 VAE 层对状态进行表示, 并利用基于 VAE 的状态表示输出动作的概率和当前状态的评估值, 不使用样本自适应策略对状态的评估值进行训练;
- VAE 状态表示、有样本自适应策略模型 VAE-AF: 该模型中, 利用 VAE 层对状态进行表示, 并利用基于 VAE 的状态表示输出动作的概率和当前状态的评估值, 使用样本自适应策略对状态的评估值进行训练;
- MDN 状态表示、无样本自适应策略模型 MDN-nAF: 该模型中, 利用 MDN-RNN 层对状态进行表示, 获得预测编码, 并利用预测编码输出动作的概率和当前状态的评估值; 此外, 该模型结合当前的动作预测下一时刻的信息, 不使用样本自适应策略对状态的评估值进行训练;
- MDN 状态表示、有样本自适应策略模型 MDN-AF: 该模型中, 利用 MDN-RNN 层对状态进行表示, 获得预测编码, 并利用预测编码输出动作的概率和当前状态的评估值; 此外, 该模型结合当前的动作预测下一时刻的信息, 并使用样本自适应策略对状态的评估值进行训练.

表 2 和图 7 给出了上述 6 种对比算法在网络设计和训练架构中的异同描述, 白色模块为通用模块, 浅灰色模块表示在训练中有无采用样本自适应策略模块, 深灰色模块表示网络设计中的不同, 即有无使用 VAE 编码的编码模块和有无基于 MDN 状态表示的预测编码模块.

表 2　6 种对比算细节对比

算法名称	状态表示方法	是否采用样本自适应策略
FC-nAF	全连接层	否
FC-AF	全连接层	是
VAE-nAF	变分自编码器	否
VAE-AF	变分自编码器	是
MDN-nAF	混合密度网络	否
MDN-AF	混合密度网络	是

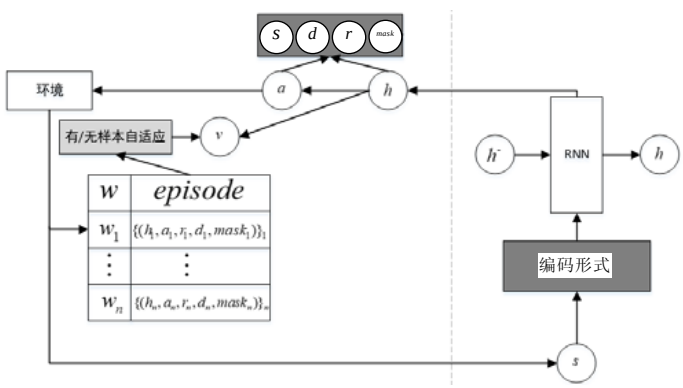


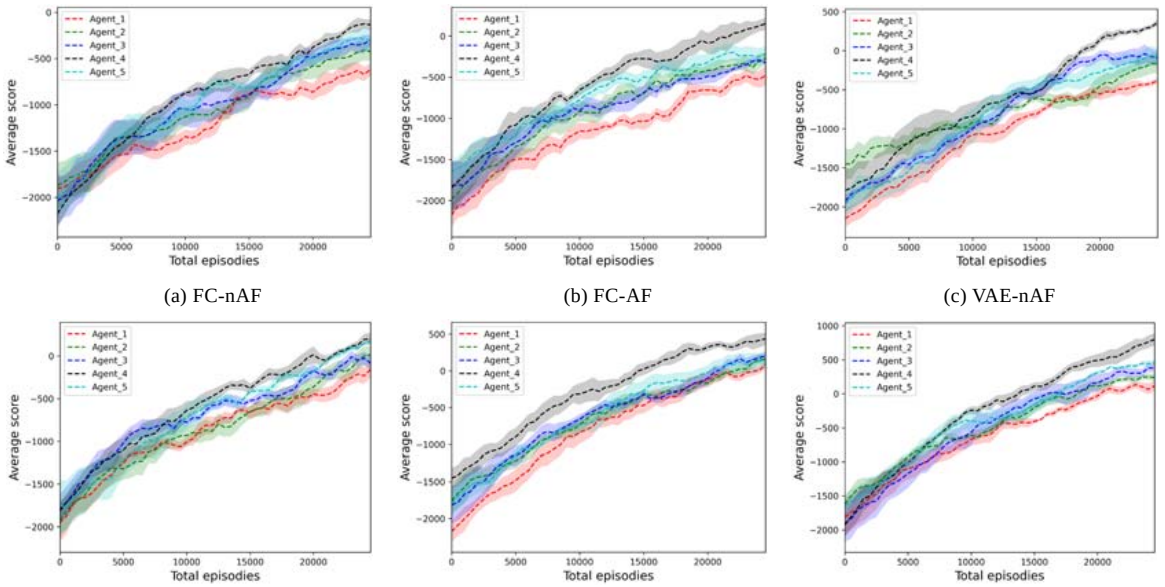
图 7　对比算法异同描述

3.2 结果分析

为了更好地展示算法性能, 本节从两个方面对结果数据进行展示: 单一学习算法智能体对抗多种基线智能体和单一基线智能体对抗多种学习算法智能体.

(1) 单一算法对抗多种基线 agent

首先展示 6 种算法分别对抗多种基线智能体的对抗结果, 如图 8 所示, 图中折线结果为平滑因子取 0.95 对 20 盘数据进行的平滑展示.



(a) FC-nAF (b) FC-AF (c) VAE-nAF

(d) VAE-AF (e) MDN-nAF (f) MDN-AF

图 8 6 种算法对抗 5 种基线智能体中的得分表现

图 8(a)展示了 FC-nAF 算法对抗 5 种基线智能体的结果. 在最终的结果中, 对抗 Agent_1 的效果最差, 其终值收敛在–700 分左右; 对抗 Agent_4 的效果最好, 终值可以达到–150 分附近; 对抗 Agent_3, Agent_5 的结果相近, 终值在–500 分附近; 对抗 Agent_2 的终值处于–600 左右. 对抗战绩排序为

Agent_4>Agent_3=Agent_5>Agent_2>Agent_1.

图 8(b)展示了 FC-AF 算法对抗 5 种基线智能体的结果. 在最终的结果中, 对抗 Agent_1 的效果最差, 其终值收敛在–600 分左右; 对抗 Agent_4 的效果最好, 终值可以达到 100 分附近; 对抗 Agent_2, Agent_3, Agent_5 的结果相近, 终值在–300 分附近, 但对抗 Agent_5 的收敛速度优于 Agent_2, Agent_3. 对抗战绩排序为

Agent_4>Agent_3=Agent_5=Agent_2>Agent_1.

图 8(c)展示了 VAE-nAF 算法对抗 5 种基线智能体的结果. 在最终的结果中, 对抗 Agent_1 的效果最差, 其终值收敛在–500 分左右; 对抗 Agent_4 的效果最好, 终值可达到 300 分附近; 对抗 Agent_2, Agent_3, Agent_5 的结果相近, 终值在–100 分附近, 但对抗 Agent_3 的收敛速度优于 Agent_2, Agent_5. 对抗战绩排序为

Agent_4>Agent_3=Agent_5=Agent_2>Agent_1.

图 8(d)展示了 VAE-AF 算法对抗 5 种基线智能体的结果. 在最终的结果中, 对抗 Agent_1 的效果最差, 其终值收敛在–300 分左右; 对抗 Agent_4 和 Agent5 的效果最好, 终值可以达到 150 分附近; 对抗 Agent_2, Agent_3 的结果相近, 终值在–100 分附近. 对抗战绩排序为: Agent_4=Agent_5>Agent_3=Agent_2>Agent_1.

