/Author (Homer Simpson) /Title (Robots: Our new overlords) /CreationDate (D:20101201120000) /Subject (Robots) /Keywords (Robots;Overlords)

CARoL: 机器人学习的情境感知适应

Zechen Hu, Tong Xu, Xuesu Xiao, and Xuan Wang

***Abstract*—使用强化学习（RL）从零开始学习新的机器人任务通常效率不高。利用先验知识有可能显著提高学习效率，但这也提出了两个关键挑战：如何确定现有知识的相关性，以及如何自适应地将其整合到学习新的任务中。在本文中，我们提出了一种名为 Context-aware Adaptation for Robot Learning (CARoL) [1] 的新框架，旨在通过先验知识有效地学习一个相似但又不同的新任务。CARoL 通过分析系统动态中的状态转变来实现上下文感知，以识别新任务与先验知识之间的相似性。然后，它利用这些识别出的相似性为新任务优先和调整特定的知识片段。此外，CARoL 具有广泛的适用性，涵盖基于策略的、基于值的和演员-评论者的 RL 算法。我们在模拟的机器人平台和实际的地面车辆上验证了 CARoL 的效率和通用性。模拟环境包括 CarRacing 和 LunarLander，其中 CARoL 在学习新任务的策略时表现出更快的收敛和更高的奖励。在实际实验中，我们展示了 CARoL 如何使地面车辆能够快速有效地调整在模拟中学到的策略，以顺利通过现实世界的越野地形。**

近年来，强化学习（RL）方法在动态环境中的高级机器人控制和复杂任务学习方面取得了显著的成功，使其在各种领域的应用成为可能，例如自主导航 [? ?] 、操控 [? ?] ，以及人机交互 [?] 。尽管取得了这些进展，RL 方法通常对计算要求很高，因为它们依赖于反复的试错探索以发现高回报的结果。

知识融合 [?] 和适应 [? ?] 提供了有前景的方法来解决强化学习效率低下的问题。它们利用先前探索过的任务中的知识（如已学习的控制策略、近似的价值函数等）来加速新任务的训练，消除了在每种情况下从头开始训练的需要。例如，考虑如图 ?? 中所示的车辆在非常复杂的越野地形中导航。假设车辆已经在几个现有环境中进行了大量训练，它理想情况下应能够利用以前学习的知识来适应新的地形类型。

我们的目标是利用先验知识来提高新任务的学习效率。现有的无差别知识融合方法 [?] 没有考虑（并利用）以前和新环境之间的关系，即对环境背景不了解，即使许多技能是特定于上下文的。相反，机器人应该有选择地优先利用与新任务相关的知识/技能。这提出了两个关键挑战：如何有效地量化现有知识与新任务的相关性，以及如何自适应地将其整合到学习过程中。

在这项工作中，我们提出了面向机器人学习的上下文感知适应框架（CARoL），该框架利用上下文感知来使用先验知识高效适应新任务。CARoL 引入了一种新的方法，通过使用状态转换表示作为上下文指标来量化任务相似性，这不同于其他通常使用环境或机器人表示的上下文感知工作。这一选择的理由是状态转换是马尔可夫决策过程（MDP）的核心，强化学习正是建立在此之上的。状态转换本质上捕捉了环境和机器人综合作用的效果，因此是任务相似性的更直接和全面的衡量标准，无需仔细选择来自环境或机器人的特征。例如，陡坡和部分冰面在环境特征上可能差异显著，但可以产生相似的状态转换，如轮胎牵引力损失。通过利用这一共享上下文，可以以更普遍和稳健的方式调整控制策略。

[1]代码将在 GitHub 上公开可用。

CARoL 由三个步骤组成。在第一步中，我们专注于在现有环境中获取先验知识。不同于传统的多任务学习，它仅侧重于为不同任务训练教师策略，我们的方法额外训练状态转移表示来捕捉每个任务的“上下文”。第二步涉及通过将每个状态转移表示与新任务的状态转移进行比较来评估上下文相似性。最后，我们使用相似性来优先选择最合适的上下文知识进行适应，从而能够高效学习新任务。这种适应适用于基于策略、基于价值和演员-评论家 (actor-critic) 的强化学习算法。总结起来，CARoL 的贡献有三方面：

- CARoL 显式地利用状态转换来识别任务特定的上下文相似性，从而实现更有针对性的学习，而不是对先验知识进行无差别的融合/适应。
- CARoL 具有广泛的适用性，可以集成到基于策略和基于价值（或结合演员-评论家）的强化学习算法中，以实现知识适应。
- 我们的实验研究证明了 CARoL 的有效性，不仅在模拟环境中，而且通过在实际的越野导航任务中进行验证。

I. 相关工作

利用先验知识与知识融合密切相关，例如多任务强化学习。在处理新任务时，学习适应性可以提高效率。此外，评估现有知识与新任务的相关性与上下文感知密切相关。在本节中，我们对相关文献进行了详细回顾。

多任务学习通过利用共享知识，旨在实现对多个任务的同时学习，从而提高机器人和强化学习中的样本效率和泛化能力。例如，?] 引入了一个层次贝叶斯框架用于多任务强化学习，通过利用相关任务之间的共享结构来改善知识转移。在此基础上，?] 进一步使机器人能够通过学习一个共享的嵌入空间，在各种任务中获得可迁移的技能。MT-Opt [?] 展示了机器人如何学习广泛的技能范围。对于四旋翼控制任务，?] 利用了共享的物理动态来增强样本效率和任务性能。尽管任务之间的参数共享可以直观地提高数据效率，但在训练过程中，不同任务的梯度可能会相互产生负面干扰。为了解决这个问题，提出了政策蒸馏 [? ?] 和仿演员 [?] 等方法，通过利用多个专家教师的指导来训练单一政策。这些方法在已经遇到的任务或具有已知结构的任务上表现良好。然而，在大多数机器人应用中，机器人经常面临未知任务，这使得在这些不熟悉的场景中确保融合控制策略的性能变得困难。

