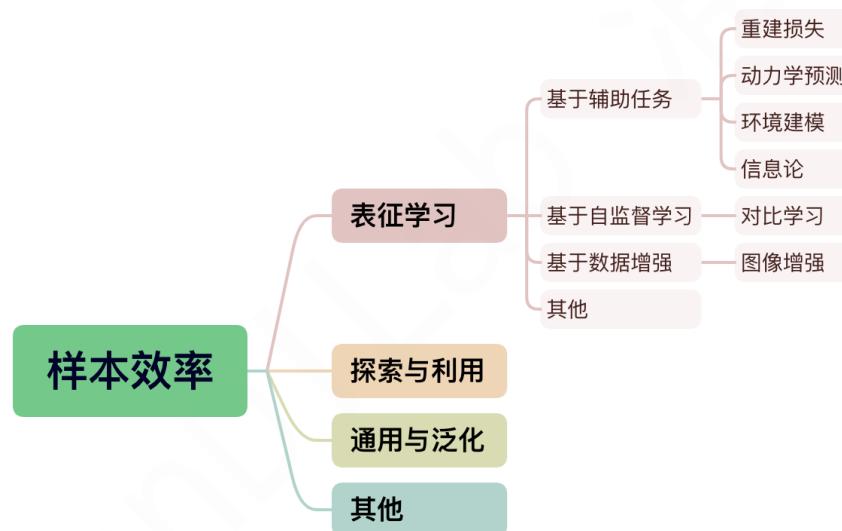


表征学习+强化学习联合训练

概述

强化学习 (Reinforcement Learning, RL) 智能体通过与环境交互不断改进自己的策略。结合深度学习的强大表征能力，RL 智能体可以在复杂任务中表现出巨大的潜力。然而目前的 RL 方法一般需要大量与环境交互在线产生数据，尤其是基于图像输入的强化学习算法，其样本效率尤其低下。如图1所示，为了提高样本效率，RL 算法一般可以从表征学习、探索与利用、通用与泛化等3个角度进行设计优化。

- 其中第一个角度【表征学习】中的主要技术包括：
 - 基于辅助任务 (auxiliary task) 的技术。包括利用重建损失，动力学预测，环境建模 (world model)，信息论等技术来获取状态的高效表征。代表论文包括基于环境建模的 PlaNet [13] 和 Dreamer [14]，基于信息论的 Skew-Fit [12] 等。
 - 基于自监督学习 (Self-Supervised Learning, SSL) 的技术。代表论文包括基于对比学习技术 (SSL 技术的一种) 的 CURL [1] 等。
 - 基于数据增强的技术。代表论文包括基于图像增强技术的 DrQ [2] 等。



(图1：RL 中提高样本效率的思考角度概览图。)

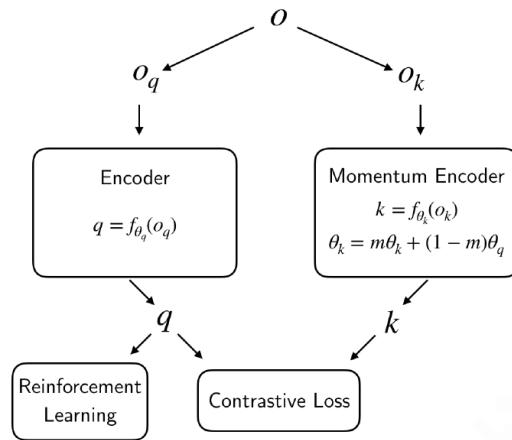
- 本文旨在概述性地讲解和探讨**表征学习+强化学习联合训练的方法**，全文结构组织如下：
 - 首先介绍基于对比学习的 CURL。
 - 接着介绍基于图像增强的 DrQ。
 - 最后介绍2022年的一篇分析总结工作 “Does Self-supervised Learning Really Improve Reinforcement Learning from Pixels?”，系统地探究了图像增强技术与各种自监督学习技术在各种环境上的实际效果，对学习到的表征进行了简单分析，并简单比较了 SSL+RL 联合训练框架与 (先SSL再RL的) 预训练框架的异同。