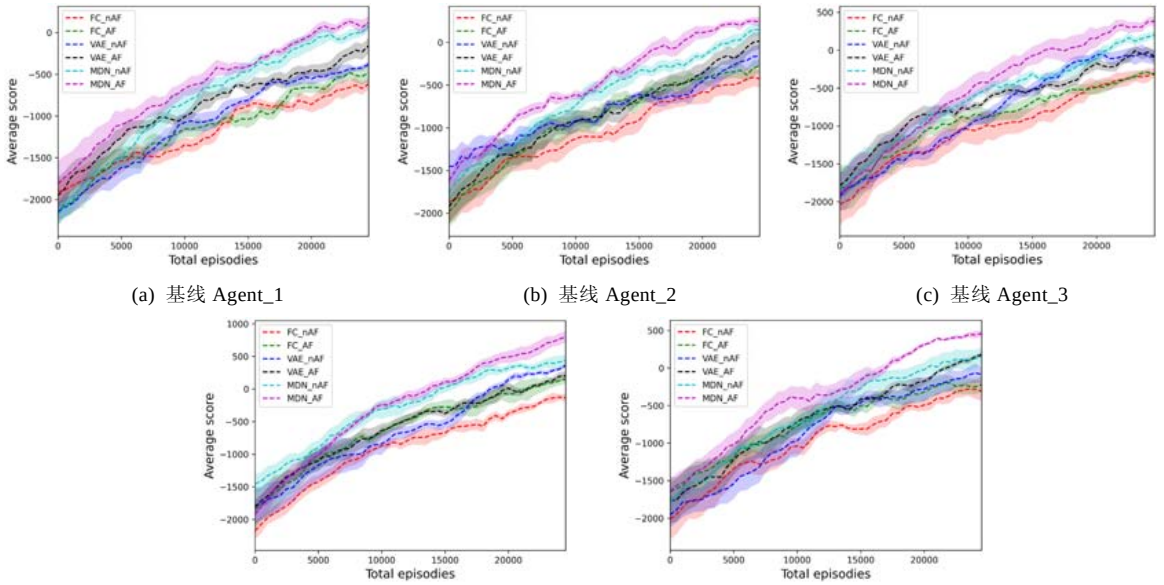
图 8(e)展示了 MDN-nAF 算法对抗 5 种基线智能体的结果. 在最终的结果中, 对抗 Agent_1 的效果最差, 其终值收敛在 0 分左右; 对抗 Agent_4 的效果最好, 终值可以达到 400 分附近; 对抗 Agent_2, Agent_3, Agent_5 的结果相近, 终值在 100 分附近. 对抗战绩排序为: Agent_4>Agent_3=Agent_5=Agent_2>Agent_1.

图 8(f)展示了 MDN-AF 算法对抗 5 种基线智能体的结果. 在最终的结果中, 对抗 Agent_1 的效果最差, 其终值收敛在 0 分左右; 对抗 Agent_4 的效果最好, 终值可以达到 600 分附近; 对抗 Agent_2, Agent_3, Agent_5 的结果处于中等水平, 水平相近, 终值在 200 分附近. 对抗战绩排序为

Agent_4>Agent_5>Agent_3>Agent_2>Agent_1.

(2) 基线智能体对抗多种学习算法智能体

上一节展示了单一学习算法对多种基线智能体的性能, 本节反向展示每个基线智能体对抗多种学习算法智能体的表现, 如图 9 所示.



(a) 基线 Agent_1 (b) 基线 Agent_2 (c) 基线 Agent_3

(d) 基线 Agent_4 (e) 基线 Agent_5

图 9 5 种基线分别对抗 6 种算法的得分表现

图 9(a)展示了基线 Agent_1 在对抗 6 种算法中的得分表现. 算法 MDN_AF 的终值为 100 分, 表现最好; 算法 MDN_nAF 的终值稳定在−100 分; 算法 VAE_AF 稳定在−300 分; 算法 VAE_nAF 稳定在−500 分; 算法 FC_AF 稳定在−600 分; 算法 FC_nAF 表现最差, 为−700 分. 算法的性能表现为 MDN_AF>MDN_nAF>VAE_AF>VAE_nAF>FC_AF>FC_nAF. 在 MDN_AF, VAE_AF 和 FC_AF 以及 MDN_nAF, VAE_nAF 和 FC_nAF 的两组对比中, MDN 类算法性能优于 VAE 和 FC 类算法, 且 FC 类算法最差; 在 MDN_AF 和 MDN_nAF、VAE_AF 和 VAE_nAF、FC_AF 和 FC_nAF 组成的 3 组对比中, AF 类算法均优于 nAF 类算法.

图 9(b)展示了基线 Agent_2 在对抗 6 种算法中的得分表现. 算法 MDN_AF 的终值为 200 分, 表现最好; 算法 MDN_nAF 的终值稳定在 100 分; 算法 VAE_AF 稳定在−100 分; 算法 VAE_nAF 稳定在−200 分; 算法 FC_AF 稳定在−400 分; 算法 FC_nAF 表现最差, 为−500 分. 算法的性能表现为 MDN_AF>MDN_nAF>VAE_AF>VAE_nAF>FC_AF>FC_nAF. 在 MDN_AF, VAE_AF 和 FC_AF 以及 MDN_nAF, VAE_nAF 和 FC_nAF 的两组对比中, MDN 类算法性能优于 VAE 和 FC 类算法, 且 FC 类算法最差; 在 MDN_AF 和 MDN_nAF、VAE_AF 和 VAE_nAF、FC_AF 和 FC_nAF 组成的 3 组对比中, AF 类算法均优于 nAF 类算法.

图 9(c)展示了基线 Agent_3 在对抗 6 种算法中的得分表现. 算法 MDN_AF 的终值为 300 分, 表现最好; 算法 MDN_nAF 的终值稳定在 100 分; 算法 VAE_AF 稳定在−200 分; 算法 VAE_nAF 稳定在−200 分; 算法 FC_AF 稳定在−400 分; 算法 FC_nAF 稳定在−400 分. 算法 FC_AF 和 FC_nAF 的性能表现最差. 算法的性能表现为 MDN_AF>MDN_nAF>VAE_AF=VAE_nAF>FC_AF=FC_nAF. 在 MDN_AF, VAE_AF 和 FC_AF 以及 MDN_nAF, VAE_nAF 和 FC_nAF 的两组对比中, MDN 类算法性能优于 VAE 和 FC 类算法, 且 FC 类算法最差; 在

MDN_AF 和 MDN_nAF、VAE_AF 和 VAE_nAF、FC_AF 和 FC_nAF 组成的 3 组对比中, MDN_AF 算法显著优于 MDN_nAF, VAE_AF 和 VAE_nAF、FC_AF 和 FC_nAF 算法的终值接近, 但 VAE_AF 算法在早期训练中提升优于 VAE_nAF, FC_AF 的收敛速度优于 FC_nAF 算法.

图 9(d)展示了基线 Agent_4 在对抗 6 种算法中的得分表现. 算法 MDN_AF 的终值为 700 分, 表现最好; 算法 MDN_nAF 的终值稳定在 400 分; 算法 VAE_nAF 稳定在 200 分; 算法 VAE_AF 稳定在 100 分; 算法 FC_AF 稳定在 100 分; 算法 FC_nAF 表现最差, 为–150 分. 算法的性能表现为 MDN_AF>MDN_nAF=VAE_nAF>VAE_AF=FC_AF>FC_nAF, MDN_nAF 和 VAE_nAF 算法得分相近, VAE_AF 和 FC_AF 算法得分相近. 在 MDN_AF, VAE_AF 和 FC_AF 以及 MDN_nAF, VAE_nAF 和 FC_nAF 的两组对比中, MDN 类算法性能优于 VAE 和 FC 类算法, 且 FC 类算法最差; 在 MDN_AF 和 MDN_nAF、VAE_AF 和 VAE_nAF、FC_AF 和 FC_nAF 组成的 3 组对比中, VAE_nAF 在早期收敛速度低于 VAE_AF 算法, AF 类算法整体上优于 nAF 类算法,.

图 9(e)展示了基线 Agent_5 在对抗 6 种算法中的得分表现. 算法 MDN_AF 的终值为 400 分, 表现最好; 算法 MDN_nAF 的终值稳定在 100 分; 算法 VAE_AF 稳定在 100 分; 算法 VAE_nAF 稳定在–100 分; 算法 FC_AF 稳定在–400 分; 算法 FC_nAF 表现最差, 为–450 分. 算法的性能表现为 MDN_AF>MDN_nAF=VAE_AF>VAE_nAF>FC_AF=FC_nAF. 在 MDN_AF, VAE_AF 和 FC_AF 以及 MDN_nAF, VAE_nAF 和 FC_nAF 的两组对比中, MDN 类算法性能优于 VAE 和 FC 类算法, 且 FC 类算法最差; 在 MDN_AF 和 MDN_nAF、VAE_AF 和 VAE_nAF、FC_AF 和 FC_nAF 组成的 3 组对比中, 虽然 FC_AF 和 FC_nAF 算法的终值相似, 但 FC_AF 收敛显著优于 FC_nAF, 因而, AF 类算法均优于 nAF 类算法.