为了保证在新任务上的性能，知识融合通常需要一个额外的适应过程。为此，各种方法被提出。例如，终身学习维护一个模型，该模型随着时间的推移不断演变以处理所有遇到的任务。这种方法使机器人能够在其整个操作生命周期中不断获取、适应和转移知识。例如，?] 运用了终身学习技术，特别是梯度情节记忆 [?] ，以学习一个与经典运动规划器互补的导航策略，适应新的导航场景而不遗忘以前的场景。?] 基于联邦学习高效获取来自不同环境的

先验知识，并动态调整共享模型，支持机器人快速适应多样的导航任务。同样地， [?] 采用了动态生成和调整网络结构以存储和适应先验知识，利用无监督的知识整合来适应长时间人机交互时新的任务，在一个模拟的人形机器人中。随着大语言模型（LLMs）的发展，[?] 使用 LLMs 作为规划者，将高层指令分解为可执行步骤，使机器人能够借助人类协助适应新任务。终身学习在存储效率上非常高效，因为它仅保留一个模型来解决所有场景。然而，这也使其易受任务顺序偏差和灾难性遗忘的影响。此外，虽然终身学习努力避免遗忘现有知识，但由于梯度的冲突，它可能会对新知识的获取效率产生负面影响。

另一方面，元学习 [?] 采用了一种不同的适应机制。它通过一组从先验知识中得出的初始参数来训练元模型。然后，这个元模型可以适应许多其他模型，每个任务一个。元学习的显著例子包括应用在腿式机器人上的动态变化适应 [?] 和基于优化的方法，如模型无关元学习 (MAML) [?]，它学习能让新任务的策略快速收敛的初始化。在融合先验知识方面，元模型通常是把所有先验知识等同对待，这种方法缺乏根据新任务上下文动态优先处理特定先验知识的能力。

A. 用于强化学习的上下文感知

情境感知学习考虑一组具有相同状态和动作空间但因情境变化而在转移概率和奖励上有所不同的马尔科夫决策过程（MDP）。因此，一个学习模型可以通过使用情境感知作为增强输入来实现自适应 [? ? ? ? ? ?]。例如，在 [?] 的工作中，提出了情境 MDP 来建模人在不同环境下的决策变化。情境感知策略重用（CAPS）[?] 利用情境作为标识符以决定何时以及应该重用哪些源策略。对于机器人应用，[?] 使用潜在情境向量来捕捉机器人特定的结构特征，从而可以预测未来状态。对于不同地形上的四足机器人控制， [?] 和 [?] 将环境编码视为情境，然后作为基准策略的额外参数用于快速适应。具体来说，[?] 直接导出环境编码，而 [?] 则采用额外机制从状态转移中反向推导出这些编码。对于这些现有工作，情境通常表示为环境或机器人在端到端训练过程中生成的潜在编码。相较而言，我们的工作直接使用状态转移作为情境，绕过了对潜在表示的需求。此外，尽管现有方法主要依赖于数据驱动机制来映射情境对模型输出的影响，我们的方法明确整合情境以引导知识适应，这提供了一种更可解释和结构化的方式来提高机器人学习效率。

II. 预备知识和问题表述

在本节中，我们介绍了马尔可夫决策过程（MDP）的基本原理，并定义了解决 MDP 的知识概念。在此基础上，我们构建了上下文感知适应的问题，该问题旨在通过利用先验知识来使机器人高效学习新任务。

A. 马尔可夫决策过程和知识

强化学习任务可以通过马尔可夫决策过程 [?] 来建模，该过程定义为一个元组 $\boldsymbol{T} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}\}$，其中每个元素分别代表状态空间、动作空间、转移概率度量和奖励函数。强化学习智能体的目标是学习一种决策策略，从动作空间 $\mathcal{A}$ 中选择动作，基于状态空间 $\mathcal{S}$ 中的当前状态，以最大化累计奖励 $\mathcal{R}$。在本文中，我们使用一个通用符号 $\mathcal{K}$ 来表示此类决策所需的关键知识。根据所采用的强化学习方法，

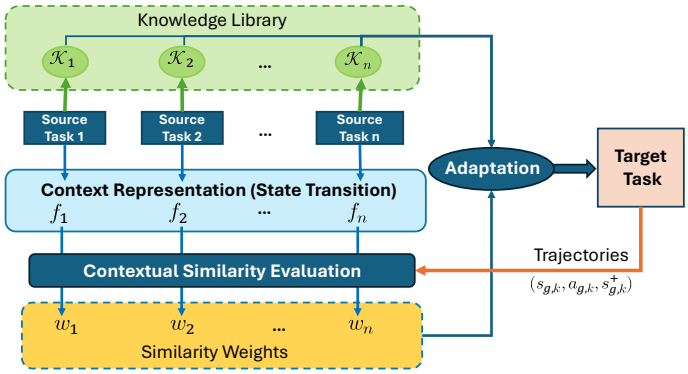

Fig. 1. 情境感知机器人学习适应（CARoL）框架概述。

$\mathcal{K}$ 可以采取不同的形式：对于基于策略的强化学习 [?]，它表示直接将状态映射到动作的策略。对于基于价值的强化学习 [?]，它表示估计行动选择成本的价值函数；对于演员-评论者强化学习 [?]，则结合了策略（演员）和价值函数（评论者）。

本文关注通过利用先前解决任务的知识，高效地适应新任务。考虑一组 n 个源任务，记作 $\mathbb{T} = \{\boldsymbol{T}_1, \boldsymbol{T}_2, \ldots, \boldsymbol{T}_n\}$，其中每个源任务 $\boldsymbol{T}_i = \{\mathcal{S}, \mathcal{A}, \mathcal{P}_i, \mathcal{R}\}$ 对于 $i \in \{1, \ldots, n\}$ 都由共享的状态空间和动作空间（$\mathcal{S}$，$\mathcal{A}$）和一个共同的奖励函数（$\mathcal{R}$）定义，但它们具有不同的转换概率度量（$\mathcal{P}_i$）。一个新的未见目标任务表示为 $\boldsymbol{T}_g = \{\mathcal{S}, \mathcal{A}, \mathcal{P}_g, \mathcal{R}\}$。目标任务与源任务共享相同的状态空间、动作空间和奖励函数，但其转换概率度量 $\mathcal{P}_g$ 是未知的。

该公式适用于具有类似目标的场景，例如具有一致成功指标的机器人导航任务。我们可以为所有场景使用相同的状态和动作空间，但机器人配置（例如底盘、车轮、轮胎等）的变化以及在各种地形（例如几何形状、表面材料等）上的部署将导致转移概率测量的差异。