注解：RL 中算法 (智能体) 的样本效率 (sample efficiency) 一般通过以下指标来衡量：在相同的与环境交互的步数(environment steps) 与相同的网络更新步数 (gradient steps) 下，智能体在该环境上的性能指标。该性能指标越好，说明该智能体样本效率越高。

CURL: Contrastive Unsupervised Representations for Reinforcement Learning

简介

- 对于图像输入(称为 pixel-based 或 image-based)的强化学习方法,由于一般需要先从图像中提取高阶语义特征用于决策,其样本效率(sample efficiency)会比 state-based (state 一般为紧凑的低维向量)的方法低很多。
- 受最近研究对比学习的工作 MoCo [8] 的启发,CURL 通过增加对比表征学习的辅助任务,使得其样本效率能够和 state-based 的方法相比较。在 DeepMind Control Suite (DMC) 和 Atari 游戏上,CURL 优于现有的基于模型和无模型的图像输入的方法,在环境交互步骤为 100K 的基准测试下,性能分别提高 1.9 倍和 1.2 倍。在 DMC 任务中,CURL 是基于图像的方法中第一个与基于状态的方法采样效率相当的算法。CURL 原理如图2所示。



(图2: 用于强化学习的对比无监督表征方法(CURL):结合了实例对比学习(instance contrastive learning)和强化学习。CURL 通过优化对比损失来训练一个视觉表征编码器: 主要思想是确保原始数据 o 的不同增强版本(或者说不同视角) o_q 和 o_k 相匹配。query observations o_q 被视为锚点(anchor), 而 key observations o_k 包含正负样本, 是与 RL 更新使用的同一个(采样的) minibatch 数据。key k 是 key observations o_k 使用 query encoder 的动量平均(momentum averaged)版本进行编码得到的。RL 策略和(或)值函数也是建立在 query encoder 之上的, 因此 query encoder 是通过对比学习和强化学习目标一起进行训练。CURL 是一个通用框架, 可以用于任何需要从高维图像中学习的 RL 算法。)

- 在 RL 与 CV 中采用对比学习主要有两个区别:
 - RL 没有像 CV 中大量的有标签的图像数据集, 在 RL 中数据集是通过智能体与环境的交互在线收集的, 其分布会随智能体的策略演进而动态变化;
 - 智能体一般需要同时进行无监督学习和强化学习, 而不是像 CV 一样先进行无监督学习, 再针对特定的下游任务微调预训练好的网络。

CURL 实现

下面首先介绍 CURL 用到的对比学习损失函数 InfoNCE loss (Information Noise Contrastive Estimation) [15]。

InfoNCE loss

对比学习可以理解为学习一个可微分字典查询任务(learning a differentiable dictionary look-up task)。给定一个**查询(query)** q 和很多个**键值(keys)** $\mathbb{K} = \{k_0, k_1, \dots, k_{K-1}\}$, 对应于 q 的显式的已知分区 $P(\mathbb{K}) = (\{k_+\}, \mathbb{K} \setminus k_+)$, 即包括1个正样本 k_+ , 其他为负样本。对比学习的目标是确保 q 与 k_+ 的匹配程度比与集合 $\mathbb{K} \setminus k_+$ 中的任意一个 key 的匹配程度都强。在对比学习的文献中, $q, \mathbb{K}, k_+, \mathbb{K} \setminus k_+$ 分别被称为锚点, 目标点, 正样本, 负样本(anchor, targets, positive, negatives)。

CURL 中用于对比学习的损失函数为 InfoNCE loss, 数学表达式如下:

$$\mathcal{L}_q = \log \frac{\exp(q^T W k_+)}{\exp(q^T W k_+) + \sum_{i=0}^{K-1} \exp(q^T W k_i)}$$

