计算机科学
COMPUTER SCIENCE

基于多任务强化学习的优先级加权软模块化方法:SM-PHT

潘嘉豪, 冯翔, 虞慧群

引用本文

潘嘉豪, 冯翔, 虞慧群. 基于多任务强化学习的优先级加权软模块化方法:SM-PHT[J]. 计算机科学, 2026, 53(4): 366-376.

PAN Jiahao, FENG Xiang, YU Huiqun. SM-PHT:Robust,Scalable,and Efficient Method for Multi-task Reinforcement Learning [J]. Computer Science, 2026, 53(4): 366-376.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.250700198

基于多任务强化学习的优先级加权软模块化方法:SM-PHT

潘嘉豪　冯　翔　虞慧群
华东理工大学信息科学与工程学院　上海 200237
(y30231040@mail.ecust.edu.cn)

摘　要　近年来,强化学习在众多领域中取得了显著的成功。然而,在动态环境或多任务场景中,传统方法往往难以有效适应复杂变化,表现出一定的局限性。为解决这一问题,提出了一种名为"优先级加权软模块化"的多任务强化学习方法(SM-PHT),旨在提升智能体在多任务环境下的适应性与泛化能力。SM-PHT 融合了 3 项关键技术:优先级加权知识蒸馏、分层缓存机制和任务嵌入策略。优先级加权知识蒸馏通过加权方法整合多个高性能模型的知识,提高了学生网络的鲁棒性与稳定性。分层缓存机制分别管理低层次经验数据与高层次模型参数,提升了学习效率。任务嵌入策略则通过捕捉环境特征,增强任务表示,来促进跨任务知识迁移。实验结果表明,在 Meta-World MT10 基准测试中,SM-PHT 的成功率达到当前最优方法的两倍,平均奖励提高 30%;在更具挑战性的 MT50 任务中,成功率与平均奖励均提升约 10%。上述指标显示该方法在复杂多任务场景中具有良好的稳定性与泛化能力,展示了其在实际多任务强化学习应用中的潜力。

关键词:软模块化方法;多任务强化学习;优先级加权知识蒸馏;分层缓存;任务嵌入
中图分类号　TP181

SM-PHT:Robust,Scalable,and Efficient Method for Multi-task Reinforcement Learning

PAN Jiahao,FENG Xiang and YU Huiqun
School of Information Science and Engineering,East China University of Science and Technology,Shanghai 200237,China

Abstract　In recent years,reinforcement learning has achieved remarkable success in various domains. However,traditional RL methods often struggle with adaptability when facing dynamic environments or multiple tasks. To address this challenge,this thesis introduces SM-PHT,a robust,scalable,and efficient method for multi-task reinforcement learning. The primary objective of this research is to enhance the adaptability and generalization capabilities of reinforcement learning agents in multi-task environments by enabling them to learn and transfer knowledge across multiple tasks. SM-PHT integrates three key mechanisms:priority-weighted knowledge distillation(PWKD),hierarchical buffer,and task embedding. PWKD leverages a weighted distillation process to assimilate knowledge from multiple high-performing models,improving the robustness and stability of the student network. Moreover,the hierarchical buffer employs dual buffers to store low-level experiential data and high-level model parameters,optimizing offline learning efficiency. Finally,task embedding enrich task representations by capturing detailed environmental characteristics,facilitating effective knowledge transfer. Experiments conducted in the Meta-World environment demonstrate SM-PHT's superior performance compared to state-of-the-art methods. In the MT10 challenge,SM-PHT achieves double the success rate and a 30% increase in average rewards. In the more complex MT50 challenge,it improves the success rate by approximately 10% and increases average rewards by around 10%. These results highlight SM-PHT's ability to handle complex tasks with remarkable stability and minimal fluctuation,making it a promising approach for real-world MTRL applications.

Keywords　Soft modularization network,Multi-task reinforcement learning,Priority-weighted knowledge distillation,Hierarchical buffer,Task embedding

到稿日期:2025-07-31　返修日期:2025-10-23
基金项目:国家自然科学基金重点项目(62136003);国家自然科学基金(62276097,62372174)
This work was supported by the Key Program of National Natural Science Foundation of China(62136003) and National Natural Science Foundation of China(62276097,62372174).
通信作者:冯翔(xfeng@ecust.edu.cn)

1 引言

1.1 研究背景

深度强化学习（Deep Reinforcement Learning，DRL）作为前沿技术，已在游戏[1-2]、自动驾驶[3-4]和工业机器人控制[5-6]领域取得显著成就。它通过试错机制让智能体探索并优化行为，以获取最大累积奖励。与监督学习或无监督学习不同，强化学习不依赖预标记的数据集，而是根据环境反馈调整模型参数。结合深度学习技术，它能有效处理高维度状态空间和大规模动作空间的问题。随着技术的发展，单任务强化学习在适应多变环境方面的局限性逐渐显现，多任务强化学习（Mutli-Task Reinforcement Learning，MTRL）因此受到更多关注。在机器人操作[7-8]、自动驾驶[9-10]和游戏 AI[11-12]等领域，多任务强化学习使智能体能够更快适应新任务和环境变化，并同时处理多个任务，提升泛化能力，满足实际应用的多样化和动态变化的需求。

1.2 贡献

本文提出了优先级加权软模块化方法（Soft Modularization with Priority-weighted Knowledge Distillation，Hierarchical Buffer and Task Embedding，SM-PHT）。该方法通过引入优先级加权知识蒸馏、分层缓存和任务嵌入 3 项创新机制，不仅允许智能体在学习新任务时保留已有知识，还增强了模型对历史数据的利用能力和跨任务的知识迁移能力，显著提高了智能体在复杂多任务环境中的适应性和鲁棒性。

具体来说，本文的主要贡献集中在 4 个方面。

1）提出了创新的优先级加权知识蒸馏机制，引入加权蒸馏过程整合多个高性能模型知识，捕捉更广泛的任务特征。

2）设计了包含“经验缓存”和“参数缓存”的分层缓存机制，帮助智能体从多种策略中提取关键特征，促进知识转移并增强模型泛化能力。

3）引入任务嵌入机制，捕捉详细的环境特征来丰富任务表示，揭示任务间的内在关联，促进高效的知识迁移。

4）在 Meta-World 环境中进行实验，实验结果验证了 SM-PHT 方法在适应性、稳定性和效率方面的优越性。在 MT10 挑战中，SM-PHT 的平均成功率翻倍，平均奖励提高了 30%；在更为复杂的 MT50 挑战中，平均成功率提升了约 10%，平均奖励增加了约 10%。这些结果表明，SM-PHT 在处理复杂任务时表现出色，具有高适应性和可靠性。

2 相关工作

MTRL 作为一种有潜力的方法，允许智能体在多个相关但独立的任务间进行知识学习与迁移。依据研究焦点和方法的不同，MTRL 可被广泛划分为三大类别。

1）基于迁移的 MTRL 利用已完成任务的知识加速新任务学习。例如，Liu 等通过提炼中心策略促进知识迁移[13]，但可能存在负迁移风险，即源任务中的知识会干扰新任务的学习；而 Rusu 等提出了渐进式神经网络[14]，通过引入新“列”处理新任务，并通过横向连接复用旧“列”的知识，但随着网络规模的扩大，存储开销显著增加。

2）专家混合（Mixture of Experts，MoE）算法为不同任务训练独立的专家模型，如 Expert Gate 算法利用门控机制选择适合特定任务的专家模块[15]。此外，Shen 等将 MoE 与指令微调技术相结合[16]，优化了大语言模型，但管理和调优门控机制较为复杂。

3）Rosenbaum 等提出的路由网络算法由一个路由网络和多个模块网络组成[17]，根据任务的输入选择合适的模块，避免负迁移并提升效率。而 Yang 等提出的软模块化机制（Soft Modularization，SM）通过动态调整模块权重分配[18]，而非依赖硬编码模块划分，显著提升了性能和效率。