我们假设源任务可以充分训练以获取相应的知识集 $\mathbb{K} = \{\mathcal{K}_1, \mathcal{K}_2, \cdots, \mathcal{K}_n\}$。我们感兴趣的问题是开发一种有效的算法，通过利用先验知识 $\mathbb{K}$ 来加速学习和提高性能，从而为目标任务 $\boldsymbol{T}_g$ 导出知识 $\mathcal{K}_g$。

III. 机器人学习的情境感知适应（CARoL）

为了解释 CARoL 背后的原理，注意在给定的 MDP 中，转移概率 $\mathcal{P}_i$ 在确定解决问题所需的知识 $\mathcal{K}_i$ 方面起着中心作用。这使得 $\mathcal{P}_i$ 成为知识适应的上下文标记的自然选择：状态转移的相似性越大，源任务关联的知识与新目标任务所需知识之间的相关性就越强。尽管现有的方法通常依赖环境或机器人特征来隐式捕获这种关系，我们的方法明确使用状态转移作为适应的上下文。这提供了一种更具解释性和结构化的方法来提高学习效率。

在下面的部分中，我们将介绍所提出的 CARoL 算法的详细信息。如图 1 所示，CARoL 包括三个关键步骤。

1) 获取每个源任务的上下文表示和先验知识。
2) 评估目标任务与源任务之间的上下文相似性。
3) 执行上下文知识适应，可以应用于基于策略、基于价值或演员-评论家（actor-critic）的强化学习方法。

上下文表示。对于每个源任务 $\boldsymbol{T}_i$，除了使用标准的强化学习来获取知识 $\mathcal{K}_i$ 外，我们还训练了一个近似的转移函

数来表示上下文，用 $f_i(s_i, a_i \mid \phi_i) : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ 表示每个源任务。该函数基于当前状态-动作对预测下一个状态，并通过 ϕ_i 参数化。这些参数通过梯度下降进行更新，用于第 i 个任务，其更新如下：

$$\phi_i' = \phi_i - \alpha \nabla_{\phi_i} \mathcal{L}_{\boldsymbol{T}_i}(\phi_i), \tag{1}$$

其中 α 是学习率，任务通过从机器人轨迹中采样的状态-动作-下一状态三元组 $\{(s_{i,k}, a_{i,k}, s_{i,k}^+)\}$ 进行收集，$k \in \{1, \cdots, m_i\}$ 是样本的指示器。损失函数定义为样本中的次真状态与预测次状态之间的差异：

$$\mathcal{L}_{\boldsymbol{T}_i}(\phi_i) = \sum_{k=1}^{m_i} \| f_i(s_{i,k}, a_{i,k} \mid \phi_i) - s_{i,k}^+ \|^2. \tag{2}$$

语境相似度评估。我们使用状态转换函数 $f_i(\cdot)$ 和 $i \in \{1, \cdots, n\}$ 来评估源任务与新目标任务之间的语境相似度，以此量化它们知识的相关性。为此，一个直接的方法是为新任务学习一个转换函数 $f_g(\cdot)$ ，然后将其与源任务中的 $f_i(\cdot)$ 进行比较。然而，为一个新任务学习 $f_g(\cdot)$ 在计算上可能是昂贵的。

为了解决这个问题，我们的方法使用来自目标任务的采样轨迹 $\{(s_{g,k}, a_{g,k}, s_{g,k}^+)\}$ 和 $k \in \{1, \cdots, m_g\}$ 。这些样本被应用于每一个从源任务中学习到的状态转移函数 $f_i(\cdot)$ 和 $i \in \{1, \cdots, n\}$ ，以计算它们的相似性，即对于源任务 i 来说：

$$\mathcal{Y}_i = \sum_{k=1}^{m_g} \| f_i(s_{g,k}, a_{g,k}) - s_{g,k}^+ \|^2. \tag{3}$$

基于 (3) ，较小的 $\mathcal{Y}_i$ 值表示第 i 个源任务与目标任务之间具有更高的上下文相似性。为了计算所有源任务的相似性，我们将这些度量转换为权重，如 $i \in \{1, \cdots, n\}$,

$$w_i = \frac{\exp(-\mathcal{Y}_i)}{\sum_{j=1}^{n} \exp(-\mathcal{Y}_j)}. \tag{4}$$

上下文感知适应。相似性权重 w_i 使机器人能够优先考虑来自相应源任务的知识。这使得其对目标任务的适应更加集中和有效。在强化学习中，不同的方法例如基于策略的、基于模型的或基于演员评论者的有着不同的学习机制。因此，适应过程也不同，接下来我们将对此进行描述。

我们考虑这样一种情况，其中每个源任务的知识 $\mathcal{K}_i$ 是一个预训练的策略，用 $\pi_i(s) : \mathcal{S} \to \mathcal{A}$ 表示。上下文相似性权重为 w_i 。

设目标策略 $\pi_g(s|\theta_g)$ 由 θ_g 进行参数化，并假设它在每次迭代时以在策略与目标环境上进行交互，以当前状态 $s \in \mathcal{S}$ 和参数 θ_g 生成动作分布 $\pi_g(s|\theta_g)$ 。让 ξ_g 表示策略在目标任务中探索过的轨迹集合，每个轨迹 $\tau \in \xi_g$ 定义为状态-动作对的序列。

为了整合来自源策略 $\pi_i(s)$ 的知识，我们定义了一种学习损失，该损失基于目标策略的动作分布与源策略的动作分布的加权组合之间的散度：

$$\mathcal{L}_P = \sum_{\tau \in \xi_g} \sum_{s \in \tau} \sum_{i=1}^{n} w_i \mathcal{D}(\pi_i(s), \pi_g(s|\theta_g)), \tag{5}$$

其中，$\mathcal{D}(\cdot, \cdot)$ 是 Kullback–Leibler (KL) 散度：

$$\mathcal{D}(\pi_i(s), \pi_g(s|\theta_g)) = \Phi(\frac{\pi_i(s)}{T}) ln \frac{\Phi(\frac{\pi_i(s)}{T})}{\Phi(\pi_g(s|\theta_g)))}. \tag{6}$$