为了进一步对各个算法进行评估, 使用 Elo 评分规则, 以表 1 作为基础分, 将 6 种算法同 5 种基线智能体进行对抗并进行分级, 获得的得分情况与分级见表 3.

表 3 6 种算法的 Elo 得分、排序与分级

Agent 名称	得分	序值	分级
MDN_AF	1 301.36	1	1
Agent_1	1 292.49	2	1
VAE_AF	1 280.55	3	2
MDN_nAF	1 277.80	4	2
VAE_nAF	1 254.46	5	3
FC_AF	1 246.65	6	3
Agent_2	1 213.21	7	4
Agent_3	1 189.13	8	4
FC_nAF	1 161.60	9	4
Agent_5	1 126.10	10	5
Agent_4	1 114.77	11	5

从表 3 中可知, 6 种算法的优劣关系为 MDN-AF>VAE-AF>MDN-nAF>VAE-nAF>FC-AF>FC-nAF. MDN-AF 处于领先的位置, 可以击败 5 种基线智能体, 水平同 Agent_1 相近, 处于第 1 级; VAE-AF 和 MDN-AF 处于第 2 级, VAE-nAF 和 FC-AF 处于第 3 级, 这两级智能体可以击败基线 Agent_2, Agent_3, Agent_4, Agent_5; FC_nAF 处于第 4 级,仅能击败基线 Agent_4 和 Agent_5.

综上分析, 可以得出如下结论.

- 在状态编码形式中, 基于 MDN-RNN 的预测编码性能优于基于 VAE 的编码性能, 优于基于 FC 的编码性能;
- 在样本自适应技术中, 结合样本自适应因子的算法性能优于无样本自适应因子的算法;
- 样本自适应因子同编码技术同等重要, 共同加速了算法了收敛.

4 长时行动策略分析

为了分析智能体的行动策略, 本文使用 MDN-AF 算法同 5 种基线智能体进行对抗, 以字典的形式记录对抗数据, 字典包含的信息为

$$\{time:\{r\_state:\{state,action\},b\_state:\{state\}\}\}.$$

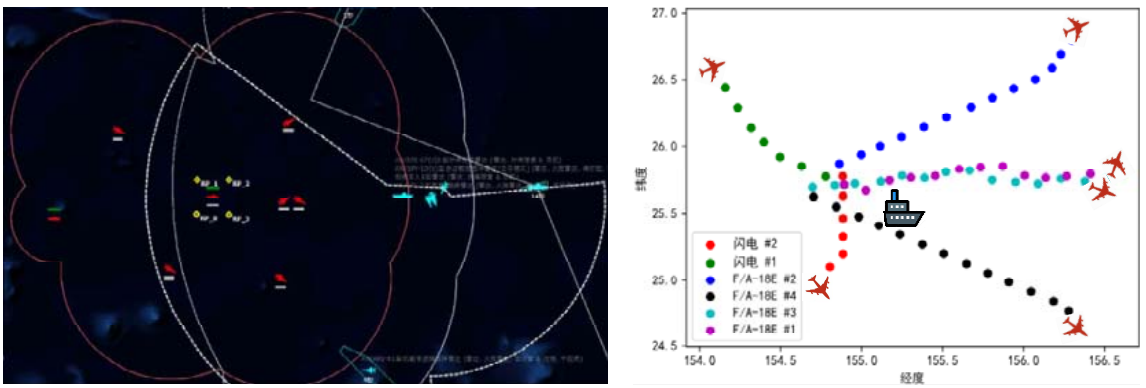
基于 MDN-AF 算法的智能体在同基线智能体对抗过程中，形成了 4 种典型的长时行动策略.

- 自主波次划分;
- 补充侦察策略;
- “蛇形”打击策略;
- 轰炸机靠后突袭.

针对上述 4 种长时行动策略，利用时空轨迹图对飞机的飞行路径进行展示，并对每一种策略产生的原因进行分析.

(1) 自主波次划分

在规则设置中，所有飞机初始部署于本方驱逐舰的后侧空中区域，在展开规划后，多机飞行轨迹出现分离，效果如图 10 所示: 两架飞机飞行至红方驱逐舰的西北侧和西南侧; 在东侧的 4 架飞机中，中间分布两架，南北两侧各分布两架，形成“1-2-1”的纵向阵型.



(a) (b)

图 10 飞机自主波次划分效果图与轨迹示意图

图 10 展示了一局比赛中，以 5 个决策步长(75s)为间隔的飞行轨迹图，飞向西侧的两架飞机为轰炸机,其主要职能为对水面舰艇进行打击，留在相对安全的西侧，能够有效地避开第一波空战对抗; 而在东侧突击的 4 架飞机为战斗机，主要职能为进行空中格斗，夺取制空权.

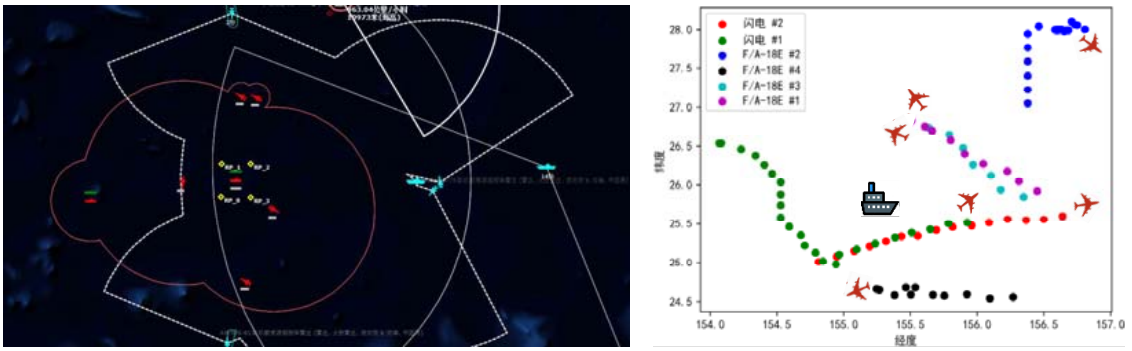
结合平台特性以及基线智能体的特征，对自主波次划分的原因进行了分析: 在想定中，每架战斗机携带 2 枚远程空空弹，6 枚近程空空弹，如果飞机聚集在一起进行战斗行动，“独狼”式的攻击队形能够有效消耗对手的弹药，降低对手的弹药效率; 在 5 种基线智能体中，飞机以编队的形式展开对抗，而学习算法中的智能体以单机控制的形式对每架飞机进行控制，分波、分批次对对手阵型进行冲击，最大限度发挥了装备的性能，有效提高了弹药的使用效率，进而提高获胜概率. 此外，在同基线 Agent_2 进行对抗的过程中发现，基线 Agent_2 采用高速逼近策略，以“闪击战”的方式，在仿真开始后的 800 s 内高速冲向红方驱逐舰对其进行打击，自主波次划分后能够有限降低此种策略对己方飞机聚集的冲击，提高在推演前期的战斗机生存率.

(2) 补充侦察策略

在对抗过程中，红方依靠驱逐舰的雷达以及飞机的自身雷达对来袭目标进行探测分析. 然而，新一代的战斗机在自身关闭雷达后，能够有效规避对手驱逐舰雷达的探测，在缺乏飞机雷达补充探测的空域实现自我隐身，达到突防的目的. 图 11(a)展示了红方飞机进行补充侦察的过程，两架红方飞机自行飞向北侧空域进行侦察. 在红方驱逐舰的各个方向均留有飞机进行空域侦察，协助驱逐舰进行空域探测.