鉴于软模块化在动态调整权重分配和抑制模块数量增长方面的优势，本研究建立在其框架基础上，力求进一步增强其在多任务强化学习中的性能和效率。

3 预备知识

3.1 马尔可夫决策过程

强化学习问题通常被形式化为马尔可夫决策过程（Markov Decision Process，MDP），用五元组 $M=(S,A,R,P,\gamma)$ 来描述。其中，S 表示智能体可能遇到的所有状态的集合；而 A 表示智能体在各个状态下可选择的动作集合；奖励函数 $R:S\times A\times S\to R$ 定义了从状态 s_t 执行动作 a_t 并转移到状态 s_{t+1} 的奖励；转移概率函数 $p(s_{t+1}\mid s_t,a_t)\in P$ 给出了在状态 s_t 下执行动作 a_t 后转移到状态 s_{t+1} 的概率；折扣因子 $\gamma\in(0,1)$ 用于折减未来奖励的价值。此外，策略 $\pi\in\Pi$ 是从状态到动作选择概率的映射，定义了在状态 s 下选择动作 a 的概率，记作 $\pi(a\mid s)$。

对于一个给定的可行 $\pi\in\Pi$，分别定义了状态值函数和状态-动作值函数，如式（1）和式（2）所示：

$$V_\pi^0(s)=E_\pi\left[\sum_{t=0}^{\infty}\gamma^t r_t\mid s_0=s\right] \tag{1}$$

$$Q_\pi^0(s,a)=E_\pi\left[\sum_{t=0}^{\infty}\gamma^t r_t\mid s_0=s,a_0=a\right] \tag{2}$$

基于上述定义，强化学习的核心目标是确定一个能够最大化累积折扣奖励的最优策略 π^*。从数学角度来看，这相当于在所有可能的状态 s 和动作 a 下，最大化状态值函数 $V_\pi^0(s)$ 或状态-动作值函数 $Q_\pi^0(s,a)$。

3.2 多任务马尔可夫决策过程

单任务强化学习聚焦于一个 MDP，而多任务强化学习则考虑一组任务 $\tau=\{T_1,T_2,\cdots,T_N\}$ 及对应的 MDP 集合 $M=\{m_1,m_2,\cdots,m_N\}$，其中 N 表示任务的总数。每个任务 T_i 由一个独立的 MDP m_i 建模，并具备一个嵌入向量 $\boldsymbol{z}_i$ 作为任务特性的潜在表示。因此，策略 $\pi\in\Pi$ 不仅依赖于状态 s 和动作 a，还依赖于任务嵌入 z。MTRL 的目标是找到一个共享策略，以最大化所有任务的累积折扣奖励之和。从数学上讲，这一目标可以形式化为：

$$\pi^*=\max_{\pi\sim\Pi}\left\{\frac{1}{N}\sum_{i=1}^{N}E_\pi\left[\sum_{t=0}^{\infty}\gamma_i^t r_i(s_t,a_t)\right]\right\} \tag{3}$$

3.3 软模块化网络

为解决传统 MTRL 算法的问题，Yang 等引入了软模块化技术[18]。其设计了一个路由网络，该网络基于当前状态和

任务标识估计不同模块之间的连接权重。这使得每个任务能够动态配置策略或价值网络中的模块，创建针对特定任务的独特架构。多头网络、路由网络和软模块化网络的比较如图 1 所示。

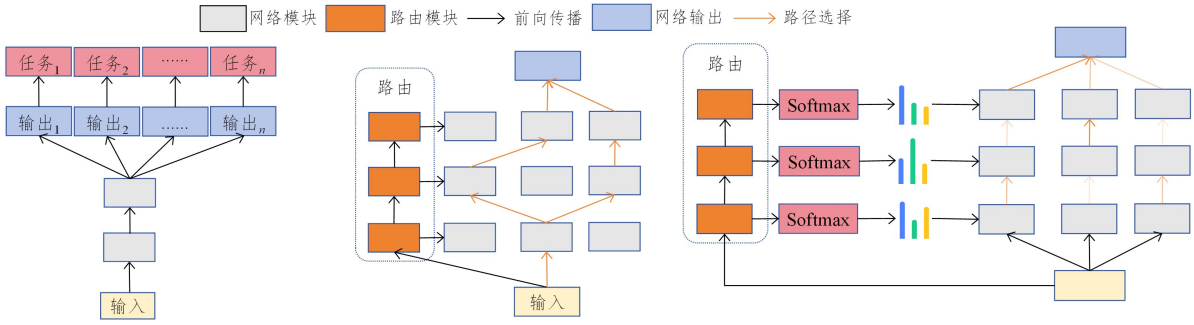


图 1 多头网络、路由网络和软模块化网络的比较

Fig. 1 Comparison of multi-head networks, routing networks, and soft modularization networks

4 优先级加权软模块化方法

本章重点介绍了 SM-PHT 的关键创新点，包括优先级加权知识蒸馏、分层缓存、任务嵌入，这些创新点显著提升了多任务强化学习系统的适应性。SM-PHT 系统的具体流程如图 2 所示。

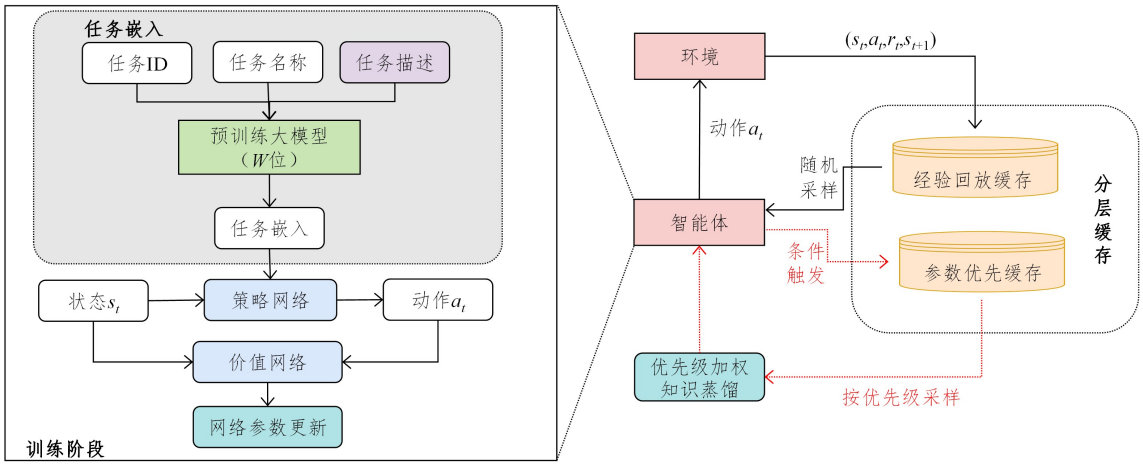


图 2 SM-PHT 算法的流程图

Fig. 2 Flowchart of SM-PHT algorithm

4.1 优先级加权知识蒸馏

为了突破传统知识蒸馏依赖单一教师模型的局限性，创新性地引入了优先级加权知识蒸馏机制(Priority-Weighted Knowledge Distillation，PWKD)。PWKD 通过加权融合多个高性能教师模型的知识，不仅提升了学生网络的鲁棒性和稳定性，还有效缓解了过拟合问题，实现了更高效的知识转移。

知识蒸馏由 Hinton 等在 2015 年提出[19]，用于将复杂的“教师”网络中的知识转移到紧凑的“学生”网络中，这种方法已成为知识迁移和模型压缩的关键工具。图 3 展示了知识蒸馏的基本原理和工作流程。

