arXiv:2302.06540v1 [cs.AI] 13 Feb 2023

基于自举对比学习的从观察中模仿

MedricB. Djeafea Sonwa蒙特利尔大学与 Mila
medric.sonwa@umontreal.ca

Johanna Hansen麦吉尔大学与 Mila
johanna.hansen@mail.mcgill.ca

Eugene Belilovsky康考迪亚大学与 Mila e
ugene.belilovsky@concordia.ca

摘要

从观察中模仿（IfO）是一种学习范式，其通过观察专家示范，在无法获取专家动作的情况下，于马尔可夫决策过程（MDP）中训练自主智能体。这些示范可以是环境状态序列，也可以是环境的原始视觉观测。近期从观察中模仿的研究主要关注低维环境状态观测这一情形；然而，实践中不太可能获得这类高度特定的观测。本文采用一种具有挑战性但更贴近现实的问题设定：仅凭专家完成任务的视觉演示，学习能够在已学习的潜在空间中运行的控制策略。我们提出 BootIfOL，这是一种从观察中模仿算法，旨在学习一个奖励函数：该函数接收智能体轨迹并将其与专家轨迹进行比较，根据智能体行为与隐含目标的相似程度给予奖励。我们将这一奖励函数视为智能体行为轨迹与专家轨迹之间的距离度量，并通过对比学习来学习该函数。对比学习目标旨在准确表征专家轨迹，并使其与非专家轨迹保持距离。对比学习中使用的非专家轨迹集合，通过从智能体使用当前奖励函数进行强化学习所得的滚动采样中自举，逐步变得更加复杂。我们在多种控制任务上评估了所提方法，结果表明，仅使用数量有限的示范轨迹便能训练出有效策略，显著优于采用原始观测的先前方法。

1 引言

从观察中模仿（IfO） [25, 17] 涉及学习一个策略函数，使智能体能够利用专家完成同一任务的一组示范来解决该任务。与模仿学习（IL） [12, 20, 18, 2], 不同，在从观察中模仿中，示范仅由专家对环境的观测序列构成。根据环境以及智能体可获得的信息，观测可以是对特定时刻环境的描述（如关节角度、物体间距离、方向向量坐标、速度等），以环境低维状态向量的形式提供；也可以是未经处理、需要进一步分析和处理的环境原始视觉信息。从观察中模仿这一形式化方法的优势在于，它使更自然的智能体学习场景成为可能，因为在这些场景中，难以获取精确的动作数据。此外，它也更准确地近似了人类在学习过程中模仿专家的方式。文献中用于训练从观察中模仿任务智能体的两种主要方法是对抗方法和奖励学习方法 [28]。对抗方法 [26, 27] 借鉴了逆强化学习 [10], 的思想，提出了一种类生成对抗网络架构 [6] ，其中智能体是一个

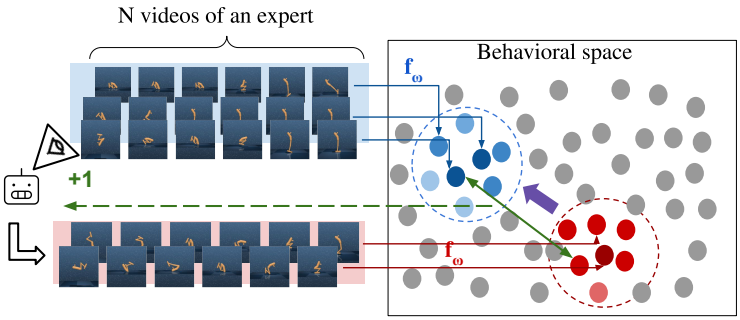


图1：我们方法的示意图。编码器接收智能体轨迹的视觉演示，并将其嵌入通过对比学习训练得到的“行为空间”。这一对比目标促使成功轨迹在潜在空间中彼此接近。我们利用该方法将 N 条专家轨迹编码到行为空间中以蓝色表示的区域。奖励函数对应于智能体轨迹与专家轨迹集合之间的距离。随着智能体不断推进，其当前轨迹会被纳入，作为对比学习的“负”样本，并以红色标示。

traj轨迹生成器，并配有一个判别器函数，用于评估状态有多相似 e
tran所产生的转移与专家的转移相同。

这些方法在低维状态观测的情况下已证明有效。处理高维视觉数据时， [26]使用由连续图像组成的小型堆栈（大小为 3）来评估伪造数据和真实数据。然而，当使用高维视觉观测而非低维状态序列时，每段视频都会记录系统状态的变化，却未明确标示描述环境的主要关键特征。将问题简化为对短序列进行分类，会限制判别器分析并识别这些描述最优动作的特征的能力。能够利用环境图像正确理解和表示系统状态，对于评估状态转移至关重要。此外，这种方法限制了对特定行为的理解，而这些行为需要规划长时域内的动作。另一方面，奖励学习方法 [14, 17, 3] 基于奖励函数的训练，随后将该函数用于奖励智能体，同时采用传统的强化学习算法。这种方法的主要优势在于，能够根据专家示范学习一个衡量智能体动作效率的奖励函数。不同的奖励学习方法会利用精确的自监督学习目标，例如多次执行之间的上下文变化 [17], ，从而使模型收敛到有意义且有效的奖励函数，以训练模仿智能体。

本文提出了 *BootIfOL*，这是一种新颖的奖励学习从观察中模仿算法，它利用原始像素示范，在潜在空间中表示智能体行为，如图1所示。与对抗式方法不同，我们使用长短期记忆网络（ LSTM） [11] 处理完整的观测序列，以跟踪智能体的动作，并表示智能体在每个序列时间步所产生的行为。该方法直接利用来自专家的有限视觉演示来训练模仿智能体，无需获取专家策略或动作。受 Berseth 等人 [1], 的启发，我们还利用智能体生成的新视觉轨迹逐步训练奖励函数。与 [1], 不同，我们的方法会在智能体训练开始前，使用一组失败轨迹训练轨迹编码函数。这有助于从智能体训练一开始就确保奖励函数的一致性。我们还使用对比式多视图编码 [24] 和稠密预测编码 [8]方法对轨迹进行自监督学习。与 [17, 1], 类似，我们通过与环境交互来训练奖励函数，以鼓励模仿智能体的行为与专家行为相似。该方法背后的动机是：(1) 在每个时间步，利用智能体的视觉观测学习并估计系统状态（因为无法获取真实的系统状态）；(2) 识别智能体在一段任务执行期间所产生的行为，并确保该行为与专家的行为服务于相同目的。我们的方法还表明，先对轨迹编码函数进行首轮训练很有价值，这样便能从智能体训练一开始就向其提供有意义的奖励；其他类似的奖励学习方法则省略了这一步骤。我们的方法成功解决了 DeepMind 控制套件 [23] 和 Meta-world 环境 [31], 中的一系列任务，达到接近任务专家的回合奖励。