图 11(b)展示了一局比赛中，以 5 个决策步长(75 s)为间隔的飞行轨迹图. 战斗机“F/A-18E #1”和战斗机“F/A-18E #3”在完成空中对抗后，自行飞向驱逐舰的正北侧进行空域补充侦察; 战斗机“F/A-18E #4”则在驱逐舰的南侧，由东南侧飞行正南侧，并对正南侧的来袭飞机进行侦察与对抗. 轰炸机“闪电 #1”和“闪电 #2”则

由后侧转移至前侧展开进攻，“闪电 #1”在向东侧飞行中优先补充了“闪电 #2”离开后的侦察空位，在战斗机“F/A-18E #4”前来支援侦察后离开侦察位置，转向东侧进攻.



(a) (b)

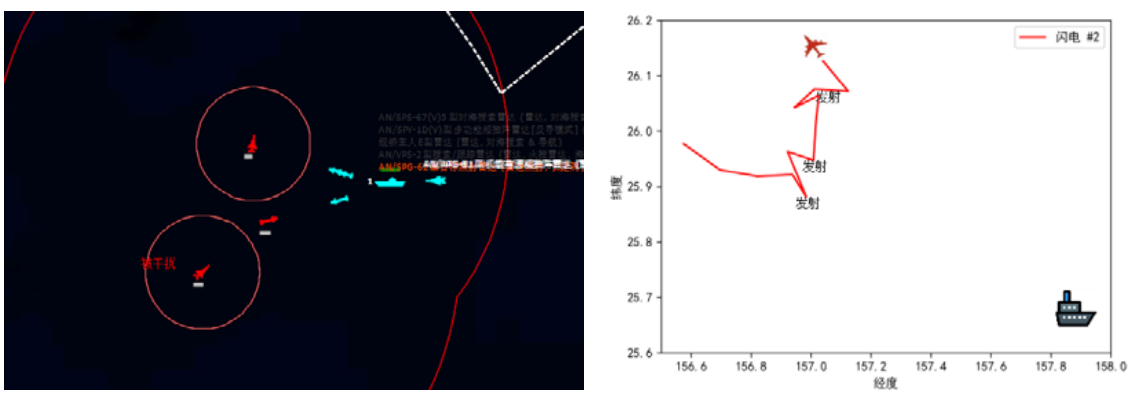
图 11 补充侦察策略效果图和轨迹图

在对补充侦察策略的产生原因进行分析时发现：在同基线 Agent_1 对抗时，蓝方飞机分成战斗机编队和轰炸机编队两拨，战斗机编队在其护卫驱逐舰的东侧，当红方的飞机飞向蓝方驱逐舰时，蓝方的战斗机编队向西机动进行防御；此外，蓝方的轰炸机编队则关闭自身雷达，从战场南侧和战场北侧绕向红方的驱逐舰进行打击，红方的驱逐舰不能有效发现这两架轰炸机，只能依靠飞机雷达对相关空域进行探测. 这一对抗策略使得学习智能体学会了补充侦察策略，依靠飞机侦察补足驱逐舰侦察的不足，发现探测不准的飞机，保护己方驱逐舰，获得胜局.

(3) “蛇形”打击策略

由于驱逐舰携带防空武器，在对驱逐舰进行打击时，靠近驱逐舰过近容易被防空弹打击. 图 12(a)展示了红方轰炸机飞向蓝方驱逐舰附近空域并向其发射空地导弹打击，蓝方驱逐舰发射舰载防空导弹对目标导弹进行拦截，同时对来袭红方飞机进行打击.

由于舰船打击过程较短，且点位轨迹表示难以有效展示飞机轨迹点的时间先后次序，因而使用两个决策步长为间隔的点位轨迹，并使用连接线进行连接. 如图 12(b)展示了红方轰炸机“闪电 #2”对蓝方驱逐舰进行打击的线路，红方飞机呈现“蛇形”打击路线，首先靠近蓝方驱逐舰发射空地弹，发射成功后向斜向机动，尽可能远离蓝方的防空区域；满足新的打击状态后，再重新抵近发射导弹，然后斜向远离. 重复这个动作多次,直到完成对所携带导弹的发射.



(a) (b)

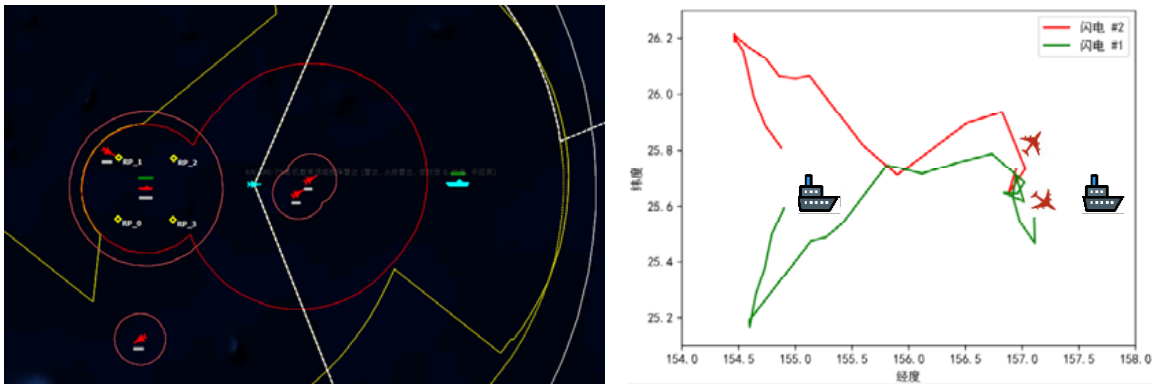
图 12 “蛇形”打击舰船效果图和轨迹图

结合平台特性，对轰炸机“蛇形”打击舰船的策略形成原因进行分析. 在平台内置想定中，轰炸机所携带的空地弹射程为 80 多公里，而驱逐舰的防空弹的拦截距离为 60 多公里. 然而在武器有效射程内的 75%距离下，所携带武器才具有较高命中率，而驱逐舰所发射的拦截弹拦截空地弹多发生在驱逐舰周边 40 km 的位置中. 这意味着轰炸机需要抵近驱逐舰周边 60 km 才具有较高的打击成功率，然而这一范围使得轰炸机面临较高的被打击概率. 如果轰炸机在空地弹的最大射程发射武器，则有较低概率命中驱逐舰，迫使其更靠近驱逐舰后再开火；如果轰炸机在近距离发射空地弹后不能及时后撤躲避，则面临被打掉的风险. 综上原因，使得学习智能体在学习中发现了“蛇形”打击舰船的策略，有效减少了轰炸机的损失.

(4) 轰炸机靠后突袭策略

想定中包含了轰炸机和战斗机两类飞机，其中仅有轰炸机，即“闪电”，可打击驱逐舰，而摧毁驱逐舰可以获得较高的分数，决定对抗的胜败. 图 13(a)展示了轰炸机靠后突袭策略，此时空中对抗已经结束，红方的两架战斗机在驱逐舰的西侧和南侧进行侦察，两架轰炸机在中心区域向蓝方驱逐舰飞行，准备进行驱逐舰打击.

为了更好地展示轰炸机的行动策略，使用以 5 个决策步长为间隔的点位轨迹连线图进行表示，如图 13(b)所示. 轰炸机“闪电 #1”和“闪电 #2”在对抗初期没有进入主要战场，而是避开空中对抗进行空域侦察，在红方空战不占优后，迅速向主要战场机动，补充空战实力，帮助红方获得制空权，在此之后，结合已有状态展开对蓝方驱逐舰的打击.



(a) (b)

图 13 轰炸机靠后突袭效果图和轨迹图

在分析策略形成原因时发现，轰炸机较早展开空中对抗容易被击落，进而无法完成想定的最高任务目标——击沉蓝方驱逐舰；此外，由于轰炸机携带空地导弹，所带燃油小于战斗机燃油总量，较早地展开空中对抗，使得轰炸机油耗急剧增加，过早地返航. 因而，轰炸机应尽可能少参与空中对抗，在保证空中优势的情况下展开对蓝方驱逐舰的打击，是较优策略.