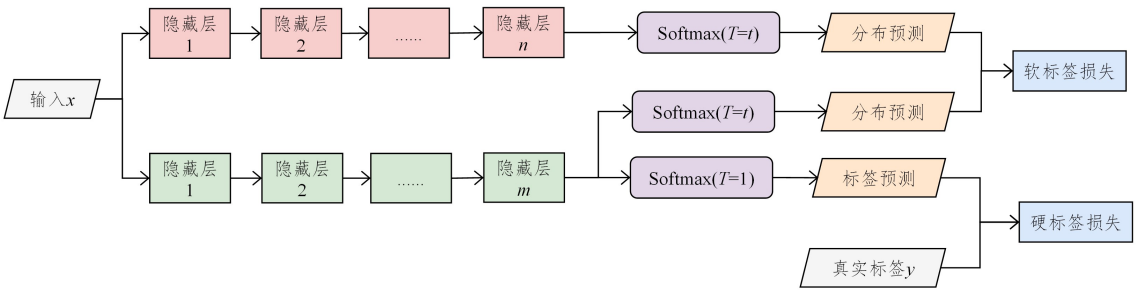


图 3 知识蒸馏原理图

Fig. 3 Diagram of knowledge distillation

从数学角度来看，在蒸馏过程中，教师网络 T 和学生网络 S 共同为输入样本 x 生成预测，目标是通过最小化一个复合损失函数来优化学生网络的表现。该损失函数包含两个部分：硬标签损失 $L_{\text{hard}}(S(x), y)$，用于衡量学生网络预测与真实

标签的差异；软标签损失$L_{soft}(S(x),T(x))$，用于衡量学生网络与教师网络预测分布的差异。其中，y表示真实标签。

$$L_{hard}(S(x),y)=-\sum_i y_i \log(S(x)_i) \tag{4}$$

$$L_{soft}(S(x),T(x))=-\sum_i T(x)_i \log(S(x)_i) \tag{5}$$

最终的知识蒸馏损失函数是硬标签损失与软标签损失的加权总和。λ是一个超参数，用于调控硬标签损失与软标签损失之间的平衡关系。

$$L_{distill}=L_{hard}(S(x),y)+\lambda L_{soft}(S(x),T(x)) \tag{6}$$

此外，自蒸馏(Self-Distillation，SD)是知识蒸馏的一种变体，其独特之处在于模型在同一过程中既充当教师也充当学生。图4不仅提供了知识蒸馏与自蒸馏核心原理的直观解释，还清晰地展示了两者之间的差异与联系。

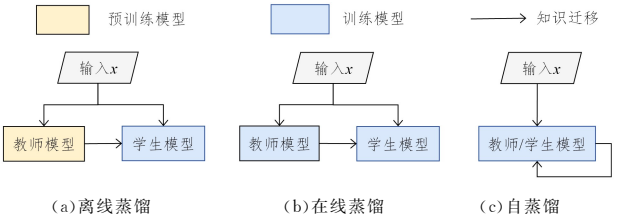


图4 知识蒸馏和自蒸馏的对比

Fig. 4 Comparison of knowledge distillation and self-distillation

在自蒸馏机制中，学生网络$S(x)$的预测结果不仅用于指导其自身的学习过程，还作为教师网络的基准输出。因此，无论是硬标签损失还是软标签损失，均基于学生网络自身的预测结果进行计算。其数学表达形式如下：

$$L_{hard}(S(x),y)=-\sum_i y_i \log(S(x)_i) \tag{7}$$

$$L_{soft}(S(x),S(x))=-\sum_i S(x)_i \log(S(x)_i) \tag{8}$$

$$L_{distill}=L_{hard}(S(x),y)+\lambda L_{soft}(S(x),S(x)) \tag{9}$$

不同于传统方法仅依赖单一前代模型作为教师的做法，PWKD在自蒸馏基础上进行了创新，通过加权蒸馏整合了多个高性能教师模型的知识。这些教师模型的选择基于其训练过程中的性能指标，如平均奖励值或成功率。PWKD为每个教师模型分配权重，表现优异的模型将获得更高的权重，从而更有效地传递多源知识。

图5展示了PWKD的详细流程。从数学角度来看，如果有N个候选教师模型，假设$T_j(x)$表示每个教师模型T_j的输出，P_j表示其性能指标，那么PWKD机制可以描述如下：

$$L_{hard}(S(x),y)=-\sum_i y_i \log(S(x)_i) \tag{10}$$

$$L_{soft}(S(x),T(x))=-\sum_i T(x)_i \log(S(x)_i) \tag{11}$$

$$L_d=L_{hard}(S(x),y)+\sum_{j=1}^{N}\omega_j L_{soft}(S(x),T_j(x)),$$
$$\text{where } \omega_j=\frac{\exp(P_j/\beta)}{\sum_{n=1}^{N}\exp(P_n/\beta)} \tag{12}$$

其中，β是温度参数，用于控制权重分配的集中程度。当$\beta\to\infty$时，权重分配趋近于均匀；当$\beta\to 0$时，权重分配趋近于只选择最佳模型。

这种加权损失机制确保学生网络能够更高效地从表现优异的教师模型中汲取知识，同时通过引入表现较弱但多样化的模型来缓解过拟合问题。因此，PWKD不仅能显著加速收敛过程，还能大幅提升最终模型的性能。

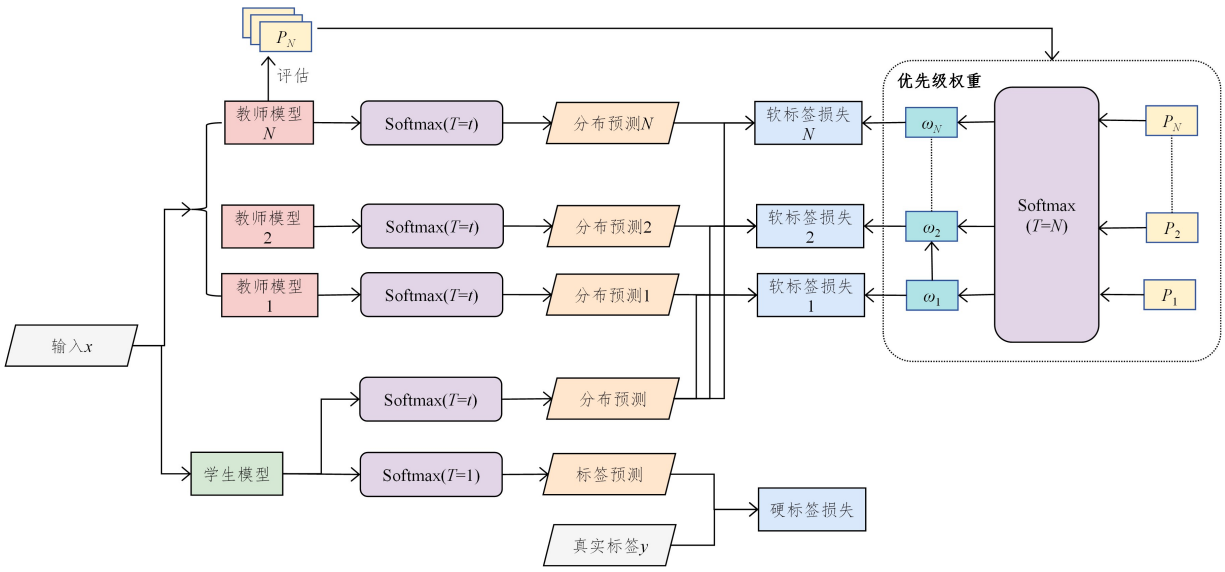


图5 PWKD流程图

Fig. 5 Flowchart of PWKD

4.2 分层缓存

为了进一步优化PWKD的效果，借鉴了互补学习系统(Complementary Learning Systems，CLS)[20]理论，创造性地设计了分层缓存(Hierarchical Buffer，HB)机制。该机制结合了经验回放缓存和参数优先级缓存，如图6所示。分层缓存机制能够动态调整参数的重要性，不仅提升了对重要参数的管理效率，还增强了系统的鲁棒性和灵活性，促进了高效的历史数据利用。