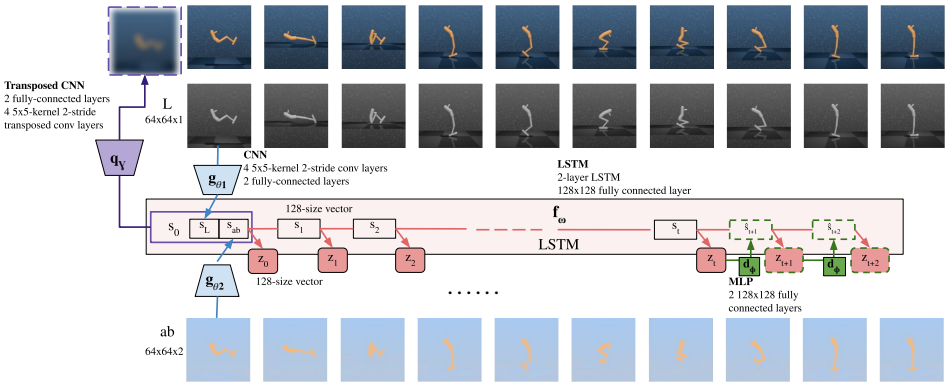


图2：模仿函数的训练架构。对于每个回合，视频会在 Lab 色彩空间中分解，形成 L 视图和 ab 视图。每一帧由 g_θ 编码，并由q_γ解码。得到的状态序列 s_0、……，s_t 使用长短期记忆网络 f_ω 进行编码，以生成序列编码 z_t。随后，z_t 由 d_ϕ 和 f_ω 处理，以预测未来的图像编码。

2 方法

智能体训练 我们旨在仅使用一组专家多次执行同一任务的视频 $\mathcal{D}_e$ 来训练智能体执行该任务。我们的方法基于轨迹匹配目标，即训练智能体，使其轨迹与专家的轨迹相似。换言之，我们希望从专家视频中提取的唯一信息就是其行为。我们的算法分为两个阶段：对齐阶段和交互阶段。在对齐阶段，我们学习一个序列编码函数 f_ω，用于编码智能体轨迹中的帧序列，并学习两段智能体轨迹的序列之间相应的距离度量。我们还会联合训练一个图像编码函数 g_θ。请注意，这个图像编码函数 g_θ 也可以替换为预训练编码器。我们将在第 3 节讨论这种替换的影响。在对齐阶段，我们只使用来自两种分布的轨迹：专家轨迹的分布$O \sim p(\mathcal{D}_e)$，以及由随机定义的策略函数生成的轨迹分布 $O \sim p(\mathcal{D}_a)$。在交互阶段，我们使用标准强化学习（RL）算法 [21]学习智能体策略 π。为了获得奖励，我们使用学习得到的距离（在对齐阶段训练），该距离衡量专家与当前智能体的序列编码之间的差异。关键的是，我们会利用在线环境交互中获得的额外视觉观测，对图像和序列编码函数进行微调。在这项工作中，我们使用 DrQ-v2 [30]强化学习算法训练智能体策略 π及其相应的 Q 值函数 Q，不过我们的方法不受强化学习算法选择的限制。具体而言，在交互阶段，智能体的每个新回合开始时，我们都会采样一个专家回合 $O_e \sim p(\mathcal{D}_e)$，将其作为参考专家策略。在智能体的每一步 t，我们通过 $d(o_{0:t}, o_{e,0:t}) = ||f_\omega(g_\theta(o_{0:t})) - f_\omega(g_\theta(o_{e,0:t}))||$，评估 $o_{0:t}$（即智能体截至步骤 t的轨迹）与 $o_{e,0:t}$（即专家截至 t 的轨迹）之间的距离。最终，在这一步分配给智能体的奖励为 $r_t = -d(o_{0:t}, o_{e,0:t})$。对齐阶段中的函数 f_ω 仅使用专家轨迹和随机轨迹进行训练，因此随着智能体策略的改进，智能体轨迹的分布会发生变化，导致由于轨迹分布偏移而产生的距离不再一致。为了解决这一问题，在交互阶段，我们会继续利用智能体在 π 训练期间生成的新轨迹更新f_ω 和 g_θ。我们将在第 3 段讨论为何必须在训练智能体之前进行对齐阶段。完整的训练过程详见算法 1。

图像编码 我们的目标是对视频（观测轨迹）进行编码，因此需要找到单帧的编码。虽然可以使用预训练图像模型，但并非总能找到合适的模型，或者该模型未必适用（例如，对该任务而言成本过高）。因此，我们提出并评估了将图像编码与序列编码一同学习的方法。图像级目标包含三项：(1) $\mathcal{L}_{triplet}$，用于约束相邻帧的图像编码相似；(2) $\mathcal{L}_{ae}$，使编码能够解码还原为图像；最后，(3) 我们按照 [24]，基于 Lab 色彩空间 [4] 图像表示，学习对比式多视图编码。第一项 $\mathcal{L}_{triplet}$比较锚点图像 o、在时间上接近锚点的正样本图像 o_p，以及在序列中远离锚点图像的负样本图像 o_n 之间的距离：

$$\mathcal{L}_{triplet}(\theta) = ||s - s_p||^2 + \max(\rho - ||s - s_n||^2, 0) \tag{1}$$

算法 1： 从观察中模仿与自举对比学习

$D_e = \{O_e^i\}_{1 \le i \le N} = \{(o_e^{i,0}, o_e^{i,1}, ..., o_e^{i,T})\}_{1 \le i \le N}$
初始化 f_ω, g_θ, q_γ, d_ϕ
// 对齐阶段：训练 f, g
当 $k \le N_{pretrain}$ **执行**
  $\{O_e^i\}_{1 \le i \le n} \leftarrow sample(D_e)$
  $\{O^i\}_{1 \le i \le n} \leftarrow \pi_{random}(env)$
  评估 $\mathcal{L}(\theta, \gamma, \omega, \phi)$，使用 $(\{O_e^i\}, \{O^i\})$
  优化 θ、γ、ω、ϕ
end
// 交互阶段：训练 π、Q、f、g
初始化 $D_a = \{\}$
初始化 $\mathcal{D} = \{\}$, Q, π // 经验回放缓冲区、Q 值函数、策略
当 $step \le N_\pi$ **时执行**
  $O_e \leftarrow sample(D_e)$
  $o_0 \leftarrow env.reset()$
  对于 $t \in 0, .., T-1$ **执行**
    $a_t \leftarrow \pi(o_t)$
    $o_{t+1} \leftarrow env.step(a_t)$
    $r_{t+1} \leftarrow -||f(o_{0:t+1}) - f(o_{e,0:t+1})||$
    将 $(o_t, a_t, o_{t+1}, r_{t+1})$ 保存到 $\mathcal{D}$
    $(o, a, o', r) \leftarrow sample(\mathcal{D})$
    使用 DrQ-v2 更新 Q,π，其中 (o, a, o', r)
  end
  如果 $(step \le N_{train})$ 并且
  $(step$ 模 $N_{update} = 0)$**则**
    将 $O = \{o_{0:t+1}\}$ 保存到 D_a
    $\{O_e^i\}_{1 \le i \le n} \leftarrow sample(D_e)$
    $\{O^i\}_{1 \le i \le n} \leftarrow sample(D_a)$
    评估 $\mathcal{L}(\theta, \gamma, \omega, \phi)$，使用 $(\{O_e^i\}, \{O^i\})$
    优化 θ、γ、ω、ϕ
  end
end