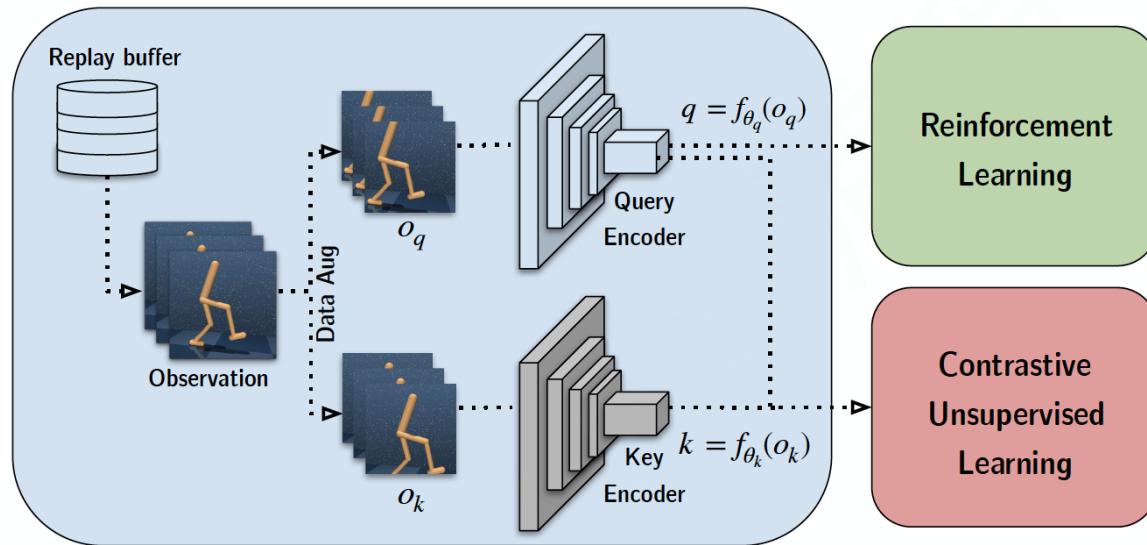
其中 W 是一个参数可学习的矩阵。该损失函数可以看做一个 K-way softmax 分类器的 log loss 版本，即把 query 与 targets (包括正负样本) 的匹配问题看做一个分类问题，即把 query 的真实类别 (即 label) 看做是 k_+ 。

注解：

- 对比损失指满足下面性质的一个函数：当 query 与它的 positive key 越相似，而与所有其他的 key (negative keys) 越不相似的时候，对比损失的值越小。
- anchor 和 targets 之间的相似度一般有以下3种建模方式：
 - 点积 (dot products) ($q^T k$)
 - 双线性积 (bilinear products) ($q^T Wk$)
 - 欧几里得距离 (euclidean distances)
 - CURL 采用双线性积的形式。

CURL 体系结构

CURL 具体的结构如图3所示。



(图3: CURL 体系结构：首先从回放缓冲区 (replay buffer) 中采样一个 batch 的样本 (a batch of transitions)，然后对原始图像观察数据进行两次增强 (data-augmented，即2次变换)，得到 query observations o_q 和 key observations o_k ，然后分别使用 query encoder 和 key encoder 对其进行编码得到 query q 和 key k 。queries 被传递到 RL 算法中，而 query-key pairs 被传递到对比学习的目标函数中。与 MoCo [8] 类似，在梯度更新时仅更新 query encoder，key encoder 的权重是 query encoder 权重的移动平均值。)