具体来说，经验回放缓存负责收集低层次的经验样本，确保学习过程能够充分利用丰富的历史交互数据，从而增强模型的学习效果和泛化能力。参数优先级缓存则采用优先级队列存储高层次的模型参数，基于性能指标(如平均奖励或成功概率)进行排序。这种设计不仅提升了对重要参数的管理效率，还通过动态调整参数的重要性增强了系统的鲁棒性和灵活性。

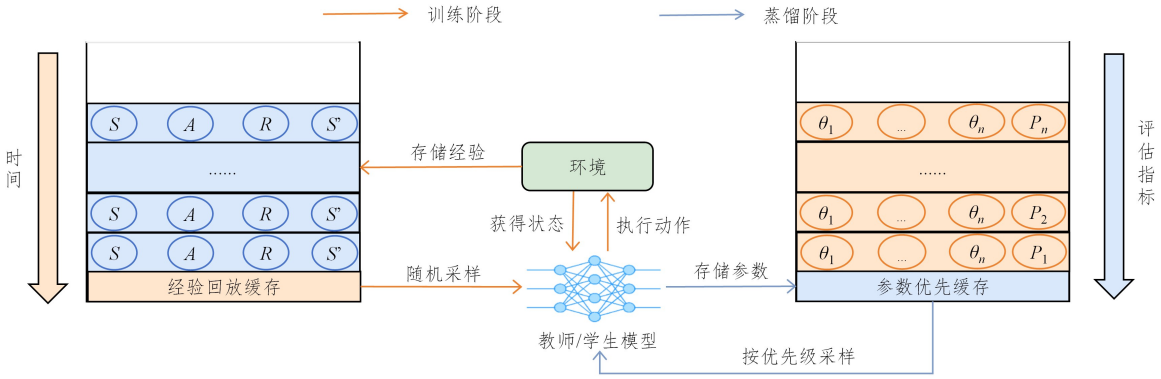


图 6 分层缓存流程图

Fig. 6 Flowchart of hierarchical buffer

4.3 任务嵌入

受到词嵌入技术的启发，创新性地提出了任务嵌入(Task Embedding，TE)机制。与传统的任务标识符(Task Identifier)或独热编码(One-Hot Encoding)不同，TE 机制结合任务标识符和详细的环境信息，利用预训练的大语言模型提取出每个任务的潜在表示，不仅可以精确捕捉任务间的语义关系和上下文信息，还能揭示任务之间的内在联系和共性特征，极大地提升了知识传递效率和多任务处理能力。

相比之下，任务嵌入机制结合任务标识符和详细的环境信息，涵盖任务名称和描述等多维度信息，利用预训练的大语言模型提取出每个任务的潜在表示，不仅能精确捕捉任务间的语义关系和上下文信息，还能揭示任务之间的内在联系和共性特征。表 1 和图 7 共同展示了独热编码与任务嵌入之间的差异，凸显了任务嵌入在捕捉任务间复杂关系方面的优势。

表 1 独热编码和任务嵌入的对比

Table 1 Differences between one-hot encoding and task embedding

任务 ID	任务名称	任务描述	独热编码	任务嵌入
1	sweep	Sweep the table surface by moving the broom	[1,0,0]	[0.94,0.75,0.20]
2	push	Push an object towards a goal position	[0,1,0]	[0.84,0.83,0.76]
3	push with stick	Grasp a stick and use it to pus an object towards a goal position	[0,0,1]	[0.78,0.77,0.59]

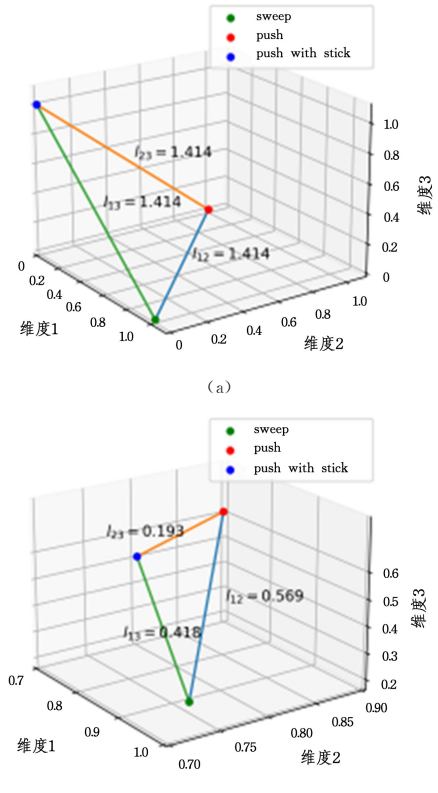


图 7 表 1 例子在向量空间中的可视化结果

Fig. 7 Visualization results of example of Table 1 in vector space

5 理论推导

为了从理论上证明 PWKD 加权蒸馏的优越性，分析权重分配对知识迁移效率的影响，重点比较两种权重分配策略：性能加权和平均加权。

前提 1 假设有 N 个教师模型，每个模型的性能指标为 P_i，其中 $i=1,2,\cdots,N$。

定义 1 在性能加权策略下，权重 ω_i 与教师模型的性能成正比：$\omega_i=\dfrac{\exp(P_i)}{\sum_{n=1}^{N}\exp(P_n)}$。

定义 2 在均匀加权策略下，所有教师模型的权重相等：$\omega_i=\dfrac{1}{N}$。

基于上述的前提和定义，为了证明性能加权机制更好，分析了两种策略下的学生模型的期望性能。假设不考虑样本质量等其他影响因素，则学生模型的性能 P_s 可以表示为教师模型性能的加权平均。

$$P_s=\sum_{i=1}^{N}\omega_i P_i \tag{13}$$

因此，两种策略下的学生模型的期望性能可以表示为：

$$P_s^{\text{perf}}=\sum_{i=1}^{N}\frac{\exp(P_i)}{\sum_{n=1}^{N}\exp(P_n)}P_i \tag{14}$$

$$P_s^{\text{uniform}}=\sum_{i=1}^{N}\frac{1}{N}P_i \tag{15}$$

可以利用 Jenson 不等式来证明 $P_s^{\text{perf}} > P_s^{\text{uniform}}$。

定理 1 对于凸函数 f，有：

$$f\left(\sum_{i=1}^{N}\omega_i x_i\right) \leqslant \sum_{i=1}^{N}\omega_i f(x_i)，当 \sum_{i=1}^{N}\omega_i = 1 \tag{16}$$

在性能加权策略中，函数 $f(x)=\exp(x)$ 是凸函数（当 $\beta>0$ 时）。因此，根据 Jensen 不等式：

$$\exp\left(\beta\sum_{i=1}^{N}\frac{1}{N}P_i\right) \leqslant \frac{1}{N}\sum_{i=1}^{N}\exp(P_i) \tag{17}$$

两边取对数：

$$\sum_{i=1}^{N}\frac{1}{N}P_i \leqslant \ln\left(\frac{1}{N}\sum_{i=1}^{N}\exp(P_i)\right) \tag{18}$$

即：

$$P_s^{\text{uniform}} \leqslant \ln\left(\sum_{i=1}^{N}\frac{1}{N}\exp(P_i)\right) \tag{19}$$

同理可得：

$$P_s^{\text{perf}} \leqslant \ln\left(\sum_{i=1}^{N}\frac{\exp(P_i)}{\sum_{n=1}^{N}\exp(P_n)}\exp(P_i)\right) \tag{20}$$