5 结 论

在诸如军事行动、反恐突击等强对抗场景中，如何根据碎片化、不确定的实时信息为类型多元、功能异构、约束复杂的不同系统规划具有博弈优势的弹性方案，是聚合单平台能力、涌现体系优势的重要手段. 本文依托全国兵棋推演大赛比赛平台，设计了基于大赛的机机对抗赛想定的典型空战，基于深度强化学习方法开展了典型空战想定下的行动策略规划研究. 首先，基于 MDN-RNN 学习了状态预测编码，采用 VAE 对原始任务的状态空间进行压缩，并在状态压缩空间内展开任务的动态模型学习，对任务的未来状态、奖赏、结束标记、可用动作空间进行了预测；基于预测编码展开行动策略规划研究，并使用样本自适应方法加快状态值函数的学习. 实验结果表明：基于 MDN 状态表示、有样本自适应策略模型——MDN-AF 以 71%获胜胜率排名第一，且学到 4 种长时行动策略；在 2020 年全国兵棋推演大赛中，基于此算法设计的智能体获得了全国一

等奖的成绩, 进一步证明了模型的有效性. 虽然本文所提方法在当前的想定中取得了较好的结果, 但在未来的研究中将更加关注: 在更广泛的算法(如 DQN, DDPG, SAC)中应用所提算法框架, 分析算法效率; 采用离线学习编码与在线学习策略的方式, 从时间和空间复杂度角度分析算法优势等.

References:

[1] Mcdermott DV, Hendler JA. Planning: What it is, what it could be, an introduction to the special issue on planning and scheduling. Artificial Intelligence, 1995, 76(1-2): 1−16.

[2] Fikes RE, Nilsson NJJAI. STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 1971, 2(3-4): 189−208.

[3] Fang C, Franck M, Min X, *et al*. An integrated framework for risk response planning under resource constraints in large engineering projects. IEEE Trans. on Engineering Management, 2013, 60(3): 627−639.

[4] Feng Y, Cai ZY, Wang XH, *et al*. A plan recognizing algorithm based on fuzzy cognitive plan map. Int'l Journal of Performability Engineering, 2017, 13(7): 1094−1100.

[5] Wilkins DE. Planning and reacting in uncertain and dynamic environments. Journal of Experimental & Theoretical Artificial Intelligence, 1995, 7(1): 121−152.

[6] Currie K, Tate AJAI. O-plan: The open planning architecture. Artificial Intelligence, 1991, 52(1): 49−86.

[7] Myers K. PASSAT: A user-centric planning framework. In: Proc. of the 3rd Int'l NASA Workshop on Planning and Scheduling. Washington: NASA, 2003.

[8] Erol K, Hendler J, Nau DS. Semantics for hierarchical task-network planning. Maryland: University of Maryland at College Park, 1994.

[9] Shao TH, Zhang HJ, Cheng K, *et al*. Review of replanning in hierarchical task network. Systems Engineering and Electronics, 2020, 12(42): 2833−2846 (in Chinese with English abstract).

[10] Yz HMN, David WAZ, Len BZ, *et al*. HICAP: An interactive case-based planning architecture and its application to noncombatant evacuation operations. In: Proc. of the 6th National Conf. on Artificial Intelligence & the 11th Innovative Applications of Artificial Intelligence Conf. on Innovative Applications of Artificial Intelligence. Orlando: ACM, 2008.

[11] Mulvehill A, Caroli J. JADE: A tool for rapid crisis action planning. In: Proc. of the 5th Int'l Command and Control Research and Technology Symp. Stockholm: MIT, 2000.

[12] Yu WH, Han JS. Research on maritime search and rescue case-based system. Microcomputer Applications, 2011, 27(4): 13−15+4 (in Chinese with English abstract).

[13] Zhang XD, Wang T, Zhang L. Demand analysis of oil support in military operations with case-based reasoning method. Journal of Military Transportation University, 2018, 20(6): 14−17 (in Chinese with English abstract).

[14] Rao AS, Georgeff MP. BDI agents: From theory to practice. In: Proc. of the 1st Int'l Conf. on Multiagent Systems. California: AAAI, 1995.

[15] Holliday P. SWARMM—A mobility modelling tool for tactical military networks. In: Proc. of the 2008 IEEE Military Communications Conf. California: IEEE, 2008.

[16] Fugere J, LaBoissonniere F, Liang Y. An approach to design autonomous agents within ModSAF. In: Proc. of the 1999 IEEE Int'l Conf. on Systems, Man, and Cybernetics. Tokyo: IEEE, 1999.

[17] Wooldridge M. An Introduction to MultiAgent Systems. New Jersey: Wiley Publishing, 2009.

[18] Li H, Chang GC, Sun P. Operational plan making based on procedure reasoning system. Electronics Optics & Control, 2008(10): 51−54 (in Chinese with English abstract).

[19] Liang XX, Feng YH, Ma Y, *et al*. Deep multi-agent reinforcement learning: A survey. Acta Automatica Sinica, 2020, 46(12): 2537−2557 (in Chinese with English abstract).

[20] Liang XX, Feng YH, Huang JC, *et al*. Novel deep reinforcement learning algorithm based on attention-based value function and autoregressive environment model. Ruan Jian Xue Bao/Journal of Software, 2020, 31(4): 948−966 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5930.htm [doi: 10.13328/j.cnki.jos.005930]

[21] Chinese Institute of Command and Control. In: Proc. of the 4th National Wargaming Competition in 2020. 2020 (in Chinese with English abstract). http://www.ciccwargame.com/

[22] Hand DJ. Who's #1? The science of rating and ranking. Journal of Applied Statistics, 2012, 39(10): 81–83.

[23] Richard SS, Andrew GB. Reinforcement Learning: An Introduction. 2nd ed., Massachusetts: MIT, 2017.

[24] Schulman J, Levine S, Abbeel P, *et al*. Trust region policy optimization. In: Proc. of the 32nd Int'l Conf. on Machine Learning. Lile: JMLR.org, 2015. 1889–1897.

[25] Schulman J, Wolski F, Dhariwal P, *et al*. Proximal policy optimization algorithms. arXiv: 1707.06347, 2017.

[26] Mnih V, Kavukcuoglu K, Silver D, *et al*. Human-level control through deep reinforcement learning. Nature, 2015, 518(7540): 529–533.

[27] Haarnoja T, Zhou A, Abbeel P, *et al*. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proc. of the 35th Int'l Conf. on Machine Learning. Stockholm: PMLR 80, 2018.

[28] Kingma DP, Welling M. Auto-encoding variational Bayes. arXiv: 1312.6114v10, 2013.

[29] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans.on Neural Networks, 1994, 5(2): 157–166.

[30] Ha D, Eck D. A neural representation of sketch drawings. arXiv: 1704.03477, 2017.

[31] Wang ZY, Bapst V, Heess N, *et al*. Sample efficient actor-critic with experience replay. arXiv: 1611.01224, 2016.

[32] Schaul T, Quan J, Antonoglou I, *et al*. Prioritized experience replay. arXiv: 1511.05952, 2015.

附中文参考文献:

[9] 邵天浩, 张宏军, 程恺, 戴成友, 余晓晗, 张可. 层次任务网络中的重新规划研究综述. 系统工程与电子技术, 2020, 42(12): 2833−2846.

[12] 于卫红, 韩俊松. 海上搜救案例库系统的研究. 微型电脑应用, 2011, 27(4): 13−15, 4.

[13] 张晓东, 汪涛, 张磊. 基于案例推理的军事行动油料保障需求分析. 军事交通学院学报, 2018, 20(6): 14−17.

[18] 李皓, 常国岑, 孙鹏. 采用过程推理系统的作战方案生成研究. 电光与控制, 2008(10): 51−54.

[19] 梁星星, 冯旸赫, 马扬, 程光权, 黄金才, 王琦, 周玉珍, 刘忠. 多 Agent 深度强化学习综述. 自动化学报, 2020, 46(12): 2537−2557.