CURL 实现细节

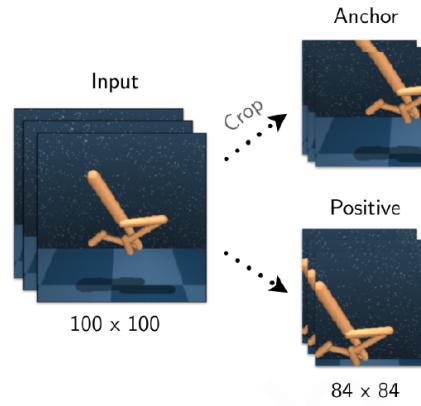
Discrimination Objective 的选择

- 对比表征学习的一个关键组成部分是相对于 anchors 如何选择正负样本，当前对比学习文献中主要包括2种方式：patch discrimination 和 instance discrimination。
 - patch discrimination：指的是把图片的一个 patch 作为一个样本，把相对 anchors 具有精心设计的空间偏移的 patch 作为正样本 (positives)，把本图像的其他 patch 和其他图片的 patch 作为负样本 (negatives)。虽然 patch 是将空间 discrimination 和实例 discrimination 结合在一起的有效方式，但它们引入了额外的超参数和架构设计选择，这可能很难适配到新问题上。
 - instance discrimination：SimCLR [16] 和 MoCo 等工作不使用 patch，直接将每张图片的不同增强版本作为 anchor 和正样本，其他所有图片作为它的负样本。可以有更少的超参数。
 - 出于以下两个原因，最好在 RL 中选择更简单的 Discrimination Objective。
 - 首先，考虑到强化学习算法的脆弱性，复杂的 Discrimination Objective 可能会破坏RL目标的稳定性。

- 其次，由于 RL 算法是在动态生成的数据集上训练的，因此复杂的 Discrimination Objective 可能会显著增加训练时间。
 - 因此在 CURL 中作者选择了类似 MoCo 的 instance discrimination 的方式。

Query-Key Pair Generation

- 与 MoCo 类似，anchor 和 positives 是同一图像的两种不同的增强版本，而 negatives 来自其他图像。CURL 主要依赖于随机裁剪的数据增强方法，即从原始图片中裁剪出随机正方形 patch。
- RL 和 CV 之间的一个显著差异是，pixel-based 的无模型 RL 算法输入的 obs 不仅仅是单个图像，而是连续时间上的多个图像帧的堆叠。例如，在 Atari 中一般堆叠4帧，DMC 中一般堆叠3帧。通过这种方式，在多个图像帧的堆叠上执行 instance discrimination 允许 CURL 学习空间和时间上的 discriminative features。有关 CURL 捕获时域特征的详细信息，请参见原论文附录E。
- 作者在一个 batch 的不同 obs 上应用随机增强即随机裁剪，但在一个 obs（如上所述，一般包含多个图像帧的堆叠）的每个图片中应用相同的随机裁剪坐标，以保留 obs 中关于时间结构的信息。数据增强过程如图4所示。详细信息请参阅原论文附录A。



(图4：直观地展示了使用 stochastic random crops 生成锚点及其正样本的过程。裁剪的纵横比 (aspect ratio) 为 0.84，即从原始渲染的 100x100 的图像裁剪得到 84x84 的图像。对于叠好的4帧应用相同的随机裁剪坐标可以确保时间一致的空间抖动 (time-consistent spatial jittering)。)

Target Encoding with Momentum

- 类似 MoCo，如果将对比学习看做学习一个可微分的字典查询任务 (query key 匹配问题)，期望字典拥有如下性质：(1) 字典包含的 key 尽可能多，(2) 在训练过程中字典中的 key 需要保持 consistent。
- 直观地说，更大的字典可以更好地采样底层连续的高维视觉空间，而字典中的 key 应该由相同或相似的编码器得到，以便它们与 query 的比较是一致的。
- key encoder 采用 query encoder 的移动平均形式，有助于 key 在训练过程中保持一致 (consistent)。

$$\theta_k = m\theta_k + (1 - m)\theta_q$$

相似度的度量 (Similarity Measure)

Discrimination Objective 中的另一个决定因素是用于衡量 query-key pairs 的 agreement 程度的方式。CURL 采用双线性内积 (bilinear products) $\text{sim}(q, k) = q^T W k$ ，其中 W 是可学习的参数矩阵。作者发现这种相似性度量优于最近计算机视觉中最先进的对比学习方法如 MoCo 和 SimCLR 中使用的归一化点积 $q^T k$ (参见附录 A 中的图 5)。