在这里，$\Phi(\cdot)$ 是 softmax 函数，T 是温度参数。通过最小化重新加权的损失 $\mathcal{L}_P$ ，目标策略的参数 θ_g 通过基于梯度的优化迭代更新。此过程有选择性地从相关的源策略中整合知识以优化目标任务，具体总结见算法 1 。

Algorithm 1: 基于策略的强化学习的 CARoL

```
input Knowledge as pre-trained source policies
  {π_1, ..., π_n} ; contextual similarity weights
  {w_1, ..., w_n} ; the target task T_g ;
initialize target policy π_g(s|θ_g) ; a learning rate η ;
for iteration = 1 to K do
    Collect trajectories ξ_g by interacting with the
     target task T_g using π_g(s|θ_g) ;
    for each minibatch of data from ξ_g do
        Compute L_P with equation (5);
        Update: θ'_g = θ_g − η · ∇_{θ_g} L_P ;
    end
end
return target policy π_g .
```

价值驱动的强化学习算法使用价值函数来估计从给定的状态-动作对（或从某个状态）开始的预期累积分数。价值函数用于指导动作选择。在下面的讨论中，我们认为每个源任务的知识 $\mathcal{K}_i$ 是一个预训练的状态-动作值函数 [2] $Q_i(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ 。上下文相似性权重为 w_i 。

设目标值函数为一个由 ψ_g 参数化的神经网络 $Q_g(s, a \mid \psi_g)$ ，它估计目标任务中状态-动作对的值。遵循基于值的强化学习算法中的一般方法，动作是通过 epsilon 贪婪策略为目标任务贪婪地选择的，从早期的探索过渡到后期的利用。在利用期间，我们选择能从 Q 网络中当前状态产生最高 Q 值输出的动作。在探索和利用期间收集的数据存储在重放缓冲区中以便于将来的更新。

一旦重放缓冲区达到足够的大小，我们随机提取一个小批量 $\mathcal{B}$ 的数据。对于这个小批量中的每一对数据，计算目标值。然而，与传统的基于价值的强化学习算法不同，我们不依赖于未充分训练的目标 Q_g –网络进行更新。相反，我们利用源 $Q_i(s, a)$ 函数的知识，通过上下文相似性重新加权，如下所示：

$$Q_{\text{next}} = \sum_{i=1}^{n} w_i Q_i(s^+, \arg\max_{a^+}(Q_i(s^+, a^+))), \tag{7}$$

其中 s^+ 是下一个状态，$\arg\max$ 通过最大化每个源 Q_i 价值函数来选择动作。

使用含有折扣因子 γ 的 Bellman 方程，我们定义了以下的学习损失，其基于目标价值函数和使用重加权的源 Q_i 网络作为先验知识的价值之间的差异：

$$\mathcal{L}_Q = \sum_{s \in \mathcal{B}} \left[\| Q_g(s, a \mid \psi_g) - r - \gamma Q_{\text{next}} \|^2 \right], \tag{8}$$

[2] 注意，类似的机制可以很容易地推广到状态价值函数 $V_i(s)$ 的情况。

，其中 $\mathcal{B}$ 是从重放缓冲区中提取的小批量，而 r 是对应状态-动作对的奖励。

通过最小化重加权损失 $\mathcal{L}_Q$ ，目标值函数的参数 ψ_g 通过下面的算法 2 迭代更新。在算法 2 中总结的过程选择性地结合了相关源值函数的知识，以近似目标任务的值函数。为了测试在 $\boldsymbol{T}_g$ 中学到的 $Q_g(s,a\,|\psi_g)$ ，可以使用贪婪的方法来选择动作。

Algorithm 2: 基于价值的强化学习的 CARoL

input Knowledge as pre-trained source Q networks $\{Q_1, Q_2, \ldots, Q_n\}$; contextual similarity weights $\{w_1, \cdots, w_n\}$; the target task $\boldsymbol{T}_g$;
initialize target value function network $Q_g(s,a\,|\psi_g)$; a replay buffer $\mathcal{M}$; a learning rate η
for *episode = 1 to K* **do**
  Greedy method to choose action a with explorations or $\arg\max_a(Q_g(s,a\,|\psi_g))$ in target task $\boldsymbol{T}_g$ to get next state s' and reward r ;
  Store (s,a,r,s') in $\mathcal{M}$;
  if *the size of* $\mathcal{M}$ *is sufficient* **then**
    Randomly sample a minibatch $\mathcal{B}$ from $\mathcal{M}$;
    Compute $\mathcal{L}_Q$ with equation (8);
    Update: $\psi_g' = \psi_g - \eta \cdot \nabla_{\psi_g}\mathcal{L}_Q$;
  end
end
return *target value function* Q_g .

如果先验知识 $\mathcal{K}_i$ 是以演员-评论家结构的形式，包括策略 $\pi_i(s)$ （演员）和价值函数 $Q_i(s,a)$ （评论家），那么它们都可以被利用来增强新任务的学习。我们专注于学习目标演员（即策略）$\pi_g(s\,|\,\theta_g)$ ，同时采用一种静态融合方法来利用源评论家（价值）函数，而不使用参数化函数逼近，而不是直接结合之前介绍的方法来同时为目标任务学习策略和价值函数。否则，如果演员和评论家同时更新，该方法可能会面临不稳定和对 Q 值的高估等问题，这些问题在演员-评论家强化学习中是常见的 [**?**] 。

类似于算法 1 中概述的基于策略的方法，我们将学习损失的一部分定义为 $\mathcal{L}_P$ ，它测量目标策略的动作分布和源动作分布之间的差异的加权组合。

此外，回忆一下在 actor-critic 方法中，actor π_g 被训练以选择一个动作 $a=\pi_g(s)$ 来针对一个给定的状态 s ，以最大化由 critic 估计的价值。基于这一原则，我们定义一个来自 critic 的损失如下：

$$\mathcal{L}_C = -\sum_{\tau\in\xi_g}\sum_{s\in\tau}\left[\sum_{i=1}^{n} w_i \cdot Q_i(s,\pi_g(s|\theta_g))\right]. \qquad (9)$$