由于 $f(x)=\exp(x)$ 是凸函数，结合凸函数的性质，可以得出结论：在所有 P_i 相同的情况下，P_s^{uniform} 和 P_s^{perf} 拥有相同的性能下界 $\sum_{i=1}^{N}\frac{1}{N}P_i$；而在相同参数的情况下，$P_s^{\text{perf}}$ 比 P_s^{uniform} 拥有更高的性能上界。

以上通过严谨的数学推理过程证明了性能加权的权重分配策略比均匀加权策略更优，表明 PWKD 能加速学生模型的学习过程并提升其性能。

6 SM-PHT 算法实现与伪代码

PWKD 机制的伪代码如算法 1 所示，任务嵌入机制的具体过程如算法 2 所示，SM-PHT 的具体过程如算法 3 所示。

算法 1 PWKD

输入：教师模型集合 S_{teacher}，学生模型 σ_0，测试输入集合 C_x，教师模型数量 W

输出：蒸馏后的学生模型 σ^*

1. for i=1,2,…,W;
2. x←C_x[i];
3. 根据式(10)和式(11)分别计算 $L_{\text{hard}}(S(x),y)$，$L_{\text{soft}}(S(x),T_i(x))$;
4. end for;
5. 根据式(12)计算 L_{distill};
6. 反向传播优化得到 σ^*.

算法 2 任务嵌入

输入：预训练大模型 LLM，任务标识集合 S_{id}，任务名称集合 S_{name}，任务描述集合 $S_{\text{description}}$，任务数量 M

输出：任务嵌入集合 $S_{\text{embedding}}$

1. 初始化 $S_{\text{embedding}}$←∅;
2. for m=1,2,…,M;
3. 从相应集合中取出 id,name,description;
4. embedding←LLM(id,name,description);
5. $S_{\text{embedding}}$[m]←embedding;
6. end for.

算法 3 SM-PHT

输入：策略网络 θ_0，价值网络 ω_0，经验回放缓存 $B_{\text{experience}}$，参数优先级缓存 $B_{\text{parameter}}$，智能体环境 E，总训练轮次 N，智能体交互次数 STEP

输出：最优策略网络 θ^*，最优价值网络 ω^*

1. 初始化 $B_{\text{experience}}$←∅，$B_{\text{parameter}}$←∅;
2. 根据算法 2 得到 $S_{\text{embedding}}$;
3. for n=1,2,…,N;
4. 随机采样任务 task_m;
5. embedding←$S_{\text{embedding}}$[m];
6. for t=1,2,…,STEP;
7. 得到动作 a_t←$\theta(s_t,\text{embedding})$;
8. s_{t+1}, r_{t+1}←$E(s_t,a_t)$;
9. 将元组 $(s_t,a_t,s_{t+1},r_{t+1})$ 存入 $B_{\text{experience}}$;
10. end for;
11. 从 $B_{\text{experience}}$ 中随机采样一个批次进行训练，记录其性能指标 p;
12. 反向传播 θ←θ_n，ω←ω_n;
13. 将 θ_n，ω_n，p 存入 $B_{\text{parameter}}$;
14. if(n 是指定轮次) then;
15. 从 $B_{\text{parameter}}$ 中取出特定数量的样本作为 S_{teacher};
16. 使用算法 1 分别对 θ_n 和 ω_n 进行 PWKD;
17. end if;
18. end for;
19. 得到 θ^* 和 ω^*.

7 实验设置与结果分析

本章通过一系列实验将 SM-PHT 与主流算法（如 MT-SAC 和 MHMTSAC）进行了对比，在 Meta-world 环境下系统地评估并验证了 SM-PHT 的优越性。

7.1 实验环境

在探索多任务强化学习与元强化学习的过程中，先前的研究常受限于狭窄的任务分布。为此，Yu 等引入了 Meta-World 实验环境[8]，旨在为多任务强化学习和元强化学习提供一个多样且结构化的任务分布平台。Meta-World 包含 50 个不同的机器人操作任务，如抓取和放置物体、开门和按按钮等，所有任务中主体机器人使用相同的机械臂、动作空间和状态空间，但使用不同的客体和任务目标，因此平衡了多样性和共享结构。所有任务的示意图如图 8 所示。

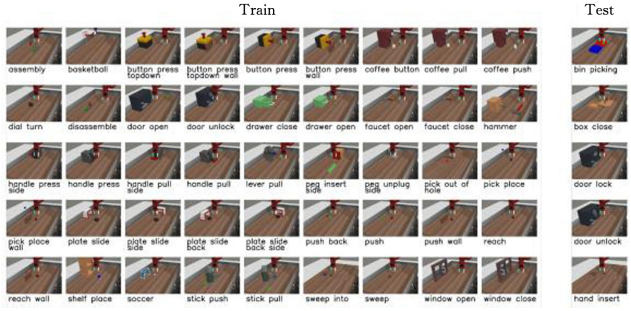


图 8 Meta-World 中的所有任务

Fig. 8 All tasks in Meta-World

7.2 实验设置

Meta-World 提供 MT10 和 MT50 两种训练环境，分别要求智能体执行 10 种和 50 种任务。为增强实验多样性，贴近

实际应用中的动态变化，在原有基础（分别称为 MT10-Fixed 和 MT50-Fixed）上引入条件目标，形成 MT10-Conditioned 和 MT50-Conditioned 环境，以评估算法在动态需求下的适应性和稳定性。

7.3 参数设置

本节将介绍 SM-PHT 框架内的关键参数配置及神经网络架构。

本文设计了两种神经网络：浅层（Shallow-NN）和深层（Deep-NN）。Shallow-NN 包含 2 个隐藏层，每层有 2 个模块，每个模块有 400 个单元；Deep-NN 则拥有 4 个隐藏层，每层同样有 400 个单元的 4 个模块。这些结构用于评估网络深度对性能及任务知识迁移的影响。

此外，“env_name”参数指定实验任务数量，“random_init”决定任务目标是否随机化，它们共同定义了实验环境的具体配置。

7.4 对比算法

为了更加全面地评估 SM-PHT 算法的性能，设置了如下的对比算法。

1）MTSAC（Multi-Task Soft Actor-Critic）：通过共享网络促进不同任务间的信息共享；

2）MHMTSAC（Multi-Head Multi-Task Soft Actor-Critic）：在 MTSAC 的基础上增加了多头机制；

3）Soft-Modularization：软模块化路由算法；

4）SM-PHT：在软模块网络的基础上，引入 PWKD、分层缓存和任务嵌入的改进算法；

5）SM-PHT（PWKD-variant）：只引入 PWKD 和分层缓存机制的变体；

6）SM-PHT（Embedding-Variant）：只引入任务嵌入的变体。

7.5 算法性能对比

为验证 SM-PHT 的有效性，将其与多个主流算法进行对比分析。所有算法在相同环境和参数下运行，以确保实验的严谨性。图 9 和图 10 分别展示了所有算法在 100 个和 500 个训练周期的训练过程。