其中 $s = g_\theta(o)$, $s_p = g_\theta(o_p)$。接下来，自编码器的训练目标是最小化损失函数 $\mathcal{L}_{ae}(\theta, \gamma) = ||o - q_\gamma(g_\theta(o))||^2$。最后，我们按照 [24]引入对比式多视图编码，以丰富视觉表征。具体而言，我们考虑颜色空间变换 $\{R, G, B\} \to \{L, ab\}$，其中：(1) 分量 L 是视图 v_1，由卷积神经网络 g_{θ_1}处理；(2) 分量 ab 是视图 v_2，由 g_{θ_2}处理。图像 o的最终编码 s 由 $s = g_\theta(o) = [g_{\theta_1}(v_1)\ g_{\theta_2}(v_2)]$给出。按照公式 [24], ，我们使用记为 $\mathcal{L}_S$的对比学习损失对齐不同视图。该过程如图2所示。总体而言，训练图像编码函数时需要最小化的目标函数为：

$$\mathcal{L}_{frame}(\theta, \gamma) = \mathcal{L}_S(\theta) + \mathcal{L}_{triplet}(\theta) + \mathcal{L}_{ae}(\theta, \gamma) \tag{2}$$

序列编码 为了构建奖励函数，我们学习以具有语义意义的距离来表征帧序列 [7, 8]。我们使用两种损失：（1） $\mathcal{L}_Z$ 一种标准的视频表征学习方法——稠密预测编码（DPC） [8]；（2）以及 $\mathcal{L}_O$ 自举专家行为损失，它鼓励专家轨迹与非专家轨迹之间彼此区分。损失函数 $\mathcal{L}_O$是我们方法的核心，使序列表征能够随着非专家轨迹的改进而逐步提升。DPC 是一种自监督学习方法，其代理任务是将之前的帧序列作为输入来预测下一帧。考虑一个视频数据集 $\mathcal{D} = \{O^i\}_{1 \le i \le N} = \{o^i_{0:T}\}_{1 \le i \le N}$。记作 d_ϕ，它根据状态序列 $s_{0:t}$的编码 z_t 来预测该序列的下一个状态 $\hat{s}_{t+1}$。按照 [8]，我们可以将其用于时空表征学习。我们先预测下一个状态 $\hat{s}_{t+1} = d_\phi(f_\omega(s_1, ..., s_t))$，随后利用该状态预测另外 K-1 个状态 $\hat{s}_{t+k} = d_\phi(f_\omega(s_1, ..., s_t, \hat{s}_{t+1}, \cdots, \hat{s}_{t+k-1}))$。然后，我们应用学习目标来最小化

损失函数 $\mathcal{L}_Z$，其定义如下：

$$\mathcal{L}_Z(\theta,\omega,\phi) = -\mathop{\mathbb{E}}_{o_{0:T}}\left[\frac{1}{K}\sum_{1\le k\le K} L_{t+k}\right] \tag{3}$$

$$L_t = \log\frac{\mathtt{h}(\hat{s}_t, s_t)}{\mathtt{h}(\hat{s}_t, s_t) + \sum_{t'\neq t}\mathtt{h}(\hat{s}_t, s_{t'})} \tag{4}$$

其中 $\mathtt{h}(a,b) = \exp\left(\frac{a^T b}{\tau\|a\|\|b\|}\right)$，且 τ是温度参数。事实上，该损失函数也是一种对比损失函数，其中正样本对为 $(\hat{s}_t, s_t)$，负样本对为 $(\hat{s}_t, s_{t'})_{t\neq t'}$，且二者来自同一示范。为了使来自同一分布的序列彼此更接近，我们对 f_ω返回的序列编码向量 z 使用另一种对比损失。因此，对于属于两个轨迹分布之一的序列 O，我们考虑一个来自相同分布的正序列 O_p，以及一个来自不同分布的负序列 k $\{O_{n,i}\}_{1\le i\le k}$。待最小化的目标函数为 $\mathcal{L}_O$：

$$\mathcal{L}_O(\theta,\omega) = -\mathop{\mathbb{E}}_{(O,O_p,\{O_{n,i}\})}\left[\log\frac{\mathtt{h}(z, z_p)}{\mathtt{h}(z, z_p) + \sum_i \mathtt{h}(z, z_{n,i})}\right] \tag{5}$$

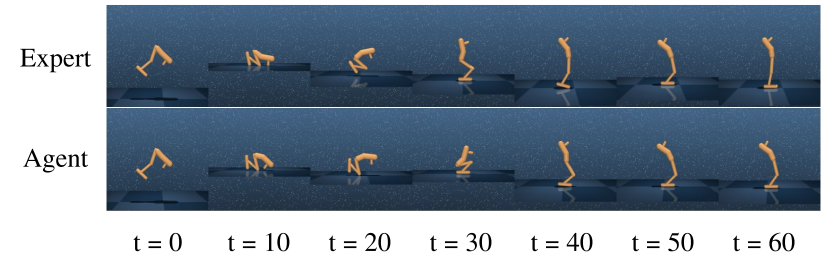
$$z = f_\omega(g_\theta(o_0), \ldots, g_\theta(o_T)) \tag{6}$$

我们要最小化以训练序列编码函数的目标函数为$\mathcal{L}_{seq}(\theta,\omega,\phi) = \mathcal{L}_Z(\theta,\omega,\phi) + \mathcal{L}_O(\theta,\omega)$。最后，对于三元组 $(O, O_p, \{O_{n,i}\}_{1\le i\le k})$，用于评估 $\mathcal{L}_{frame}$ 的帧来自该样本的序列，因此我们可以计算目标损失函数 $\mathcal{L}(\theta,\gamma,\omega,\phi) = \mathcal{L}_{frame}(\theta,\gamma) + \mathcal{L}_{seq}(\theta,\omega,\phi)$。

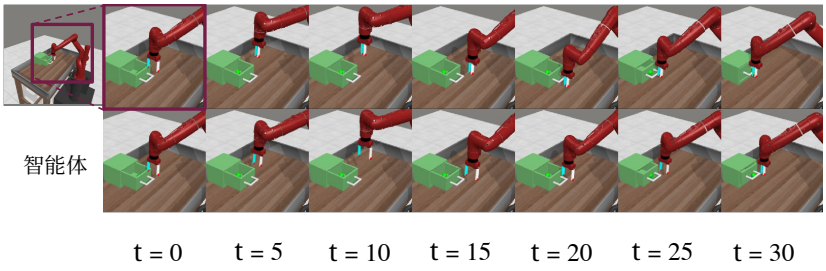
3 实验

我们使用一组多样化的连续控制任务来评估我们的方法，其中包括来自 DM Control Suite [23] 的 4 项任务（Reacher Hard、Finger Turn Easy、Hopper Stand 和 Walker Run），以及来自 Meta-world 环境 [31]的 3 项任务（按按钮、盘子滑动、抽屉关闭）。DM Control 任务要求我们的智能体学习协调多个力矩控制执行器。Meta-world 中的末端执行器在 Sawyer 机械臂上建模，在一个 3DOF 任务空间中进行控制，并配有一个 1DOF 平行夹爪。这种环境设置要求我们的智能体学习如何操纵外部物体，并与环境进行复杂的物理交互。在这些操作任务中，我们还必须对机器人和物体间交互进行建模，例如在按按钮和抽屉关闭任务中，受约束物体关节必须沿单一轴线激活。下面我们将介绍网络架构、训练流程的细节，以及在这两个数据集上的结果。所有回合轨迹均从机器人和交互物体（如适用）的随机初始状态开始。