在这个损失函数中，每个状态 s 的动作由当前的 actor π_g 决定。因此，$\mathcal{L}_C$ 使用源 critic 的加权组合来评价目标 actor 的性能，优先考虑那些在情境上与目标任务更相似的 critic。最小化 $\mathcal{L}_C$ 帮助通过利用源 critic 的指导来迭代地改进目标 actor。

在 (5) 和 (9) 中的组合损失得到：

$$\mathcal{L}_{AC} = \mathcal{L}_P + \beta\mathcal{L}_C, \qquad (10)$$

，其中 $\beta\in\mathbb{R}_+$ 是一个可调参数。通过最小化 $\mathcal{L}_{AC}$ ，目标演员 π_g 通过 $\mathcal{L}_P$ 从源演员 $\pi_i(s)$ 以及通过 $\mathcal{L}_C$ 从源评论员 $Q_i(s,a)$ 接收信息。β 的值决定了每个的重要性。此过程确保目标演员有效地整合来自源演员和评论员的知识，以提高目标任务的表现。算法 3 总结了如何使用 CARoL 来进行演员-评论员强化学习。

Algorithm 3: 用于演员-评论家强化学习的 CARoL

input Knowledge as pre-trained source actors $\Pi=\{\pi_1,\pi_2,\ldots,\pi_n\}$ and critics $\{Q_1,Q_2,\ldots,Q_n\}$; contextual similarity weights $\{w_1,\cdots,w_n\}$; the target task $\boldsymbol{T}_g$;
initialize target actor $\pi_g(s|\theta_g)$;
for *iteration = 1 to K* **do**
  Collect trajectories ξ_g by interacting with the target task $\boldsymbol{T}_g$ using $\pi_g(s|\theta_g)$;
  for *each minibatch of data from* ξ_g **do**
    Compute $\mathcal{L}_{AC}$ with equation (5) and (9);
    Update: $\theta_g' = \theta_g - \eta\cdot\nabla_{\theta_g}\mathcal{L}_{AC}$;
  end
end
return *target actor* π_g .

Remark 1. 在本节中，引入的 *CARoL* 专注于利用先验知识来解决新目标任务。它忽略了使用标准强化学习方法从任务中获取新知识的方面。这是因为当先验知识已经足够解决任务时，额外的学习可能并不总是必要的。然而，如果需要从目标任务中学习，这可以通过为策略方法、价值方法和演员-评论家方法分别在函数 (5) 、(8) 和 (10) 中添加标准 *RL* 损失项来实现。这种简单的扩展使模型不仅能够适应先验知识，还能够通过直接从目标任务中学习来改进它。一个实现版本，称为 *CARoL+*，将在实验部分中包含用于比较。

在本节中，我们展示了模拟和物理实验，以评估 CARoL 在利用先验知识和上下文感知解决新任务方面的有效性。

为确保可重复性，模拟实验使用 OpenAI 的 CarRacing 和 LunarLander 任务进行，如图 **??** -(a,b) 所示。为了展示 CARoL 在真实世界场景中的能力（图 **??** -(c)），我们将其部署在地面车辆上，从而实现从源任务学习的策略能够快速高效地适应并顺利导航真实世界的越野地形。在这些设置中，CARoL 与各种 RL 算法集成进行知识适应，其性能与多个基线方法进行比较。

如图 **??** -(a) 所示，CarRacing 是一个连续控制任务，其中自动移动机器人必须在程序生成的赛道上导航，以最大化累积奖励。该任务涉及在赛道上高效前进的同时，平衡速度和控制以避免出轨。

在 CarRacing 中，汽车的动态性能依赖于赛道的摩擦力。我们设计了三个不同的任务环境，分别具有低 $\mu=0.5$ 、中等 $\mu=1$ 和高摩擦力 $\mu=2$ ，作为源任务。我们的目标是在具有新摩擦力水平的目标任务中迅速获得一个策略。对于机器人来说，源任务和目标任务的摩擦力水平都是未知的。

TABLE I
与 CarRacing 中每个源任务的上下文相似性。

	T_1	T_2	T_3
a: $\mu = 0.2$	1.00	0.00	0.00
b: $\mu = 0.5$	0.98	0.02	0.00
c: $\mu = 0.7$	0.83	0.17	0.00
d: $\mu = 1.0$	0.05	0.94	0.01
e: $\mu = 1.3$	0.10	0.64	0.26
f: $\mu = 1.7$	0.06	0.15	0.79
g: $\mu = 2.0$	0.06	0.11	0.83
h: $\mu = 2.3$	0.08	0.13	0.79

1) CARoL 及其他基线的实现：

- CARoL：在 CarRacing 中，我们使用 PPO（近端策略优化）获取所有源知识 [**?**]。虽然 PPO 有一个用于提高训练稳定性的价值函数，但它主要是一个以执行者为中心的方法。源知识被表示为预训练的源策略。因此，我们的 CARoL 框架利用算法 1 适应目标任务。
- 策略蒸馏 (PD) [**?**]：我们在基线中将 PD 作为多任务强化学习的代表方法。PD 的实现是利用经过良好训练的源策略作为教师来训练一个目标策略。PD 和 CARoL 的关键区别在于它们的适应机制：CARoL 利用相似性权重来实现上下文感知的适应，而 PD 则以不加区分的方式均等地融合所有源策略。
- 无关模型的元学习（MAML）[**?**]：我们将 MAML 作为元学习的代表方法纳入我们的基线。元模型使用所有源任务进行训练。然后，使用 PPO 将元模型适配到新环境中。
- 从头学习（LfS）：我们结合了在源知识训练中使用的相同方法，但调整网络结构以匹配 CARoL，从而能够从头开始针对目标任务进行学习。
- 源知识（SK）：我们直接将源知识应用于不同的目标任务。累积奖励值通过对 20 个情节取平均值进行评估。