伪代码

```

# f_q, f_k: encoder networks for anchor
# (query) and target (keys) respectively.
# loader: minibatch sampler from ReplayBuffer
# B-batch_size, C-channels, H,W-spatial_dims
# x : shape : [B, C, H, W]
# C = c * num_frames; c=3 (R/G/B) or 1 (gray)
# m: momentum, e.g. 0.95
# z_dim: latent dimension
f_k.params = f_q.params
W = rand(z_dim, z_dim) # bilinear product.
for x in loader: # load minibatch from buffer
    x_q = aug(x) # random augmentation
    x_k = aug(x) # different random augmentation
    z_q = f_q.forward(x_q)
    z_k = f_k.forward(x_k)
    z_k = z_k.detach() # stop gradient
    proj_k = matmul(W, z_k.T) # bilinear product
    logits = matmul(z_q, proj_k) # B x B
    # subtract max from logits for stability
    logits = logits - max(logits, axis=1)
    labels = arange(logits.shape[0])
    loss = CrossEntropyLoss(logits, labels)
    loss.backward()
    update(f_q.params) # Adam
    update(W) # Adam
    f_k.params = m*f_k.params+(1-m)*f_q.params

```

实验

在 DeepMind Control Suite 和 Atari 游戏的复杂任务上，CURL 优于（包括基于模型和无模型的）现有的图像输入的 RL 方法。具体地，在交互步骤为 100K 的基准测试下，这 2 类任务的性能分别提高 1.9 倍和 1.2 倍。在 DMC 任务中，CURL 是基于图像的方法中第一个与使用基于状态的方法采样效率相当的算法。