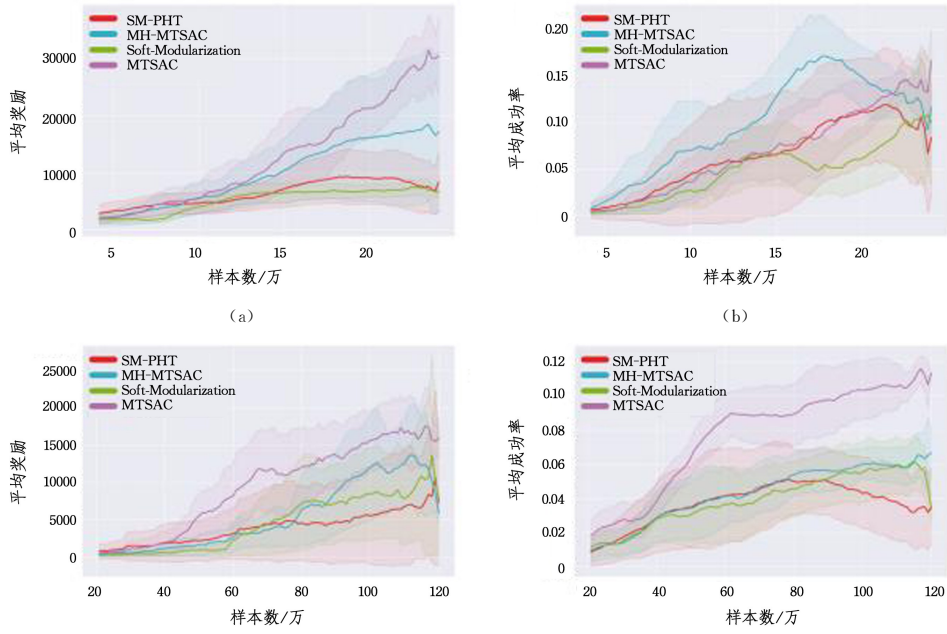


图 9 所有算法在 100 个周期的训练过程比较

Fig. 9 Comparison of training process between different algorithms with 100 epochs

在 MT10 环境经过 100 个训练周期后（见图 9(a)和图 9(b)），MTSAC 和 MHMTSAC 的表现优于 SM-PHT 和 Soft-Modularization，这种优势源于 MTSAC 和 MHMTSAC 的简单网络结构使得它们能够更高效地利用较少的样本进行学习。然而，在 MT50 环境中（见图 9(c)和图 9(d)），随着任务复杂度的增加，尽管 MTSAC 和 MHMTSAC 在奖励和成功率方面仍保持较高水平，但 SM-PHT 和 Soft-Modularization 逐渐缩小了与它们的差距。因为随着任务复杂度的增加和样本数量的增多，SM-PHT 和 Soft-Modularization 能更好地从中受益，逐步缩小了与前者的性能差距。

值得注意的是，经过 500 个训练周期后，结果与 100 个周期完全相反。

在 MT10 环境中（见图 10(a)和图 10(b)），随着样本数量的增加，SM-PHT 的奖励和平均成功率大幅超过其他算法。在 MT50 环境中（见图 10(c)和图 10(d)），MTSAC 和 MHMTSAC 在训练后期的奖励呈现下降趋势。分析认为这是由于其简单的网络结构缺乏有效的知识迁移策略和防止灾难性遗忘的机制，从而引起性能下降。相比之下，SM-PHT 在复杂任务中表现出色，奖励和成功率远超其他算法。

进一步分析图 10(b)和图 10(d)中的平均成功率，SM-PHT 不仅成功率最高，还在稳定性和波动控制方面表现卓越。其他算法的成功率波动较大，表明它们在维持多样化任务的成功率方面面临更多挑战。

此外，表 2 列出了各种算法在训练 500 周期后的各项性能指标。通过定量分析，在 Meta-World 的 MT10 挑战中，SM-PHT 算法的成功率达到了 0.37，是现有最佳方法的两倍左右，平均奖励为 62 057.60，提高了约 54%；在更复杂的 MT50 挑战中，成功率提升至 0.18，比现有最佳方法提升了约 12%，平均奖励增加至 32 625.53，提高了约 9%。这些结果不仅展示了 SM-PHT 算法在复杂任务中的泛化能力，还凸显了其在实际多任务强化学习应用中的巨大潜力。

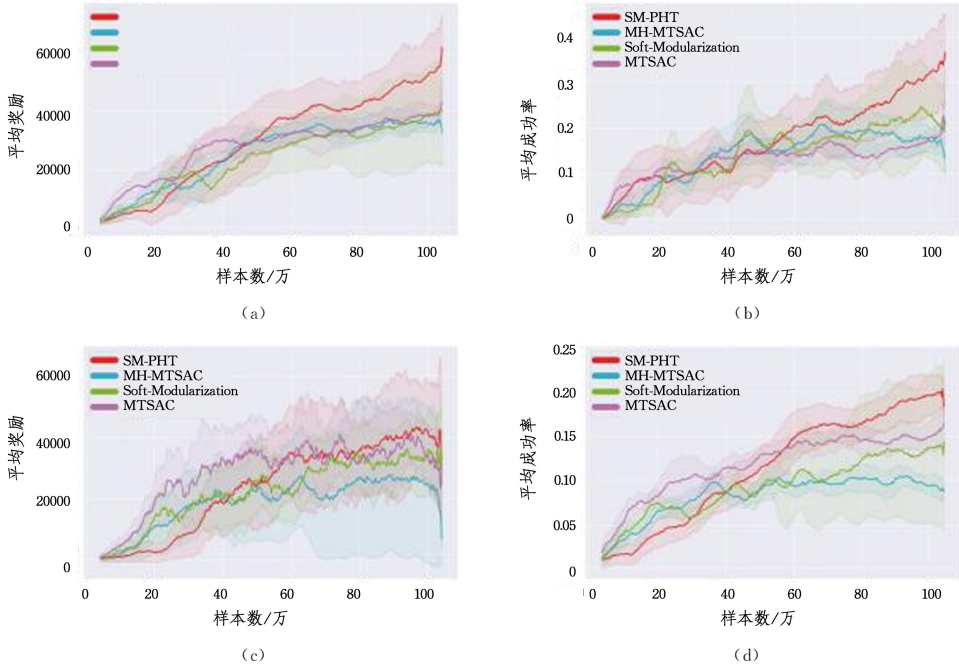


图 10 所有算法在 500 个周期的训练过程比较

Fig. 10 Comparison of training process between different algorithms with 500 epochs

表 2 所有算法在 500 个训练周期的性能比较

Table 2 Comparison of performance between different algorithms with 500 epochs

算法名称	MT10		MT50	
	平均奖励	平均成功率	平均奖励	平均成功率
MTSAC	42 723.36	0.22	24 728.76	0.14
MHMTSAC	32 320.57	0.13	11 728.00	0.08
Soft-Modularation	39 697.31	0.22	30 695.17	0.16
~SM-PHT	**62 057.60**	**0.37**	**32 625.53**	**0.18**
性能提升比例/%	45.25	68.18	6.28	12.50

为了进一步验证 SM-PHT 的稳定性，图 11 展示了各算法在所有任务上的成功率分布。对比结果显示，SM-PHT 在低成功率区间的任务数量最少，而在高成功率区间（60%～80%和 80%～100%）中任务数量较多，表明其在多样化任务中具有稳定的高性能表现。

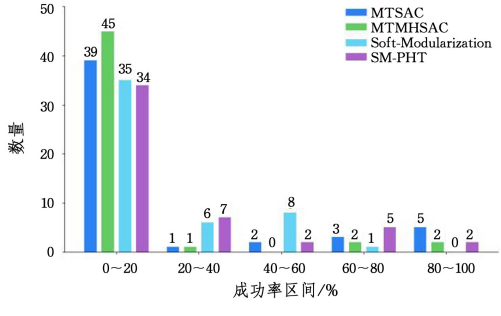


图 11 所有算法在各种任务中的成功率分布

Fig. 11 Distribution of success rates across different algorithms for various tasks

7.6 算法时间复杂度对比

为了全面评估 SM-PHT 方法引入 PWKD、分层缓存和任务嵌入机制后对训练效率的影响，本节进行了详尽的算法复杂度分析，并与传统方法进行了对比。

在 Meta-World 环境下的 MT10 和 MT50 任务上进行了实际测试，记录了每种算法在整个训练过程中的平均 CPU/GPU 使用时间。表 3 对比了 MTSAC，MHMTSAC，Soft-Modularation 和 SM-PHT 这 4 种算法在不同实验环境下的运行时间。从表 3 可以看出，在 100 迭代周期和 500 迭代周期下，SM-PHT 的运行时间分别为 37 327.76 s 和 111 942.74 s，相较于其他算法，虽然在时间上略有增加，但这种增加是有限的。因此，SM-PHT 不仅在性能上超越了现有方法，在资源利用效率方面也表现出色，特别是在处理高维度状态空间和大规模动作空间的任务时，其优越性更加明显。