网络架构我们的图像编码器函数 g_θ 由一对卷积神经网络 g_{θ_1} 和 g_{θ_2}组成。这些模型的架构相似，但 g_{θ_1} 的输入通道数为 1，g_{θ_2}的输入通道数为 2。我们沿用 [17]提出的编码器架构。该架构由四个步幅为 2 的卷积层依次组成，滤波器数量分别为 64、128、256 和 512。这些卷积层后接两个卷积层，滤波器数量分别为 512 和 128。除最后一层外，所有层后均接批归一化 [13] 和带泄漏的ReLU [9] 激活函数（$leak = 2$）。图像解码函数 q_γ 采用与编码器相反的架构，但最后 4 层使用转置卷积层。更多网络细节见附录A中的表 3 和表 4。序列编码器函数 f_ω是一个 2 层长短期记忆网络，输入和输出维度均为 128。输出序列后接一个输入和输出维度均为 128 的全连接层。下一状态预测器由两个全连接层依次组成，两层之间使用带泄漏的ReLU激活函数。我们采用图像编码函数 g_θ，从高维观测中学习潜在表示。该函数通过一种结合不同替代目标的自监督方法进行训练。然而，许多现代研究 [5, 29, 22] 已表明，使用在大型语料库上训练的预训练模型或基础模型对下游任务有益。我们通过采用 EfficientNet [22]，（一种在 ImageNet [19], 上训练的卷积模型）作为骨干网络，展示了在我们的设置中使用这一策略的可能性



(a) Hopper Stand



(b) 抽屉关闭

图3：对比专家（上排）策略与使用我们的方法训练出的模仿智能体（下排）所采取的动作。我们展示了在相同初始条件下训练出的智能体在 Hopper Stand 和抽屉关闭任务中的表现。可以看出，在 Hopper Stand 任务中，我们训练出的智能体的行为与专家非常相似。对于抽屉关闭任务，尽管训练出的智能体采取了与专家不同的轨迹（例如，保持夹爪张开得更大），但它仍然能够完成任务。

图像编码器。在这种设置下，函数 g_θ 由权重冻结的骨干网络模型和一个包含 3 层、规模为 $1280 \times 512 \times 128$ 且参数可训练的多层感知机（MLP）组成。尽管我们采用 EfficientNet-B0 作为骨干网络，但预训练编码器还有许多合理的选择。

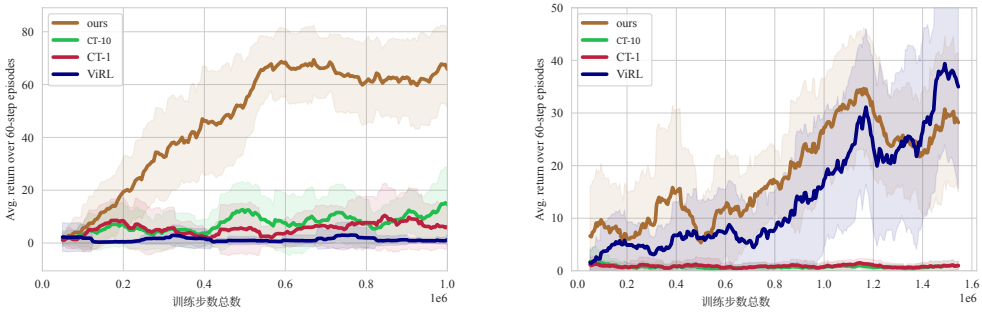
训练：对齐阶段对于每项任务，我们通过运行经过训练的强化学习智能体，并按需渲染环境的视觉图像，为每个时间步生成带彩色图像观察的专家回合轨迹数据集。我们还使用相同的流程，利用随机定义的策略生成第二个轨迹数据集。在本研究中，我们使用 DrQ-v2 智能体作为模仿学习的专家；不过，由于无需获取智能体的状态，因此也可以使用任何合理的专家（基于图像的强化学习、规划智能体、人类或动物专家）。在从头训练图像编码器的情况下（BootIfOL），使用示范数据集 $\mathcal{D}_e$ 和 $\mathcal{D}_a$作为输入，其规模为 $5000 \times 51 \times 64 \times 64$ （轨迹 $\times$ 时间步 $\times$ 高度 $\times$ 宽度）。在利用大规模预训练的 Eff-BootIfOL 中，我们使用较小的示范数据集进行微调：$1500 \times 61 \times 224 \times 224$ （轨迹 $\times$ 时间步 $\times$ 高度 $\times$ 宽度）。

训练：交互阶段 编码函数的参数会在交互阶段的$N_{\text{train}} = 375K$ 训练步数中进行更新。在这些训练步数之后，编码函数的参数将被冻结。我们的强化学习智能体训练持续 $N_\pi = 1550K$训练步数。我们采用 DrQ-v2 [30], 这一基于图像的强化学习智能体，该智能体基于 DDPG [16]架构，并使用默认超参数作为我们的强化学习策略学习器。其他超参数见表7。

总体结果 我们在 DeepMind 控制套件 [23]中的 Reacher Hard、Finger Turn Easy、Hopper Stand 和 Walker Run 任务上进行训练。我们将算法与用于从观察中模仿的上下文转换（CT） [17] 方法进行比较，具体做法是改变每个训练回合中用于预测 CT 智能体轨迹的专家轨迹样本数量 n。我们还将 ViRL [1]和 GAIfO [26]作为基线。在 DM Control 的 4 个任务中进行示范后，如表1所示，我们的方法在通过视觉演示学习完成任务方面表现出色，而 GAIfO 在所有任务上的表现都不佳。为便于比较，我们将表1中的平均回报缩放到 0 到 1 之间。专家的平均回报最高，为 1；随机智能体的平均回报为 0。未缩放的绝对值见附录 B。我们还展示

表1：采用上下文转换（CT）[17] 和 ViRL [1] 算法训练的智能体在 500步回合中的平均回报评估。我们在 Reacher Hard、Finger Turn Easy、Hopper Stand 和 Walker Run 任务上评估这些智能体。在其中 3 个环境中，我们的方法大幅优于现有方法。在 Reacher Hard 上，我们取得了与专家策略相当的奖励，而对比方法则完全未能学到良好的策略。尽管这是一个相当简单的控制问题（逆运动学的视觉版本），但与我们考察的其他任务相比，其初始状态和目标的分布相当广泛。在这种情形下，我们采用失败示范进行训练的技术尤其有优势，因为这样可以看到更多状态空间。表中的回报经过缩放，使专家的平均回报为 1，随机智能体的平均回报为 0。

智能体	平均回报			
	Reacher Hard	Finger Turn Easy	Hopper Stand	Walker Run
BootIfOL（我们的方法）	**0.99**	0.13	**0.74**	**0.06**
ViRL	0.0	0.08	0.58	0.0
CT（n=10）	0.10	**0.18**	0.0	0.0
CT (n=1)	0.04	0.13	0.0	0.03