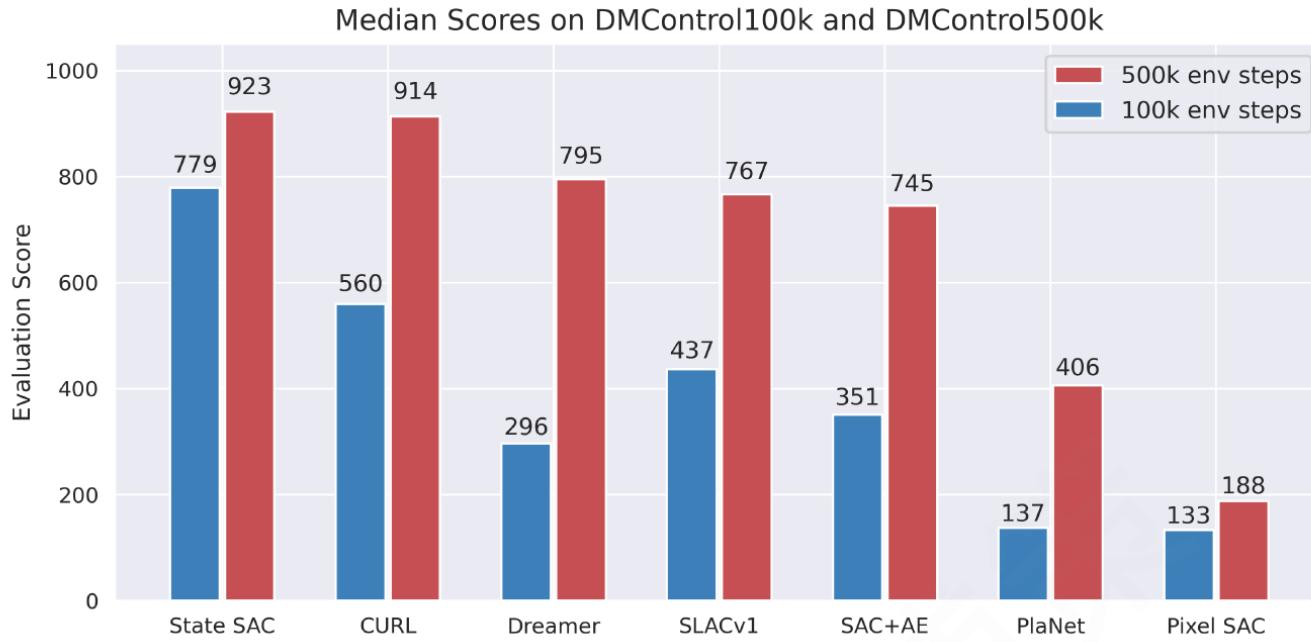
500K STEP SCORES	CURL	PLANET	DREAMER	SAC+AE	SLACv1	PIXEL SAC	STATE SAC
FINGER, SPIN	926 ± 45	561 ± 284	796 ± 183	884 ± 128	673 ± 92	179 ± 166	923 ± 21
CARTPOLE, SWINGUP	841 ± 45	475 ± 71	762 ± 27	735 ± 63	-	419 ± 40	848 ± 15
REACHER, EASY	929 ± 44	210 ± 390	793 ± 164	627 ± 58	-	145 ± 30	923 ± 24
CHEETAH, RUN	518 ± 28	305 ± 131	570 ± 253	550 ± 34	640 ± 19	197 ± 15	795 ± 30
WALKER, WALK	902 ± 43	351 ± 58	897 ± 49	847 ± 48	842 ± 51	42 ± 12	948 ± 54
BALL IN CUP, CATCH	959 ± 27	460 ± 380	879 ± 87	794 ± 58	852 ± 71	312 ± 63	974 ± 33
100K STEP SCORES							
FINGER, SPIN	767 ± 56	136 ± 216	341 ± 70	740 ± 64	693 ± 141	179 ± 66	811 ± 46
CARTPOLE, SWINGUP	582 ± 146	297 ± 39	326 ± 27	311 ± 11	-	419 ± 40	835 ± 22
REACHER, EASY	538 ± 233	20 ± 50	314 ± 155	274 ± 14	-	145 ± 30	746 ± 25
CHEETAH, RUN	299 ± 48	138 ± 88	235 ± 137	267 ± 24	319 ± 56	197 ± 15	616 ± 18
WALKER, WALK	403 ± 24	224 ± 48	277 ± 12	394 ± 22	361 ± 73	42 ± 12	891 ± 82
BALL IN CUP, CATCH	769 ± 43	0 ± 0	246 ± 174	391 ± 82	512 ± 110	312 ± 63	746 ± 91