表 3 所有算法在不同实现环境下的运行时间比较

Table 3 Comparison of runtime of all algorithms

(s)

算法名称	100 迭代周期	500 迭代周期
MTSAC	32 760.15	104 775.28
MHMTSAC	33 297.325	105 548.45
Soft-Modularation	38 625.83	113 339.47
SM-PHT	37 327.76	111 942.74

综上所述，尽管 SM-PHT 引入了一些额外的计算步骤，但通过合理的架构设计和优化策略，它能够在不大幅增加资源消耗的前提下，显著提升多任务强化学习的效果。

7.7 消融实验

首先，为了考查 PWKD 和分层缓存机制在 SM-PHT 中

的效果，设计了以下消融实验，对 MTSAC、MHMTSAC、包含 PWKD 的变体（PWKD-variant）和不包含 PWKD 的变体（Soft-Modularization）进行了比较。所有模型使用相同的网络结构和超参数，以确保比较的公平性。

为了直观评估 PWKD 和分层缓存机制对多任务挑战中智能体性能的影响，实验选择了 10 个任务（包括 4 个不相关任务和 6 个相关任务），通过雷达图可视化智能体的成功率和奖励，如图 12 所示。结果显示，PWKD 变体的表现优于其他算法，因为在雷达图中橙色多边形代表的 PWKD 变体覆盖了其他算法的区域，表明 PWKD 变体保持高水平表现。

具体来说，PWKD 变体在所有任务中展现了更为均衡的性能。例如，在“按钮按压”任务中，PWKD 变体相比其他方法获得了更高的成功率和奖励，这一现象也在“拔出侧边插销”“扫入”和“足球”等任务中得到验证。此外，在“关闭水龙头”和“打开水龙头”这两个相关任务中，PWKD 变体保持了高水平的表现，表明其在促进相关任务间知识迁移方面具有显著优势。类似的趋势也在其他相关任务组中被观察到。

通过上述消融实验，可以得出结论：在不相关任务中，PWKD 机制有效防止了任务间的相互干扰，避免了先前学到的技能对当前任务的阻碍；而在相关任务中，PWKD 促进了高效的知识迁移，使智能体能够基于已有技能进行扩展，并适应新的挑战。

此外，为了评估任务嵌入机制在 SM-PHT 算法中的影响，进行消融实验，对 MTSAC、MHMTSAC、包含任务嵌入的变体（Embeddings-variant）和不包含任务嵌入的变体（Soft-Modularization）进行了比较。所有模型使用相同的网络结构和超参数，以确保比较的公平性。为了更好地进行比较，仅选择了各算法中成功率大于 50% 的任务进行可视化，如图 13 所示。

具体而言，SM-PHT 在所有算法中的表现尤为突出。首先，在分布均匀性方面，SM-PHT 在高奖励值和高成功率区间表现出更为均衡的分布，显示出其对广泛任务复杂度的适应能力。相比之下，MTSAC 和 MHMTSAC 等算法的数据点更多集中在较低的奖励值和成功率区间，表明它们在处理高度复杂任务时存在局限性。

此外，SM-PHT 在其椭圆表示中展现出更低的方差，表明其在不同任务间具有更高的稳定性和可靠性。这凸显了任务嵌入机制在减少任务间干扰和促进高效知识迁移方面的有效性。

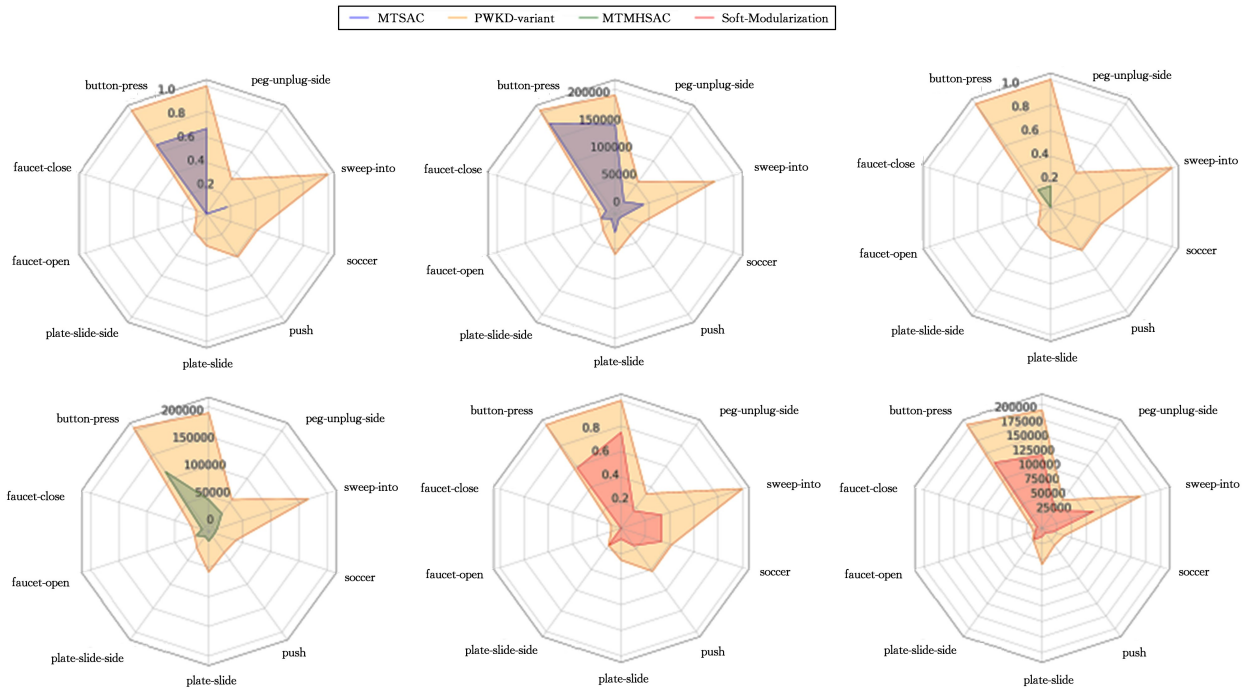


图 12 PWKD 机制消融实验的可视化结果（电子版为彩图）

Fig. 12 Visualization results of ablation experiments on the PWKD mechanism

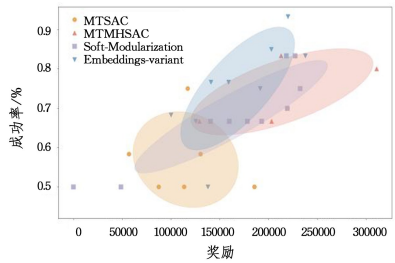


图 13 任务嵌入机制消融实验的可视化结果

Fig. 13 Visualization results of ablation experiments on the task embedding mechanism