(a) Reacher Hard　　(b) Hopper Stand

图 4：在交互阶段的智能体训练过程中，Reacher Hard 和 Hopper Stand 任务上的平均回报，与 CT 和 ViRL 智能体进行对比。我们观察到，与其他基线方法相比，智能体在这两种情况下都能快速取得进展。尽管 ViRL 在 Hopper Stand 上表现出色，但它无法应对所有环境（例如 Reacher Hard）。

如表2所示，通过在 Meta-world 环境中采用预训练骨干网络，在复杂视觉场景中取得了令人鼓舞的结果。图4展示了交互阶段强化学习训练期间的平均回合回报。图3展示了我们的智能体在 Hopper Stand 和抽屉关闭任务中执行动作的示例。图9展示了其他任务的示例，并在附录E中进行了说明。我们观察到，在 Walker Run 等任务中，由于控制机器人需要关节之间高度协调，从观察中模仿的智能体即使达到专家奖励的 10% 都很困难（见表1）。

Meta-World 上的评估结果考虑到 Meta-world 环境 [31], ，我们精确地研究了预训练卷积网络作为特征提取模型的重要性。Meta-world 任务所具有的视觉复杂性，使我们能够评估在训练编码函数期间，准确的图像编码器有多重要。按照“网络架构”部分（3）的描述，我们使用预训练图像编码，通过学习得到的距离度量训练了新的智能体，并在 Meta-world 环境中的按按钮、抽屉关闭和盘子滑动任务上对其进行评估。在表2中，我们将该智能体的平均回报与最初的智能体进行比较。表2中的结果表明，图像编码函数对于学习得到的距离度量的有效性至关重要。具体而言，这些结果显示，奖励函数学习的成功与对每一帧的解读准确度有关。我们观察到盘子滑动任务的结果尤其不理想。我们推测，这一失败是由于序列编码函数存在困难，而它承担着不同的作用。尽管特征提取器准确，该任务对我们的智能体而言仍然难以模仿。

表2：在500步回合中评估智能体（*BootIfOL*）的平均回报，其中编码器从头开始训练；并将其与使用 EfficientNet-B0 [22] 作为骨干模型的智能体（ *Eff-BootIfOL* ）进行比较。我们在 Meta-world 模拟器 [31]中对这些智能体进行评估，任务包括按按钮、盘子滑动和抽屉关闭。回报经过缩放，使专家的平均回报为1，随机策略的平均回报为0。我们发现，在这一具有挑战性的视觉环境中，需要使用预训练图像表征才能达到最佳性能。值得注意的是，其他基线方法无法应对这一环境，而我们的方法在部分任务（按按钮和抽屉关闭）中取得了接近专家水平的表现，在其他任务（盘子滑动）中也取得了不俗的成绩。我们在附录 B 的表6中给出未缩放的绝对值。

智能体	平均回报		
	按按钮	盘子滑动	抽屉关闭
Eff-BootIfOL（我们的方法）	**0.70**	0.25	**0.91**
BootIfOL（我们的方法）	0.0	0.25	0.53

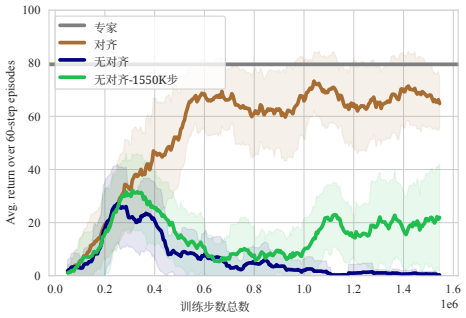
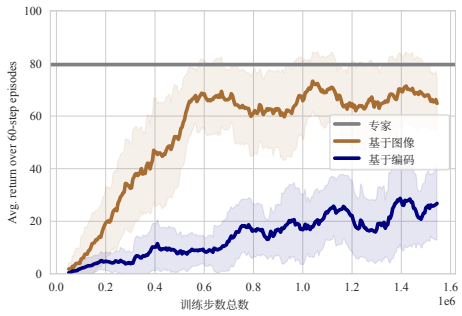


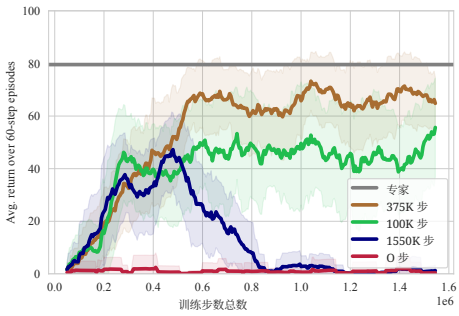
图5：消融研究编码函数在对齐阶段（第2节所述）的训练对智能体在 Reacher Hard 任务上的最终表现的影响。对齐是执行了对齐阶段的初始模型；无对齐和无对齐-*1550K*步是未执行对齐阶段的模型。在无对齐-*1550K*步中，编码函数会持续更新，直至智能体训练结束。

消融对齐阶段在我们的算法中，我们引入了第2节所述的对齐阶段，使我们能够在两种轨迹分布上引导编码函数 f 和 g 的训练：（1）专家轨迹；（2）随机轨迹。这一特点在大多数方法中并不存在，例如类GAN方法。我们假设，从智能体训练一开始就提供显著且稳定的奖励，能够显著提升对最优模仿策略的搜索效果。我们通过两种方式移除对齐阶段，以评估该步骤的重要性：（1）在交互阶段中，编码函数经过总计375K个训练步的训练（标准情况）；（2）在交互阶段中，编码函数贯穿智能体的整个训练过程进行训练（类似于 ViRL 和 GAIfO）。我们在图5中展示了该实验的结果。我们观察到，采用对齐阶段能够显著提升性能，凸显了这一阶段的关键作用。尽管在整个交互阶段训练这些函数能带来一定进展，但其效率远不及包含对齐阶段的引导式方法。

消融研究：在强化学习中使用学习得到的图像编码器迄今为止，在我们的实验中，强化学习智能体通过彼此独立的策略函数 π 和 Q 值函数 Q进行学习，如 [30]中所述。在这种架构中，策略函数 π 由一个卷积神经网络 E 组成，该网络在内部对观测图像进行编码，然后由一个多层感知机将图像编码作为输入并输出动作。 E 由策略 π 及其关联的 Q 值函数 Q共享，后者将编码-动作对作为输入并输出 Q 值。一个自然的问题是，我们能否通过去掉卷积神经网络 E，将学习得到的图像编码函数 g_θ直接连接到 π 和 Q 。这种方法的意义在于，将 g的学习迁移给智能体，使其能够直接学习提取特征。我们知道，在现代强化学习方法中，使用低维状态向量通常能提升智能体的性能和效率，因为它们对状态的描述更精确。我们希望了解：通过基于像素的重建损失优化得到的观测特征描述，是否包含足够的状态信息，能够让强化学习策略在无法访问真实模拟器状态或原始像素的情况下，学会高效且优质的控制策略。具体而言，我们用一个共享的多层感知机替换 E



(a) 基于编码的智能体评估



(b) 编码器训练时长评估

图 6：消融研究。我们展示了智能体在 Reacher Hard 任务上进行 5 个回合、每回合 60 步时的平均回报。左侧的 6a 评估我们能否将模仿函数中的图像编码卷积神经网络重新用于策略学习（基于编码），还是强化学习智能体应优化一个新的图像编码网络（基于图像）。我们发现，尝试将 BootIfOL 中的编码直接用于策略网络（基于编码）会降低性能。右侧的 6b 评估了编码函数的训练时长对智能体的影响。如果在对齐阶段之后不再训练编码函数（*0* 步），奖励将无法提供有效信息。同样，如果在智能体开始收敛到强策略时仍继续训练编码函数，奖励信号也可能退化。