[20] 梁星星, 冯旸赫, 黄金才, 王琦, 马扬, 刘忠. 基于自回归预测模型的深度注意力强化学习方法. 软件学报, 2020, 31(4): 948−966. http://www.jos.org.cn/1000-9825/5930.htm [doi: 10.13328/j.cnki.jos.005930]

[21] 中国指挥与控制协会. 2020 第四届全国兵棋推演大赛. 2020. http://www.ciccwargame.com/

附录 A. 预测编码模型损失项

- 状态值函数的损失 $loss_v$

使用均方误差(mean-square error)进行度量, 反映估计量与被估计量之间差异程度, 估计的均方误差越小越好:

$$loss_v = mse\_loss(v(z_t), \tilde{v}(z_t)) = \frac{1}{2}\mathbb{E}(v(z_t) - \tilde{v}(z_t))^2.$$

- 奖赏值函数 r_{t+1} 损失 $loss_r$

根据隐层状态 h_t 预测一个标量值, 因而使用均方误差损失函数进行度量.

- 结束标志 d_{t+1} 损失 $loss_{done}$

结束标志 d_{t+1} 为 onehot 编码的向量, 形如[1,0], 结束与未结束的标志是互斥的, 是典型的多分类问题, 因而使用交叉熵(cross entropy loss)损失函数, 目标分类的概率值 $P(y|x)$越大越好, 反之越小越好, 利用极大似然估计的原理对预测类别的概率进行表示 $\log P(y|x)$. 在神经网络中, 优化器向最小化方向优化, 因而使用 $-\log P(y|x)$作为损失函数, 多分类交叉熵损失为

$$\log P(y\mid x)=\log(\hat{y}^{y}\cdot(1-\hat{y})^{(1-y)})=y\log\hat{y}+(1-y)\log(1-\hat{y}),$$
$$loss_{done}=-[y\log\hat{y}+(1-y)\log(1-\hat{y})].$$

- 可用动作 $mask_{t+1}$ 损失 $loss_{mask}$

可用动作 $mask_{t+1}$ 表示当前步中的可选动作, 兵棋推演想定中的智能体动作在不同的状态下是可变的, 因而需要使用一组标志位用于表示当前步的可选动作, 表示为

$$\begin{bmatrix}[0,1,0,...,1,0]\\ [0,0,1,...,1,1]\\ [0,1,1,...,1,1]\end{bmatrix}.$$

可选动作的表示是 0,1 编码的一组向量, 除去第一个动作, 剩余各个动作之间是相容的, 不能使用交叉熵损失函数进行学习. 根据数据特征, 使用深度学习图像识别任务中的多标签图像分类任务进行优化. 在该任务下, 图像的分类含有多个标签值, 在设计神经网络时, 输出端的神经元的概率和不为 1, 而是每个神经元输出表示其所代表标签的概率值, 各个输出神经元间相互独立. 在多分类任务中, 神经元的最后一层是使用 *softmax* 函数进行归一化; 而在多标签分类任务中, 神经元的最后一层使用 *Sigmoid* 函数将单个神经元输出到[0,1]之间, 利用二元交叉熵(binary cross entropy)对单个标签的损失进行衡量:

$$-[y\log\hat{y}+(1-y)\log(1-\hat{y})],$$

其中, y 为标签, $\hat{y}$ 为预测值. 当标签为 0 时, 上式前半部分为 0, $\hat{y}$ 需要尽可能为 0 才能使后半部分数值更小; 当标签为 1 时, 后半部分为 0, $\hat{y}$ 需要尽可能为 1 才能使前半部分的值更小, 这样就达到了让 $\hat{y}$ 尽量靠近标签的预期效果. 在多标签分类任务中的损失函数为多个标签的 BCE 的均值误差, 即:

$$loss_{mask}=\frac{1}{n}\sum_{i=1}^{n}-[y_i\log\hat{y}_i+(1-y_i)\log(1-\hat{y}_i)].$$

可用动作的预测误差同多标签分类问题一致, 使用 BCE 损失能够度量预测可用动作 $mask_{t+1}$ 的损失.

在 PyTorch 深度学习框架中, 内嵌了衡量多标签损失的 *BCEWithLogitsLoss* 函数, 该函数在执行中直接将 *Sigmoid* 函数和 *BCELoss* 进行合并, 省去了最后一步的 *Sigmoid* 函数操作, 提高了算法的可读性.

- 编码 $\tilde{z}_{t+1}$ 预测损失 $loss_z$

兵棋推演中的环境非平稳性受环境自身转移特性和对抗智能体的容量影响, 不确定性相较于一般的强化学习环境显著. 为了更加准确地估计环境中转移的不确定性, 使用基于多元高斯分布的混合密度函数对下一时刻的状态压缩编码 $\tilde{z}_{t+1}$ 进行表征. 在概率与统计中, 利用包含多个随机变量的随机变量集合生成一个新的随机变量, 则该随机变量的分布称为混合分布: 根据给定概率从集合中随机选取一个随机变量, 然后再实现该随机变量的值. 如在单变量混合分布中, 给定概率密度函数集合 $p_1(x),\ldots,p_n(x)$以及权重 $w_1,\ldots,w_n$, 则密度函数为 $p(x)=\sum_{i=1}^{n}w_ip_i(x)$, 其中, $w_i\geqslant 0$, $\sum_{i=1}^{n}w_i=1$. 如果集合中的随机变量是连续的, 则生成的随机变量也将是连续的, 其概率密度函数被称为混合密度; 如果集合中的随机变量是随机向量(每个向量的维数相同), 则混合分布又被称作多变量分布. 因而, 混合密度分布下的编码预测损失表示如下式所示:

$$loss_z=-\frac{1}{N}\sum_{i=1}^{N}\log\left(\sum_{j=1}^{M}w_j\mathcal{N}(\tilde{\boldsymbol{z}}_{i,j}\mid\boldsymbol{\mu}_{i,j},\boldsymbol{\sigma}_{i,j},\boldsymbol{\rho})\right),$$

其中, N 表示预测变量的维度, M 表示单个变量下的概率密度函数数量, $\boldsymbol{\rho}$表示概率密度函数见的协方差(在训练中假定概率密度函数之间相互独立, 因而忽略该项).

附录 B. 想定设计与设置

为了方便开展研究, 凸显算法性能, 本文依托全国兵棋推演大赛的机机对抗赛的想定——“海峡风暴”精简版想定开展研究.

想定持续时间为 100 min, 选择在长度约 1 000 km、宽度约 600 km 的开放海域展开海上攻防战. 推演双方为红方和蓝方, 为了保证对抗的公平性, 双方的武器装备布置对称, 如图 14 所示.

以红方推演方为例, 对想定中的单方要素进行描述. 想定中的要素包含海上、空中单元两类.

- 海上单元: 一艘航母和一艘驱逐舰, 具体配置见表 4. 由于舰船上的武器配置是按照相关的条令配置, 同想定实际对抗无关的武器配置不列出(如鱼雷等). 其中, 航母位于驱逐舰后侧约 200 公里处,舰船均可移动. 在想定中, 航母不可被击毁得分, 驱逐舰可以被击毁得分;
- 空中单元: 空中作战飞机隶属于航母, 自航母上起飞后展开军事行动. 在表 3 中给出了各型飞机的数量. 各型飞机的具体配置见表 5.

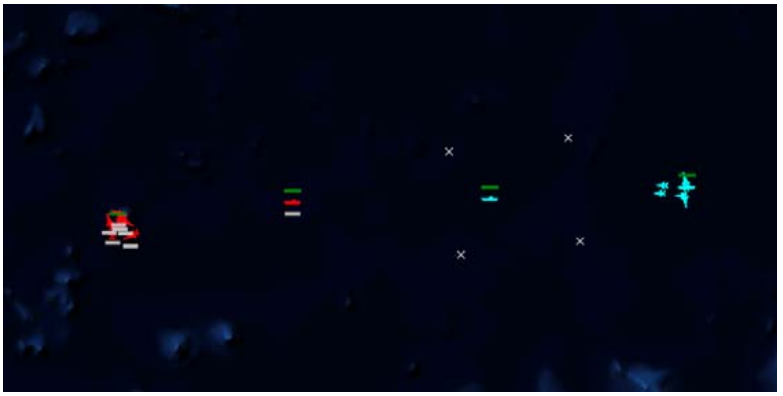


图 14 想定示意图

表 4 海上单元配置信息

类型	作战武器	载机
CVN-78“杰拉德.R.福特”福特级号航空母舰	880x 12.7 毫米/50 高射机炮[10 发备弹] 42x RIM-116C 滚体导弹(拉姆导弹) 16x RIM-162D 舰对空导弹(海麻雀)	4x F/A-18 型战斗机 2x F-35C 型战斗机
DDG 113“约翰.芬”阿里伯克级 Flight IIA 导弹驱逐舰	440x 12.7 毫米/50 高射机炮[10 发备弹] 15x Mk15 型 20 毫米“火神”密集阵近程防御武器系统[300 发备弹] 228x 25 毫米/75Mod2 型大毒蛇舰炮炮弹[12 发备弹] 56x RGM-109E 战术战斧 Blk IV 巡航导弹 8x RIM-162A 型“海麻雀”舰对空导弹	无