实现细节：在训练 SKs 时，我们采用基于 CNN 的 actor 网络，该网络由 6 层卷积层组成，通道数逐渐增加，分别为 { 8, 16, 32, 64, 128, 256 }，以及一个全连接层（100 个单元）。actor 网络的输入通过以下步骤获得：首先将（自上而下视角的）RGB 图像转换为灰度图像，然后将当前图像与前三个时间步的图像堆叠在一起。为了使 CARoL 具备上下文感知能力，我们使用一个 4 层的 MLP 学习状态转换函数 { 16, 32, 16, 2 }，隐藏层中使用 ReLU 激活函数。该模型将车辆速度和陀螺仪读数视为状态，转向、加速和制动视为动作。CARoL、PD 和 LfS 分别为目标任务输出具有相同架构的策略：一个较轻的 actor 网络，具有 4 层卷积层 { 8, 16, 32, 64 } 和一个全连接层（50 个单元）。所有模型均使用 Adam 优化器训练，学习率固定为 1×10^{-4}，折扣因子为 0.99。SK 和 MAML 源任务内循环的训练总时间步数设置为 6×10^6 时间步，而 CARoL、PD 和 LfS 的适应阶段限制为 2×10^6 时间步，以实现更快的适应和公平的比较。对于 MAML，我们以 3 个源任务的 meta 批次大小训练 meta 策略，并在 5 个梯度步上以相同的学习速率进行适应。

2) 实验结果与分析： 表格 I 展示了在 CarRacing 模拟实验中，不同目标任务下三个源任务所学习的上下文相似度权重。我们观察到，为给定目标任务分配的源任务的上下文相似度权重通常与它们的摩擦差异呈负相关。换句话说，与目标任务摩擦较为接近的源任务往往会获得更高的相似度权重，反之亦然。特别是，任务 $\mu = 0.5$ 、$\mu = 1$ 和 $\mu = 2$ 对应的情况是目标任务与某个源任务相同。这些任务作为算法正确识别上下文相似度的合理性检查。结果证实了我们方法的有效性，因为相应的源任务始终获得最高的相似度权重。

在训练细节方面，图 **??** 展示了奖励曲线，其中 x 轴代表训练步数，y 轴代表奖励值。我们观察到每个 SK 在其对应任务上表现良好，这表明 SK 已经经过良好训练。在所有情况下，CARoL 与其他方法相比展现了更快的收敛速度。这证明了 CARoL 可以高效地利用现有知识并快速适应目标任务的能力。就最终奖励而言，CARoL 通常实现了最佳表现。具体而言，当目标环境与某个源环境匹配时，如子图 (图 b, d, g) 中所示，CARoL 产生的结果与经过良好训练的源知识 (SK) 策略相当。与不加区分地融合 SK 策略的 PD 相比，CARoL 能够优先考虑相关的 SK，导致在情况 (a, c) 中表现更好。此外，CARoL 表现出更稳定的训练过程，而 PD 则在情况 (a, c) 以及 (b, e, f, g) 中间部分出现显著的波动。

LfS 使用与 CARoL 相同的参数设置。LfS 在大多数任务中难以在给定的训练步骤内收敛，而 CARoL 适应得明显更快。此外，LfS 表现出明显的训练低效趋势，更具挑战性的任务（例如低摩擦环境）收敛需要更长时间。这是因为在低摩擦环境中，车辆在探索时更容易偏离轨道。相比之下，CARoL 不受此问题的影响。关于 MAML，尽管进行了广泛的元训练和适应测试，其表现仍然较差。这可能是由于不理想的元模型难以有效泛化到新环境（如果训练时间足够长，MAML 的奖励将开始增加并与 LfS 匹配）。然而，无论最终表现如何，元训练计算成本高，而 CARoL 不需要这样的过程并且可以直接利用 SK。

如图 **??** -(b) 所示，LunarLander 是一个控制任务，其中一个自主着陆机器人必须在模拟的月球环境中安全降落在指定的着陆平台上。机器人必须平衡推力和控制，以实现平稳着陆，同时尽量减少燃料消耗。

LunarLander 环境允许调整重力和风力，根据这些我们可以创建不同的源任务和目标任务。不同的重力水平 $g \in [-12, -2]$ 影响着着陆机器人上的垂直力，而变化的风力 $F_w \in [0, 10]$ 影响水平力 [3]。此实验的源任务和目标任务配置在图 2 中可视化。横轴代表重力，纵轴代表风力，它们共同展开一个二维配置空间。蓝点表示源任务，定义如下：$(-5, 0)$；$(-10, 0)$；$(-5, 10)$；$(-10, 10)$。红色叉号表示用于测试的目标任务。这些目标任务位于配置空间上由源任务形成的方形的上方、左侧、右侧和内部。注意对于机器人来说，源任务和目标任务的环境配置都是未知的。通过上下文意识，机器人将为每个目标任务推导出一个独特的源知识优先级。

实现细节：为了训练 SK，我们使用一个全连接的 actor 网络，该网络由一个 4 层的 MLP 构成，其层大小为 { 96, 192, 96, 2 }，隐藏层使用 ReLU 激活函数，输出层则使

[3]在我们的实验中，风力被建模为一个时间不变的固定值，以强调其对状态转移的影响。这与标准环境不同，标准环境中每个时间步风力是随机的。

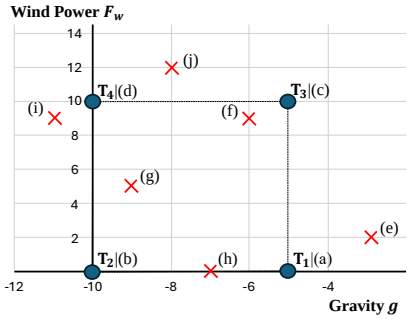

Fig. 2. 月球着陆器中的任务配置。在图中，圆形代表源任务（$\boldsymbol{T}_1$ - $\boldsymbol{T}_4$）；叉号代表目标任务（a-j）。该图直观地展示了源任务与目标任务之间的相似性关系。机器人对环境配置（包括源任务和目标任务）是未知的。

TABLE II
与 LunarLander 中每个源任务的上下文相似性。

	$\boldsymbol{T}_1$	$\boldsymbol{T}_2$	$\boldsymbol{T}_3$	$\boldsymbol{T}_4$
a: (-5, 0)	0.95	0.05	0.00	0.00
b: (-10, 0)	0.01	0.98	0.00	0.01
c: (-5, 10)	0.06	0.01	0.91	0.02
d: (-10, 10)	0.00	0.01	0.01	0.98
e: (-3, 2)	0.62	0.09	0.28	0.01
f: (-6, 9)	0.09	0.00	0.81	0.10
g: (-9, 5)	0.19	0.37	0.18	0.26
h: (-7, 0)	0.35	0.34	0.22	0.09
i: (-11, 9)	0.01	0.12	0.10	0.77
j: (-8, 12)	0.13	0.17	0.29	0.41