连接 π 和 Q。 g 输出的图像编码是多层感知机 E的输入数据。请注意，在这项实验中，我们冻结 g_θ 的参数，使其不受强化学习更新的影响，并将这一消融实验称为基于编码的智能体。我们在图 6a 中展示了原始基于图像的智能体和基于编码的智能体在 Reacher Hard 任务训练过程中的平均回报。这使我们认为， g 输出的图像编码存在偏差，限制了策略对环境的理解，因此阻碍该策略达到与能够直接访问基于图像的状态的智能体相近的性能。我们推测，这种偏差是因为 g 和 f 是通过联合训练得到的，而其学习目标并未针对策略函数进行优化。

交互阶段中编码器训练时长的影响 在交互阶段，我们先训练轨迹和图像编码 N_{train} 步，然后切换为仅使用标准强化学习算法。现在，我们研究这一阶段的时长如何影响智能体最终获得的奖励。我们评估了算法在 N_{train} 设为总训练步数 0、100K、375K 和 1550K 时的表现。编码器参数每隔 $N_{update} = 50$ 步更新一次。图 6b 展示了这些实验在 Reacher Hard 任务上的结果。我们发现，若在交互阶段不更新编码函数，学习便无法继续，因为对齐阶段提供的奖励不足。最初，增加 N_{train} 步数对每回合平均回报有积极影响；当步数为 $N_{train} = 375$K 时，平均回报接近专家每回合的回报。然而，如果在交互阶段训练时不冻结编码函数，已学习的奖励函数会因状态编码不断变化而变得脆弱。因此，我们发现， $N_{train} = 1550$K 步最终导致策略崩溃。

4 结论

我们提出了一种新的从观察中模仿方法，它依靠利用视觉观测中的示范样例对智能体行为进行编码。此方法使用状态和序列的自监督学习，逐步训练一个供强化学习智能体使用的奖励函数。我们在一组包含 7 项模拟机器人任务上展示了该方法的优势，这些任务仅能获取有限的专家示范。我们指出，尽管我们采用的问题框架——从图像观察中进行模仿——具有重要的潜在实际应用价值，但对于需要精确控制的任务而言，从计算和观察角度来看都颇具挑战性。本文中，我们通过让智能体使用任务专用图像编码器和预训练图像编码器提供的信息进行训练，来探索这一挑战，并表明编码函数的保真度对下游控制任务至关重要。我们提升学习到的表征性能的方法，是保留智能体在交互阶段生成的数据。在未来的工作中，我们计划扩展序列优化函数，使其评估的不仅是目前已有的两种智能体轨迹

分布（专家和非专家），还包括智能体训练过程中产生的多种性能水平的分布。

参考文献

[1] Glen Berseth、Florian Golemo 和 Christopher Pal。从单个视频演示中学习模仿。arXiv 预印本 arXiv:1901.07186，2019 年。[2] Mariusz Bojarski 、 Davide Del Testa 、 Daniel Dworakowski 、 Bernhard Firner、Beat Flepp、Prasoon Goyal、Lawrence D Jackel、Mathew Monfort、Urs Muller、Jiakai Zhang 等。端到端自动驾驶汽车学习。arXiv 预印本 arXiv:1604.07316，2016年。[3] Wei-Di Chang、Juan Camilo Gamboa Higuera、Scott Fujimoto、David Meger 和 Gregory Dudek。IL-Flow：使用归一化流进行基于观察的模仿学习。arXiv 预印本 arXiv:2205.09251，2022年。[4] Christine Connolly 和 Thomas Fleiss。RGB 到 CIELAB 色彩空间转换的效率与准确性研究。IEEE 图像处理汇刊，6(7):1046–1048，1997年。[5] Carl Doersch、Abhinav Gupta 和 Alexei A Efros。通过上下文预测进行无监督视觉表征学习。发表于 IEEE 国际计算机视觉会议论文集，第 1422–1430 页，2015年。[6] Ian Goodfellow、Jean Pouget-Abadie、Mehdi Mirza、Bing Xu、David Warde-Farley、Sherjil Ozair、Aaron Courville 和 Yoshua Bengio。生成对抗网络。ACM 通讯，63(11):139–144，2020年。[7] Daniel Gordon、Kiana Ehsani、Dieter Fox 和 Ali Farhadi。看着世界流逝：从无标签视频中进行表征学习。arXiv 预印本 arXiv:2003.07990，2020年。[8] Tengda Han、Weidi Xie 和 Andrew Zisserman。通过稠密预测编码进行视频表征学习。发表于 IEEE/CVF 国际计算机视觉会议研讨会论文集，第 0–0 页，2019年。[9] Kaiming He、Xiangyu Zhang、Shaoqing Ren 和 Jian Sun。深入研究整流器：在 ImageNet 分类上超越人类水平的表现。发表于 IEEE 国际计算机视觉会议论文集，第 1026–1034 页，2015年。[10] Jonathan Ho 和 Stefano Ermon。生成对抗模仿学习。神经信息处理系统进展，29，2016年。[11] Sepp Hochreiter 和 Jürgen Schmidhuber。长短期记忆网络。神经计算，9 1780，1997 8 1735( ): – 。[12] Ahmed Hussein、Mohamed Medhat Gaber、Eyad Elyan 和 Chrisina Jayne。模仿学习：学习方法综述。ACM 计算调查（CSUR），50(2):1–35，2017年。[13] Sergey Ioffe 和 Christian Szegedy。批量归一化：通过减少内部协变量偏移加速深度网络训练。发表于国际机器学习会议，第 448–456 页。PMLR，2015年。[14] Daiki Kimura、Subhajit Chaudhury、Ryuki Tachibana 和 Sakyasingha Dasgupta。基于观察的内部模型用于奖励塑形。arXiv 预印本 arXiv:1806.01267，2018年。[15] Diederik P Kingma 和 Jimmy Ba。Adam：一种随机优化方法。arXiv 预印本 arXiv:1412.6980，2014年。[16] Timothy P Lillicrap、Jonathan J Hunt、Alexander Pritzel、Nicolas Heess、Tom Erez、Yuval Tassa、David Silver 和 Daan Wierstra。深度强化学习中的连续控制。arXiv 预印本 arXiv:1509.02971，2015年。[17] YuXuan Liu、Abhishek Gupta、Pieter Abbeel 和 Sergey Levine。从观察中模仿：通过上下文转换，从原始视频中学习模仿行为。发表于 2018 IEEE 国际机器人与自动化会议（ICRA），第 1118–1125 页。IEEE，2018年。[18] Stéphane Ross、Geoffrey Gordon 和 Drew Bagnell。将模仿学习和结构化预测简化为无悔在线学习。发表于第十四届国际人工智能与统计会议论文集，第 627–635 页。JMLR 研讨会与会议论文集，2011年。[19] Olga Russakovsky、Jia Deng、Hao Su、Jonathan Krause、Sanjeev Satheesh、Sean Ma、Zhiheng Huang、Andrej Karpathy、Aditya Khosla、Michael Bernstein 等。ImageNet 大规模视觉识别挑战赛。国际计算机视觉杂志，115:211–252，2015年。[20] Stefan Schaal。示范学习。神经信息处理系统进展，9，1996年。[21] Richard S Sutton 和 Andrew G Barto。强化学习：导论。麻省理工学院出版社，2018年。[22] Mingxing Tan 和 Quoc Le。EfficientNet：重新思考卷积神经网络的模型缩放。发表于国际机器学习会议，第 6105–6114 页。PMLR，2019年。[23] Yuval Tassa、Yotam Doron、Alistair Muldal、Tom Erez、Yazhe Li、Diego de Las Casas、David Budden、Abbas Abdolmaleki、Josh Merel、Andrew Lefrancq 等。DeepMind 控制套件。arXiv 预印本 arXiv:1801.00690，2018年。[24] Yonglong Tian、Dilip Krishnan 和 Phillip Isola。对比式多视图编码。发表于欧洲计算机视觉会议，第 776–794 页。Springer，2020年。[25] Faraz Torabi、Garrett Warnell 和 Peter Stone。基于观察的行为克隆。arXiv 预印本 arXiv:1805.01954，2018年。[26] Faraz Torabi、Garrett Warnell 和 Peter Stone。基于观察的生成对抗模仿。arXiv 预印本 arXiv:1807.06158，2018年。[27] Faraz Torabi、Garrett Warnell 和 Peter Stone。利用本体感觉进行视频模仿学习。