7.8 动态目标实验

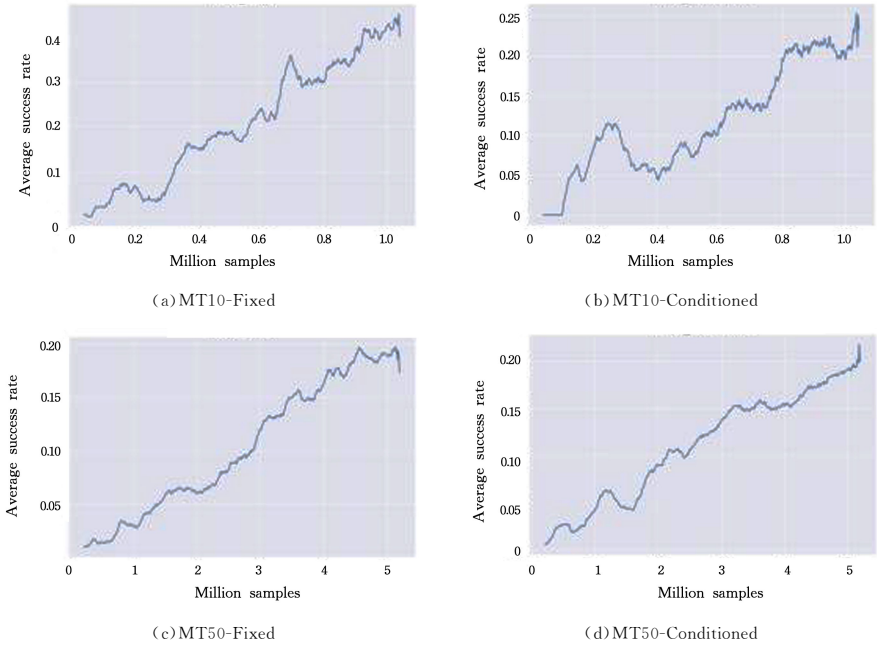
为了验证 SM-PHT 方法在动态目标切换场景下的适应性改进，扩展实验设置，在 MT10-Fixed 和 MT50-Fixed 实验环境的基础上引入了条件目标，分别形成了 MT10-Conditioned 和 MT50-Conditioned 实验环境，并测试了 SM-PHT 在所有上述环境下的表现。实验的可视化结果如图 14 所示。

具体来看，图 14(a)显示在 MT10-Fixed 环境下，随着样本数量从 0 增加到 100 万，算法平均成功率从接近 0 逐渐上升至约 0.4，表明 SM-PHT 在较小规模的固定任务中能够快速学习并提高成功率。图 14(b)则展示了 MT10-Conditioned

环境下的结果。同样地，随着样本数量的增加，平均成功率从接近 0 逐步提升至约 0.25，尽管增长速度略慢于固定任务，但依然显示出显著的性能提升。

在更大规模的 MT50 环境中，图 14(c)和图 14(d)分别对应 Fixed 和 Conditioned 任务。图 14(c)中，平均成功率从初始的接近 0 缓慢上升至约 0.2，这表明在更多任务的固定环境下，SM-PHT 的学习过程相对更为复杂，但仍然能够实现稳定的提升。值得注意的是，在图 14(d)中，MT50-Conditioned 环境下的平均成功率也从接近 0 逐渐增至约 0.2，其最终性能已接近在 MT50-Fixed 中所达到的稳定水平。

因此，SM-PHT 在不牺牲显著性能的前提下，成功将适用场景从静态任务扩展到更具挑战性的动态条件任务，为实际应用中频繁变化的操作需求提供了可行的技术路径。



(a) MT10-Fixed (b) MT10-Conditioned

(c) MT50-Fixed (d) MT50-Conditioned

图 14 动态目标实验的可视化结果

Fig. 14 Visualization results of dynamic target experiments

结束语 本文提出了优先级加权软模块化方法，通过整合优先级加权知识蒸馏、分层缓存和任务嵌入 3 项关键技术，解决了传统多任务强化学习方法在动态环境或多任务情境下适应性不足的问题。实验表明，SM-PHT 在 MT10 和 MT50 环境中显著提升了成功率和平均奖励，尤其在复杂任务中表现出更高的稳定性和更低的波动性。这些成果不仅证明了 SM-PHT 在动态环境中的优越性能，还为机器人操作等领域提供了新的技术手段。未来工作将优化 SM-PHT 框架，针对不同领域的特定需求进行调整和扩展，以实现更广泛的应用和更高的性能提升。我们相信，SM-PHT 及其改进版本将在推动自动化与智能化进程中发挥重要作用。

参 考 文 献

[1] SILVER D, HUANG A, MADDISON C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. Nature, 2016, 529(7587): 484-489.

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

[3] SAUNDERS W, SASTRY G, STUHLMUELLER A, et al. Trial without error: Towards safe reinforcement learning via human intervention[J]. arXiv: 1707.05173, 2017.

[4] PENG Z, LI Q, LIU C, et al. Safe driving via expert guided policy optimization[C]// Conference on Robot Learning. PMLR, 2022: 1554-1563.

[5] LILLICRAP T P. Continuous control with deep reinforcement learning[J]. arXiv: 1509.02971, 2015.

[6] GU S, HOLLY E, LILLICRAP T, et al. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates[C]// 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017: 3389-3396.

[7] YU H, LIANG Y, ZHANG, et al. Terrain-Adaptive Imitation Learning Method Based on Multi-Task Reinforcement Learning [J]. Journal of Data Acquisition and Processing, 2024, 39(5): 1182-1191.

[8] YU T, QUILLEN D, HE Z, et al. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning[C]// Conference on Robot Learning. PMLR, 2020: 1094-1100.

[9] LUO Y T, XUE Z C. Multi-Task Assisted Driving Strategy Learning Method for Autonomous Driving[J]. Journal of South China University of Technology (Natural Science Edition), 2024, 52(10): 31-40.

[10] ISHIHARA K, KANERVISTO A, MIURA J, et al. Multi-task learning with attention for end-to-end autonomous driving[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 2902-2911.

[11] PEREZ-LIEBANA D, LIU J, KHALIFA A, et al. General video game ai: A multitrack framework for evaluating agents, games,

and content generation algorithms[J]. IEEE Transactions on Games, 2019, 11(3): 195-214.
[12] ZHANG J, GUO B, DING X, et al. An adaptive multi-objective multi-task scheduling methodby hierarchical deep reinforcement learning[J]. Applied Soft Computing, 2024, 154: 111342.
[13] LIU W, TANG X, ZHAO C. Distractor-aware tracking with multi-task and dynamic feature learning[J]. Journal of Circuits, Systems and Computers, 2021, 30(2): 2150031.
[14] RUSU A A, RABINOWITZ N C, DESJARDINS G, et al. Progressive neural networks[J]. arXiv: 1606.04671, 2016.
[15] ALJUNDI R, CHAKRAVARTY P, TUYTELAARS T. Expert gate: Lifelong learning with a network of experts[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 3366-3375.
[16] SHEN S, HOU L, ZHOU Y, et al. Mixture-of-experts meets instruction tuning: A winning combination for large language models[J]. arXiv: 2305.14705, 2023.
[17] ROSENBAUM C, KLINGER T, RIEMER M. Routing networks: Adaptive selection of non-linear functions for multi-task learning[J]. arXiv: 1711.01239, 2017.
[18] YANG R, XU H, WU Y, et al. Multi-task reinforcement learning with soft modularization[J]. Advances in Neural Information Processing Systems, 2020, 33: 4767-4777.
[19] HIN TON G, VINYALS O, DEAN J. Distilling the knowledge in a neural network[J]. arXiv: 1503.02531, 2015.
[20] KUMARAN D, HASSABIS D, MCCLELLAND J L. What learning systems do intelligent agents need? Complementary learning systems theory updated[J]. Trends in Cognitive Sciences, 2016, 20(7): 512-534.

PAN Jiahao, born in 2001, postgraduate, is a member of CCF(No. Z1979G). His main research interests include deep learing reinforcement learning, and multi-task reinforcement learning.

FENG Xiang, born in 1977, Ph.D, professor, is a member of CCF (No. 16665M). Her main research interests include reinforcement learning, distributed swarm intelligence, evolutionary computing, and big data intelligence.

(责任编辑:柯颖)