表 5 空中单元配置信息

类型	作战武器
F/A-18E 型战斗机	4x 20 毫米/85 M61A2 型火神式航空机炮[100 发备弹] 12x 通用箔条(齐射)[5x 弹药桶] 20x 通用红外干扰弹(齐射)[3x 弹药桶,双光谱] 3x AN/ALE-55 飞机光纤拖曳式诱饵 2x AIM-120D 型先进中程空空导弹 P3I.4 6x AIM-9X 型“响尾蛇”空空导弹
F-35C 型战斗机	12x 通用箔条(齐射)[5x 弹药桶] 8x 通用红外干扰弹(齐射)[3x 弹药桶, 双光谱] 4x AN/ALE-70 型光纤拖曳式诱饵装置 6x AGM-154C 联合防区外武器[内装多级侵彻战斗部] 2x AIM-120D 型先进中程空空导弹 P3I.4 2x AIM-9X 型“响尾蛇”空空导弹

想定的作战目标为有效打击敌对方空中作战力量, 寻机击毁敌对方驱逐舰. 为了达成这一目标, 对不同的单元目标赋予不同的奖惩得分, 具体得分见表 6, 想定的胜负判定为得分胜负.

表 6 单元得分标准

推演事件	得分	备注
击毁一架飞机	139	–
损失一架飞机	–139	–
击毁驱逐舰	–1 843	–
击毁航母	0	无法击毁

附录 C. 状态与动作定义

从上节的全国兵棋推演大赛平台特征介绍可知, 平台推送的敌我双方信息复杂多样且行动空间庞大. 为了方便开展智能体的开发, 需要对想定中的单元信息进行结构化处理, 并对可执行动作进行限定与表示.

在本文的研究中, 从 5 类数据对己方所感知的全局状态进行描述: 己方飞机、己方舰船(仅描述驱逐舰)、敌方飞机、敌方舰船(仅描述驱逐舰)、敌方导弹, 每类结构化信息如表 7 所示. 想定中使用经纬度表示实体的位置信息, 为了降低区域位置对模型的影响, 通过相对位置, 以手动放缩的形式将单元位置信息约束至[–1,1], 相对经度和相对纬度为实体位置距离想定区域中心的相对位置; 燃油为当前燃油同最大燃油的比值; 己方飞机中的武器列表包含了 3 种武器的剩余比例——先进中程空空导弹、空空导弹、联合防区外武器, 己方舰船的武器剩余包含了 RIM-162A 型“海麻雀”舰对空导弹等剩余比例; 飞机的状态包含起飞、在空、返航, 其中, 在空状态下飞机具备机动等能力, 为可操作实体; 飞机类型包含 F/A-18E 型战斗机和 F-35C 型战斗机, 前者仅支持对空打击, 后者既可对空打击也可对海攻击; 上步动作记录上一步采取的动作编码; 当前步可用动作是当前智能体的可用动作列表; 己方舰船的损管是当前舰船的损坏比, 舰船具有外层防护, 毁伤受平台损管模型控制, 命中一发导弹不足以将其击沉; 敌方飞机中的被打击, 用来记录当前步有无己方导弹对其进行打击; 敌方导弹武器类型根据平台发回的探测信息, 分为地对空的防空导弹和空对空的空空导弹.

表 7 状态数据表示

类型	信息种类与类型
己方飞机	相对经度(float), 相对纬度(float), 燃油(float), 武器列表(list[float]), 飞机状态(one-hot), 飞机类型(one-hot), 上步动作(one-hot), 当前步可用动作(list[bool])
己方舰船	相对经度(float), 相对纬度(float), 武器剩余(float), 损管(float)
敌方飞机	相对经度(float), 相对纬度(float), 被打击(bool)
敌方舰船	相对经度(float), 相对纬度(float)
敌方导弹	相对经度(float), 相对纬度(float), 武器类型(one-hot)

全国兵棋推演大赛平台可操作的动作包含推演方、任务与实体等多层级, 舰船、飞机等多类型, 平台、条令、实体等多种类控制动作. 在对“海峡风暴”想定进行分析后, 我们发现该想定中的制胜关键在于获得战场制空权, 因而对可灵活机动的飞机控制尤为重要, 而驱逐舰的主动防空和被动防御控制相对较少, 不是决定胜负的关键. 因而本文对飞机控制动作进行梳理, 整理出该想定下飞机的动作集合.

飞机的速度与飞行朝向是连续的控制量, 为了降低模型的学习难度, 通过离散化连续变量, 简化飞机的控制动作, 对飞机的速度控制按照平台的“油门”设定划分为 4 个值: 低速、巡航、军用、加力; 将飞行朝向设定为离散化的 6 个方向值; 将打击动作分为 6+1, 即 6 个可打击敌方飞机与 1 个驱逐舰. 具体的动作设计如表 8 所示, 共计 33 个动作.

- 0 位置的动作仅不可决策实体可以标志(为训练必需). 当飞机不是可决策实体时为 1, 如飞机燃油耗尽进入返航, 或者飞机被击毁等, 该位置同之后的 32 位动作为互斥关系;
- 1–24 位置的动作为机动控制(速度与朝向). 为了约束飞机的作战区域, 防止无意义的探索, 根据作战飞机的当前位置, 判断飞机采取此位动作后会不会飞出作战区域, 飞出区域的动作不可用, 想定中的作战飞机包含了两类飞机, 其中, F-35C 型战斗机——“闪电”的油门控制仅有 3 种;
- 25–30 为打击的敌方单元. 不同武器在不同距离下的命中概率不同, 且远距离飞行打击目标时, 同机动控制动作重叠, 为了解耦打击和机动动作, 在当前飞机可打击范围内判断是否可打击此位代表的

目标, 选中打击目标后, 作战武器会在一个决策步内成功发射武器;

- 31 为打击舰船. 舰船在武器打击范围内则可打击, 此外, 仅有“闪电”类的 F-35C 型战斗机携带空地作战武器;
- 32 为就地空中盘旋, 可以防止飞机在空间中的无序飞行.

表 8 飞机控制动作表示

0	1–24	25–30	31	32
可选	机动控制(速度与朝向)	可打击的空中单元	舰船打击	不做动作

附录 D. 实验设置

(1) 想定环境设置

空战想定的推演步长为 100 分钟, 想定的初始状态中, 飞机停留在航母中, 想定开始后, 智能体可以单机出动的形式自行选择飞机起飞时刻. 在本文的设定中, 设置为在起始时刻, 一次性起飞所有飞机, 飞向己方驱逐舰上空, 距离驱逐舰约 50km 左右时, 智能体进入强化学习自主规划阶段.

此外, 为了限制飞机在非作战区域的探索, 根据战场重心, 对智能体决策的空间区域进行限制, 以敌我双方驱逐舰的位置作为参考点, 在经度约束中, 左侧经度限定在己方驱逐舰的右侧 50km 处, 约 0.5 经度差, 右侧经度限定在敌方驱逐舰的右侧 100km 处, 约 1 经度差; 在纬度约束中, 约束至敌我双方驱逐舰纬度中心的上下约 200km, 纬度跨度约 4 纬度, 如图 15 所示.