[28] Faraz Torabi、Garrett Warnell 和 Peter Stone。基于观察的模仿学习近期进展。*arXiv* 预印本 *arXiv:1905.13566*，2019年。[29] Xiaolong Wang 和 Abhinav Gupta。使用视频进行视觉表征的无监督学习。载于*IEEE* 国际计算机视觉会议论文集，第 2794–2802 页，2015年。[30] Denis Yarats、Rob Fergus、Alessandro Lazaric 和 Lerrel Pinto。掌握视觉连续控制：改进的数据增强强化学习。*arXiv* 预印本 *arXiv:2107.09645*，2021年。[31] Tianhe Yu、Deirdre Quillen、Zhanpeng He、Ryan Julian、Karol Hausman、Chelsea Finn 和 Sergey Levine。Meta-world：用于多任务和元强化学习的基准与评估。载于机器人学习会议，第 1094–1100 页。PMLR，2020年。

A 神经网络架构

表 3：卷积神经网络 g_{θ_1} 和 g_{theta_2}的架构。 C 是每一层输出矩阵的通道数。 K 是每一层卷积核的大小。 S 是卷积操作的步幅。 P 是最初添加到输入矩阵中的填充。

编码器网络				
层	C	K	S	P
图像（64×64）	1,2	-	-	-
卷积 + 批归一化 + Leaky ReLU	64	5	2	0
卷积 + 批归一化 + Leaky ReLU	128	5	2	0
卷积 + 批归一化 + Leaky ReLU	256	5	2	0
卷积 + 批归一化 + Leaky ReLU	512	5	2	0
卷积 + 批归一化 + Leaky ReLU	512	1	1	0
Conv	128	1	1	0

表 4：基于转置卷积操作的图像解码函数架构q_γ 每层输出矩阵的通道数。C 每层卷积核的大小。 K 卷积操作的步幅。 S 添加到输出矩阵的填充。添加到输入矩阵的填充始终为 0。

解码器网络				
层	C	K	S	OP
图像编码 (1×1)	128	-	-	
卷积 +Leaky ReLU	512	1	1	0
卷积 + 批归一化 + Leaky ReLU	512	1	1	0
转置卷积 + 批归一化 +Leaky ReLU	256	5	2	0
转置卷积 + 批归一化 +Leaky ReLU	128	5	2	0
转置卷积 + 批归一化 +Leaky ReLU	64	5	2	1
转置卷积	3	5	2	1

B 我们的模型和基线方法的估计（绝对平均回合回报）

我们在表5和表6中详细列出了，在实验的7个任务中，我们的模型和基线方法分别取得的未经缩放的平均回合回报。

C 编码函数损失

在对齐阶段，我们在专家轨迹和随机轨迹上评估中间损失项 $\mathcal{L}_Z$、 $\mathcal{L}_{seq}$、 $\mathcal{L}_{ae}$ 和 $\mathcal{L}_{triplet}$。这些不同的函数用于引导

表 5：采用上下文转换（CT） [17] 和 ViRL [1] 算法训练的智能体在 500步回合中的平均回报评估。我们在 Reacher Hard、Finger Turn Easy、Hopper Stand 和 Walker Run 任务上评估这些智能体。

智能体	平均回报			
	Reacher Hard	Finger Turn Easy	Hopper Stand	Walker Run
专家	850.52 ± 315.70	893.44 ± 219.75	880.93 ± 76.89	789.77 ± 17.25
BootIfOL（我们的方法）	$\mathbf{843.16 \pm 274.84}$	199.76 ± 399.02	$\mathbf{651.33 \pm 362.95}$	$\mathbf{79.50 \pm 1.25}$
ViRL	0.44 ± 1.27	160.2 ± 366.52	516.27 ± 417.21	29.31 ± 24.72
CT（n=10）	97.84 ± 254.52	$\mathbf{238.92 \pm 422.07}$	1.42 ± 3.22	25.15 ± 18.71
CT（n=1）	38.04 ± 127.53	199.48 ± 396.99	1.01 ± 1.71	52.42 ± 34.28
随机	6.64 ± 16.94	92.6 ± 202.68	2.44 ± 5.19	29.93 ± 4.52

表 6：对我们的智能体（*BootIfOL*）在编码器从头开始训练的情况下，以及将 EfficientNet-B0 [22] 用作骨干模型的智能体（*Eff-BootIfOL*）进行 500 步回合的平均回报评估。我们在 Meta-world模拟器 [31] 中的操作任务上评估这些智能体：按按钮、盘子滑动和抽屉关闭。

智能体	平均回报		
	按按钮	盘子滑动	抽屉关闭
专家	556.06 ± 6.46	734.56 ± 172.51	4223.65 ± 21.14
Eff-BootIfOL（我们的方法）	$\mathbf{434.39 \pm 139.74}$	340.45 ± 46.56	$\mathbf{3949.90 \pm 681.42}$
BootIfOL（我们的方法）	146.56 ± 17.71	$\mathbf{342.40 \pm 93.08}$	2847.10 ± 1679.56
随机	153.29 ± 51.68	209.12 ± 87.38	409.07 ± 1288.89

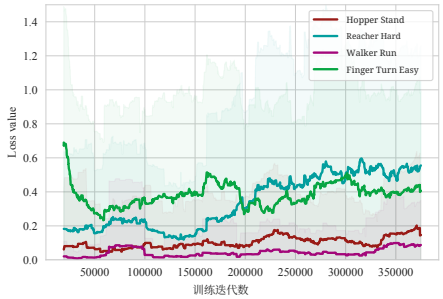
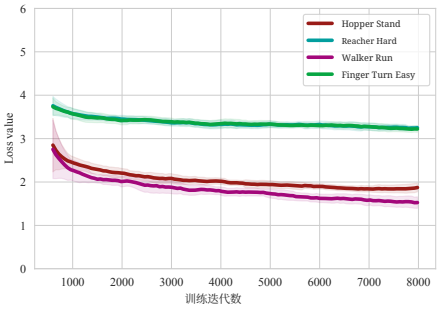