用 tanh 激活函数。评论网络使用相同的架构。为了提高 CARoL 的上下文感知能力，我们在 CarRacing 中使用相同的网络来实现状态转换函数。输入包括观测状态和执行的动作，输出是下一个观测状态。对于 CARoL, CARoL+, PD 和 LfS，我们使用相同但更轻的 actor 和 critic 网络，即 { 48, 96, 48, 2 }。所有方法都使用 AdamW 进行训练，固定学习率为 4×10^{-4} ，折扣因子为 0.98 。SK 的总训练集数为 15,000 集，每集中最多 1,000 步，而其他方法为 10,000 集，每集中最多 500 步。

3) 实验结果与分析： 表格 II 显示了在 LunarLander 模拟实验中，不同目标任务中为四种源任务学习到的上下文相似性权重。与 CarRacing 类似，我们观察到权重在图中的几何距离与其呈负相关。我们还在此进行了一项合理性检查，因为表格的前四行对应于与源任务相同的场景。我们可以观察到类似于 CarRacing 的模式。同时，通过查看表中的确切值，我们还注意到图 2 中权重与距离之间的关系并不是严格成比例的，因为环境配置对上下文（状态转换）的影响是非线性的。例如，不同的重力和风力组合可能导致相似的状态转换。然后，上下文相似性使得相应的先验知识能够得到有效适应。

在训练细节方面，图 3 显示了奖励曲线，其中 x 轴表示学习回合数，y 轴表示奖励值。与基线相比，CARoL 显示出更快和更高的奖励收敛。子图 (a-d) 对应于目标任务与某个源任务相同的场景。在这些情况下，我们观察到每个源策略在其相应的源任务上表现最佳，获得最高评价奖励。通过优先选择最具上下文相似性的知识，CARoL 能够达到 (a)、(b) 和 (d) 中的可比累计奖励。对于 (c)，CARoL 的轻微差距可以通过表 II 的第三行为解释，在该行中，分配给 $\boldsymbol{T}_3$ 的权重不像其他情况中接近 1。通过适应更多不相关源任务的知识，性能略有下降。在所有其他目标任务中，即目标任务与任何源任务不同的情况下，我们观察到任何单一源策略的平均奖励始终低于 CARoL。这是因为源策略缺乏对目标任务特定上下文的经验。而 CARoL 则通过根据上下文相似性有效地结合源知识，从而实现更好的性能。

与基线相比，与在 CarRacing 中的观察类似，我们看到 PD 在大多数目标任务中表现出比 CARoL 更低的奖励收敛性。这是因为 PD 仅专注于知识融合而不考虑新任务的上下文相似性。PD 和 CARoL 在初始阶段将多源策略的知识结合时，都表现出一些训练阶段的震荡。然而，随着适应过程的进行，这些震荡会减小。将 LfS 与 CARoL 相比，我们观察到 LfS 表现出显著的较慢的收敛速度，特别是在 (f) 和 (h) 中，并且通常达到较低的最终奖励。这是因为 LfS 和 CARoL 的网络规模相同，但不能利用先验知识。因此，训练对数据的需求量大，收敛困难。值得注意的是，在目标任务 (i) 和 (j) 的一些试验中，LfS 获得了比 CARoL 更高的奖励。这是因为源任务可能没有为目标任务提供足够有用的知识，正如 Fig. 2 中所示，这些目标任务使用的参数位于源任务的方框之外。在这些情况下，LfS 受益于从新任务中学习新知识，从而优于 CARoL。

然而，这个问题可以通过 CARoL+ 来解决。CARoL+ 在适应过程中集成了基于互动的学习，使其能够在所有情况下实现最高奖励。这弥补了 CARoL 在某些情况下相较于 LfS 在最终收敛值方面的潜在限制。此外，CARoL+ 保留了 CARoL 快速收敛的优势，同时在收敛过程中也是最稳定的。

A. 真实世界实验：地面车辆越野导航

我们将 CARoL 应用于一个有实际机器人支持的越野移动问题。如图 **??** -(c) 所示，目标是使机器人能够自主成功地从起点导航到目标。由于收集物理车辆与地形互动数据的成本高昂，现实世界的越野移动中的强化学习效率特别低。因此，在这个实验中，我们的目标是应用 CARoL，以便将模拟中学习的策略适应到现实世界。

我们利用高保真多物理场模拟器 Chrono [**?**]，来训练两种不同的越野导航策略：用于混凝土地形的 SK1 和用于草地地形的 SK2。我们生成具有起伏地形的越野地形，并改变车辆轮胎与底层地形之间的摩擦系数，即 0.9 和 0.4。这里，不同的摩擦水平会影响可通行性。例如，对于相同的坡度，摩擦力决定了机器人能否爬上去或必须绕行。在现实世界中，我们构建了一个包含数百块岩石和巨石的越野移动测试平台。我们认为真实世界中岩石和巨石的摩擦系数为 0.57 到 0.73 之间，这需要将两种 SK 策略适应为新策略。

- 观测空间包括一个 16 维的地形编码（通过 SWAE [**?**] 架构从海拔截面派生得到）、归一化的航向误差和当前速度。
- 动作空间包括连续的速度和转向指令。