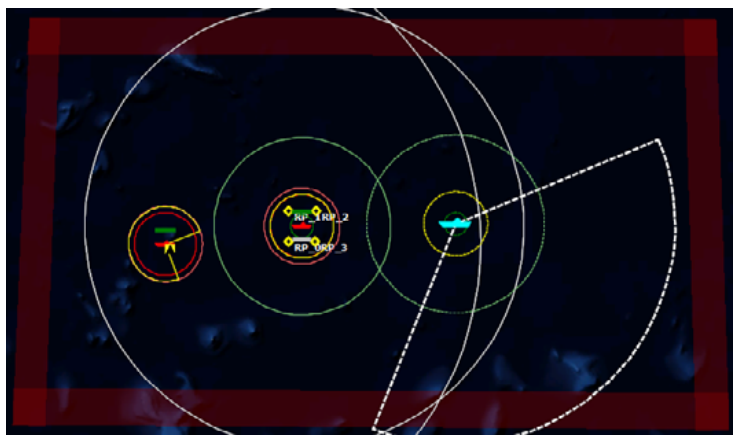


图 15 行动空域限定

兵棋推演平台内置的模型会根据作战实体的燃油消耗判定飞机是否坠毁, 因而在规则设置中, 当飞机到达 bingo 油量(剩余燃油达到返航的最低要求)后自行返航.

想定中的飞机包含空空(F/A-18E 型战斗机)和空地(F-35C 型战斗机)等两型飞机. 空空作战飞机包含 AIM-120D 型先进中程空空和 AIM-9X 型“响尾蛇”空空等两型导弹, 前者射程为 138 公里, 后者射程为 19 公里, 因而对目标分配打击规划采用的为 AIM-120D 型先进中程空空导弹, 将其设置为“从不自动攻击目标”, 由智能体决定何时发射; AIM-9X 型“响尾蛇”空空则为“最大射程自动攻击目标”, 由推演平台自行决定是否发射. 空地飞机相比空空作战飞机多了一型打击驱逐舰的武器——AGM-154C 联合防区外武器, 射程为 80 公里. 该武器数量较多, 因而在打击过程中, 每次打击使用两发齐射且为“从不自动攻击目标”, 由智能体决定何时发射.

驱逐舰在想定中具有较高的分值, 决定着对抗的胜负. 而驱逐舰对空防御能力一般, 仅有 8 枚可控的武器——RIM-162A 型“海麻雀”舰对空导弹对来袭的 AGM-154C 联合防区外武器(最大 12 枚)进行防御, 且 RIM-162A 型“海麻雀”舰对空导弹对第四代飞机的打击成功率一般, 因而在规则设置中, 仅允许该型武器对射程范围内的制导武器进行拦截.

其他项的设置遵从想定默认, 见附录 E.

(2) 学习设置

全国兵棋推演平台以实时对抗的形式连续对抗 100 min, 为了更加符合兵棋推演的目的, 本文采用 15 s 的时间间隔生成行动策略. 在训练中, 每完成一局进行一次样本存储, 最后一个转移的奖赏由最后得分同规划结束时得分的差值进行计算. 考虑到每局的决策长度不一致, 根据时间计算得到最大决策步为 600 步, 对未决策步在推演结束时统一进行补足, 并标记样本的来源.

利用优先经历回放机制对生成的样本进行存储, 存储空间大小为 2^{10} 局, 利用模型求解当前局的 TD error δ进行存储. 在 PPO 算法训练中, 每次随机采样 4 局, 每一次训练重复 8 次.

在训练过程中, 每采样 2^{10} 局后, 对 VAE 模型进行 50 次迭代, 并对训练的模型展开测试, 同各个基线模型对抗 10 次, 记录数据. 训练模型中的超参数设置见表 9.

表 9 网络超参数

参数项	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β	τ	R_{length}
值	1	1	1	1	0.5	1	0.01	0.01	0.999	0.5	32

在表 9 中, β_1 表示网络损失函数中的超参数, β, τ表示样本自适应采样中的退火因子和初始温度, R_{length} 表示 MDN-RNN 模型中的 RNN 训练长度. VAE 模型训练中, 对重构项的损失权重设置为 1 000, 正则化项损失设置为 1; 在起始训练中, VAE 和 MDN-RNN 模型采用硬更新的方式对参数进行迭代, 之后每次采用软更新的方式更新迭代参数, 软更新的参数为 0.01.

附录 E. 智能体方其他规则设计

红方推演方条令与交战规则设置如图 16 所示.

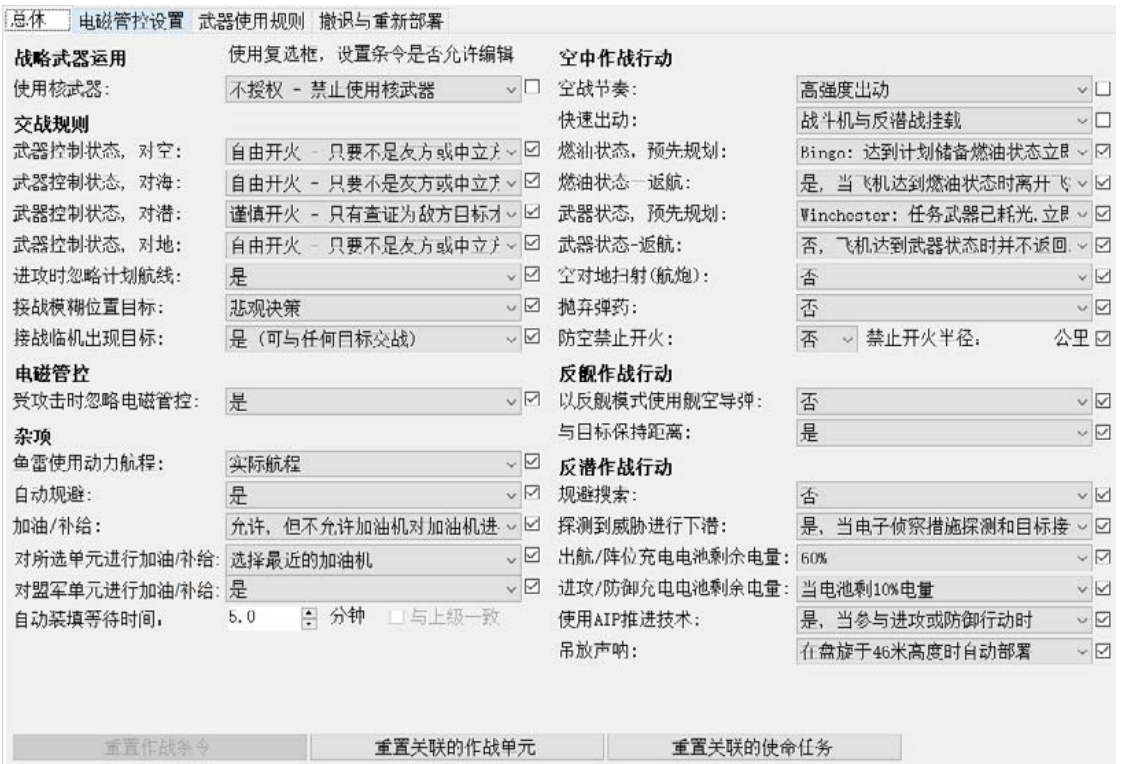


图 16 红方作战条令与交战规则

红方推演方电磁管控设置如图 17 所示.

图 17 红方电磁管控设置

红方推演方“海麻雀”舰对空导弹设置如图 18 所示.

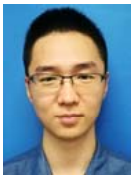
图 18 红方“海麻雀”舰对空导弹设置

红方推演方联合防区外武器设置如图 19 所示.

图 19 红方联合防区外武器设置

红方推演方先进中程空空导弹设置如图 20 所示.

图 20 红方先进中程空空导弹设置



梁星星(1992－)，男，博士生，主要研究领域为智能规划，军事智能博弈对抗.

马扬(1993－)，男，博士生，主要研究领域为网络嵌入，链路预测，图神经网络.

冯旸赫(1985－)，男，博士，副教授，主要研究领域为因果发现与推理，主动学习，强化学习.

张驭龙(1988－)，男，博士生，主要研究领域为信息系统，强化学习，智能博弈.

张龙飞(1988－)，男，博士生，主要研究领域为机器学习，深度强化学习.

廖世江(1989－)，男，主要研究领域为军事智能博弈对抗.

刘忠(1968－)，男，博士，教授，博士生导师，主要研究领域为多智能体系统.