图7：在交互阶段，损失项 $\mathcal{L}_{seq}$ 的变化情况。在此阶段，智能体按照第2节所述进行训练。该图展示了强化学习智能体在环境中学习的效果。

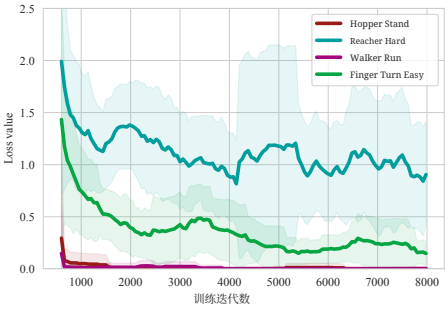
用于下一阶段的编码函数，各自都有明确的学习目标，如第2节所述。我们在图8中详细展示了对齐阶段这些损失项的变化情况。在交互阶段，训练模仿智能体时，编码函数会逐步利用智能体的新轨迹和初始专家轨迹进行训练。我们在图7中展示了 $\mathcal{L}_{seq}$ 损失函数的变化情况。可以看到， $\mathcal{L}_{seq}$ 有轻微上升的趋势。这是因为序列编码函数越来越无法区分专家序列和智能体序列。

D 超参数

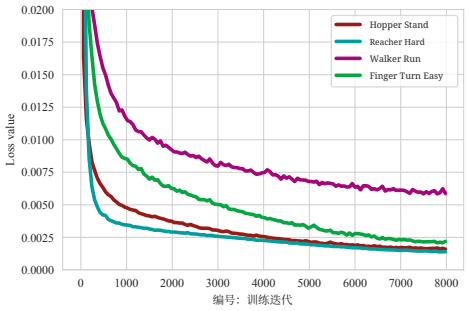
我们方法的超参数及其默认值列于表7中。



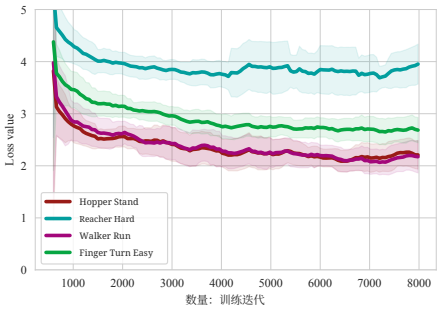
(a) 对 $\mathcal{L}_Z$的评估



(b) 对 $\mathcal{L}_{se}$的评估 $_q$



(c) 对 $\mathcal{L}_{ae}$的评估



(d) 对 $\mathcal{L}_{triplet}$的评估

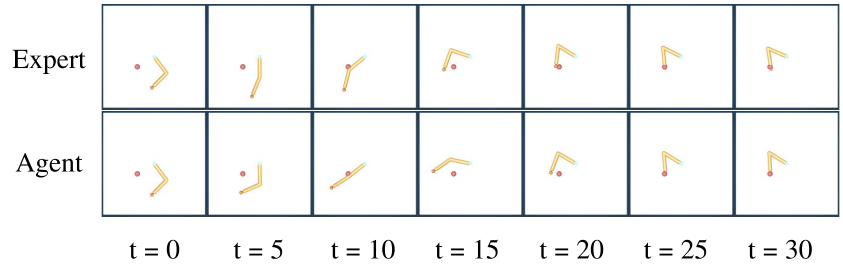
图8：对齐阶段训练编码函数 g_θ 和 f_ω 期间损失项 $\mathcal{L}_Z$、 $\mathcal{L}_{seq}$、 $\mathcal{L}_{ae}$ 和 $\mathcal{L}_{triplet}$ 的演变。

表7：超参数

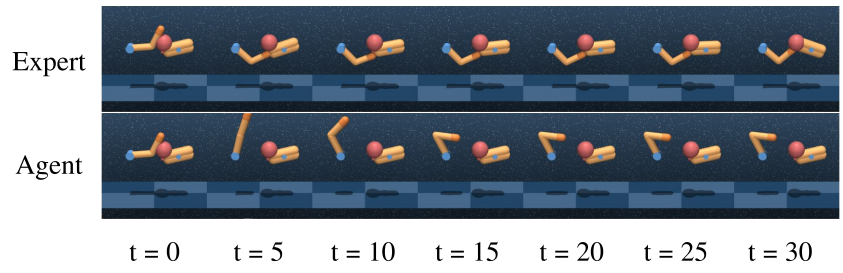
参数	值
专家轨迹数量（BootIfOL）（N）	5000
专家轨迹数量（Eff-BootIfOL）（N）	1500
每条轨迹中的帧数（BootIfOL）（T）	61
每条轨迹中的帧数（Eff-BootIfOL）（T）	51
图像尺寸（BootIfOL）	64×64
图像尺寸（Eff-BootIfOL）	224×224
对齐阶段的训练轮次数（$N_{pretrain}$）	8000
每个批次的视频数（n）	16×2（对）
编码器训练期间的训练步数总数（N_{train}）	375K
编码器参数更新周期（以训练步计）（N_{update}）	50
智能体训练步数总量（N_π）	1.55 M
学习率	1×10^{-4}
优化器	Adam [15]

E 示范

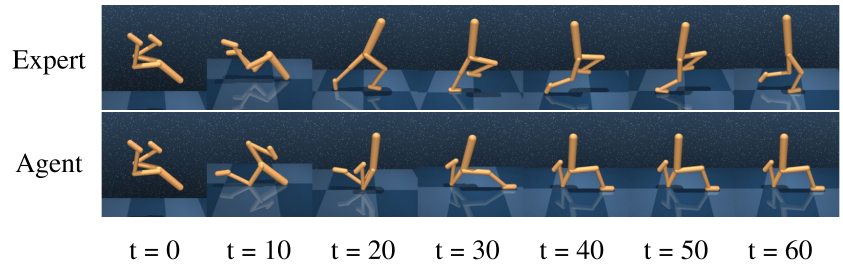
We p图9展示了专家执行各项任务的示例，以及年龄
n在相同的初始条件下。



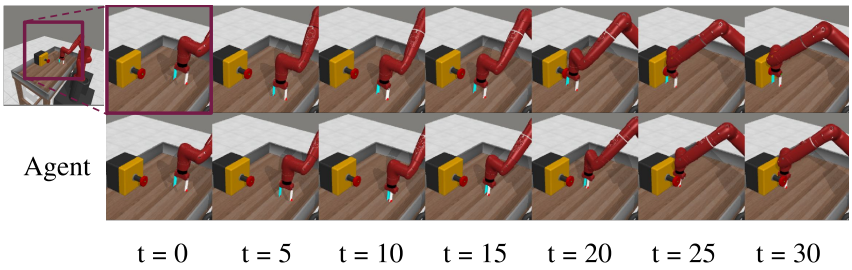
(a) Reacher Hard



(b) 手指转动



(c) Walker Run



(d) 按按钮

图9：在每个任务和相同初始条件下，专家与智能体的动作对比。