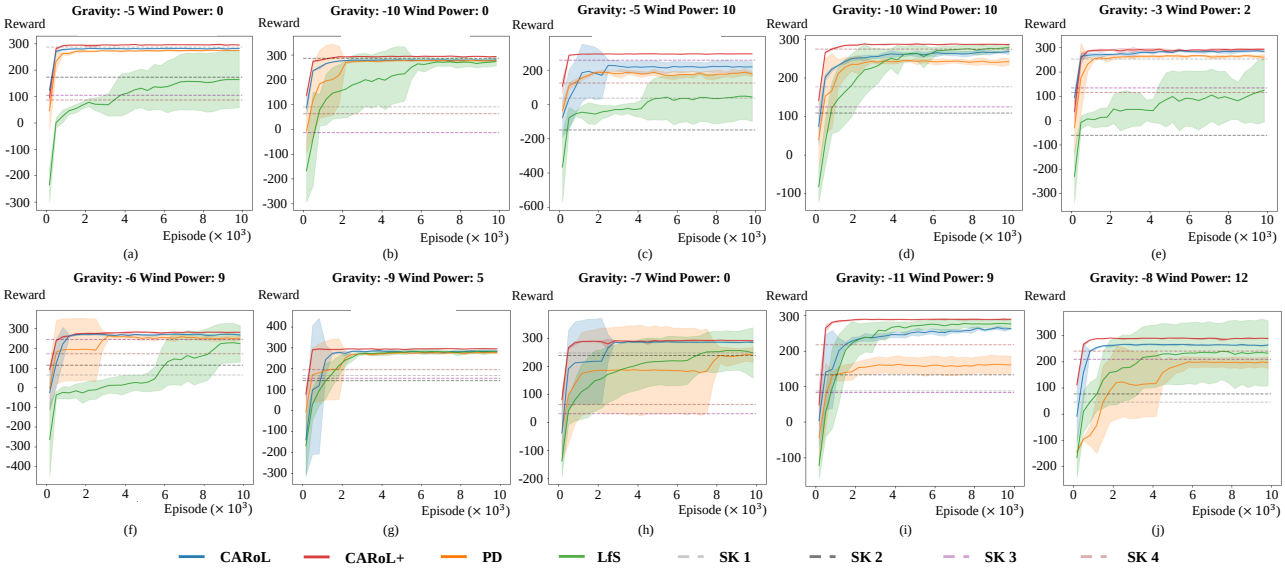

Fig. 3. 在不同的 LunarLander 任务下，对 CARoL 和基线的累积奖励进行比较。x 轴表示运行的回合数，y 轴表示奖励。虚线的高度表示在该任务下源知识（SK）的平均累积奖励。

- 机器人因朝目标移动而获得奖励，并因较大的横滚和俯仰角度而受到惩罚，这些角度是机器人在导航过程中稳定性的指标。

在模拟器中，与 CarRacing 类似，我们使用 PPO 从头训练两个源知识策略，并为每个环境训练状态转移函数作为上下文表示。为了在真实环境中实现 CARoL，我们首先计算上下文相似性。为了实现这一点，我们手动控制车辆随机在真实世界的越野地形上进行 20 次试验，以收集轨迹数据，而不是依赖随机探索，以提高真实世界的样本效率。上下文相似性通过比较在现实环境中收集的轨迹上的源任务转移函数的预测误差来量化。

基于获得的上下文相似性和源任务策略，我们采用 CARoL 算法 1 来适应真实世界。由于适应是在策略上进行的，我们迭代收集使用当前策略的轨迹，并使用收集的数据更新策略。策略模型实现为一个具有隐藏层 $\{32, 32, 2\}$ 的 3 层全连接网络，并使用 ReLU 激活函数。我们使用 Adam 优化器，固定学习率为 1×10^{-3} 。折扣因子设置为 0.99。我们总共进行了 10 次迭代，每次包括 8 次导航试验（总时间约为 130 分钟），期间车辆逐渐从最初的失败（偏离实验场地、中途卡住或翻车）到成功到达目标。在最后两次迭代中，车辆在每个回合中都成功到达目标，表明适应性能良好。然后，我们将这一收敛模型与源知识策略进行比较。由于收集大量车辆与地形交互数据的高昂成本，我们没有实现基准方法。出于同样的原因，我们也没有使用 CARoL+。

在对比实验中，每个模型从起点到目标进行五次试验。源知识指的是在 Chrono 模拟器中训练的模型，直接部署在机器人上。本次实验的评估指标与 RL 奖励一致，包括成功率、横滚和俯仰。虽然没有评估行驶时间，但在表格中提供了作为参考。

如结果表 III 所示，CARoL 达到最高的成功率，机器人在 CARoL 的适应模型下表现出最稳定的导航。尽管 CARoL 到达目标的时间最长，但这是预料之中的，因为我们在该任务中没有对行驶时间施加任何奖励或惩罚。较长的时间可能表明机器人正在根据可通过性仔细选择路径。主要目标是确保机器人自动成功到达目标，而 CARoL 在横滚和俯仰方面的高稳定性在实现这一结果中起着关键作用。SK2 在低摩擦草地上训练，得益于现实世界岩石和巨石的增加摩擦，达到了最短的行驶时间。

TABLE III
物理实验结果。

Method	Success ↑	Roll ↓	Pitch ↓	Time (no penalty)
CARoL	5/5	4.08°± 4.65°	8.55°± 5.13°	16.88s ± 1.97s
SK1	3/5	5.44°± 7.13°	10.35°± 5.29°	11.78s ± 1.44s
SK2	2/5	5.75°± 5.00°	8.61°± 5.12°	8.55s ± 0.67s

IV. 局限性

CARoL 的当前局限性包括依赖于异步处理上下文感知和适应性。具体来说，适应是在计算上下文权重之后执行的，因此，当前的 CARoL 无法实时适应动态上下文。一个潜在的解决方案是在目标环境中部署强化学习策略期间，同时实时计算相似性权重并适应源知识。此外，CARoL 严重依赖于源知识的质量。引入一个额外的机制来评估源知识的质量可能会解决这一局限性。这样的机制将使 CARoL 能够不仅根据上下文相似性，还根据源知识的可靠性来优先处理任务。这样将导致更加稳健和有效的任务适应。

在本文中，我们介绍了 CARoL。CARoL 是一个框架，首先根据源任务和目标任务的状态转换来评估它们之间的上下文相似性。根据源任务的先验知识，CARoL 然后进行目标任务的上下文知识适应。CARoL 的一个关键优势是其广泛适用于基于策略、基于价值和演员-评论员的强化学习方法。实验结果表明，我们的算法能够有效地适应新任务。我们验证了 CARoL 在两个开源模拟环境和一个物理机器人平台上的性能，CARoL 成功将模拟中学习的先

验知识转移到现实世界的越野环境中。对于未来的工作，我们计划将上下文相似性评估同步整合到适应过程中。此外，我们计划将 CARoL 扩展到多代理环境中，通过多代理协调来实现任务适应更有效的上下文表示。