（表1：CURL（基线 RL 算法为 SAC）和其他基线算法在 DMC 500k 和 DMC 100k 基准测试上获得的分数（报告了 10 个种子的平均值和标准差）。CURL 在 DMC 500k 基准的大多数环境（6 个中的 5 个）中实现了最好的性能。测试环境是根据基线方法中数据的可用性而选择的（作者总共在 DMC 的 16 个环境中运行了 CURL 实验，完整结果参考原论文图 7）。基线算法分别为 PlaNet [13]，Dreamer [14]，SAC+AE，SLAC，pixel-based SAC 和 state-based SAC。在所有实验结果中，作者根据 SLAC 算法每个 agent step 上梯度更新的步数（分别为 1 次和 3 次），分别记为 SLACv1 和 SLACv2。这里作者只与 SLACv1 进行比较，因为 CURL 和所有其他基线每个 agent step 都只进行了一次梯度更新。作者还在原论文表 5 中给出了每个 agent step 梯度更新 3 次的 CURL 与 SLACv2 的比较结果。）

GAME	HUMAN	RANDOM	RAINBOW	SIMPLE	OTRAINBOW	EFF. RAINBOW	CURL
ALIEN	7127.7	227.8	318.7	616.9	824.7	739.9	558.2
AMIDAR	1719.5	5.8	32.5	88.0	82.8	188.6	142.1
ASSAULT	742.0	222.4	231	527.2	351.9	431.2	600.6
ASTERIX	8503.3	210.0	243.6	1128.3	628.5	470.8	734.5
BANK HEIST	753.1	14.2	15.55	34.2	182.1	51.0	131.6
BATTLE ZONE	37187.5	2360.0	2360.0	5184.4	4060.6	10124.6	14870.0
BOXING	12.1	0.1	-24.8	9.1	2.5	0.2	1.2
BREAKOUT	30.5	1.7	1.2	16.4	9.84	1.9	4.9
CHOPPER COMMAND	7387.8	811.0	120.0	1246.9	1033.33	861.8	1058.5
CRAZY_CLIMBER	35829.4	10780.5	2254.5	62583.6	21327.8	16185.3	12146.5
DEMON_ATTACK	1971.0	152.1	163.6	208.1	711.8	508.0	817.6
FREEWAY	29.6	0.0	0.0	20.3	25.0	27.9	26.7
FROSTBITE	4334.7	65.2	60.2	254.7	231.6	866.8	1181.3
GOPHER	2412.5	257.6	431.2	771.0	778.0	349.5	669.3
HERO	30826.4	1027.0	487	2656.6	6458.8	6857.0	6279.3
JAMESBOND	302.8	29.0	47.4	125.3	112.3	301.6	471.0
KANGAROO	3035.0	52.0	0.0	323.1	605.4	779.3	872.5
KRULL	2665.5	1598.0	1468	4539.9	3277.9	2851.5	4229.6
KUNG_FU_MASTER	22736.3	258.5	0.	17257.2	5722.2	14346.1	14307.8
MS_PACMAN	6951.6	307.3	67	1480.0	941.9	1204.1	1465.5
PONG	14.6	-20.7	-20.6	12.8	1.3	-19.3	-16.5
PRIVATE EYE	69571.3	24.9	0	58.3	100.0	97.8	218.4
QBERT	13455.0	163.9	123.46	1288.8	509.3	1152.9	1042.4
ROAD_RUNNER	7845.0	11.5	1588.46	5640.6	2696.7	9600.0	5661.0
SEAQUEST	42054.7	68.4	131.69	683.3	286.92	354.1	384.5
UP_N_DOWN	11693.2	533.4	504.6	3350.3	2847.6	2877.4	2955.2

(表2: CURL (基线RL算法为 Eff. Rainbow) 和其他基线算法在 **Atari 100k** 基准测试上获得的分数。)

CURL在**26**个环境中的**7**个环境中实现了最好的性能。基线算法分别是 SimPLe, OverTrained Rainbow (OTRainbow), Data-Efficient Rainbow (Eff. Rainbow), Rainbow, Random Agent 和 Human Performance (Human)。CURL 是在 Eff. Rainbow 之上实现的，在**26**个环境中的**19**个环境上优于原始的 Eff. Rainbow。由于该基准在多个运行中易受高方差影响，作者使用了在**20个随机种子上**运行的平均结果。可以看到 CURL 在 JamesBond 和 Krull 环境上取得了超人的表现。)



(图5: CURL (+SAC) 与其他基线算法在10个种子上的平均性能对比。在500k基准时，CURL 与 State SAC 的中值分数相匹配。在100k的环境步数下，CURL 的平均得分比 Dreamer 高1.9倍。为了直接比较，作者只计算了原论文图1中的6个环境的中值（其中 SLACv1 例外只在4个环境上计算），在完整的16个 DMC 环境上 CURL 的学习曲线参考原论文图7。)

DrQ: Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels

摘要:

作者提出了一种简单的数据增强技术，该技术可以应用于标准的无模型强化学习算法，实现了直接基于图像输入的鲁棒的强化学习，而无需辅助损失或预训练。

该方法利用计算机视觉任务中常用的输入扰动来转换输入样本，作为一种对值函数和策略的正则化方法。现有的无模型方法，如 SAC，无法从图像输入有效地训练深度网络。

作者提出的增强技术的加入极大地提高了 SAC 的性能，使其能够在 DMC 上达到最先进的性能，超过了基于模型的方法和最近提出的结合对比学习的方法。

作者称提出的方法为 DrQ，即数据正则化Q，可以与任何无模型强化学习算法相结合。作者也将其应用于DQN进一步证明了这一点，并在Atari 100k基准上显著提高了其样本效率。

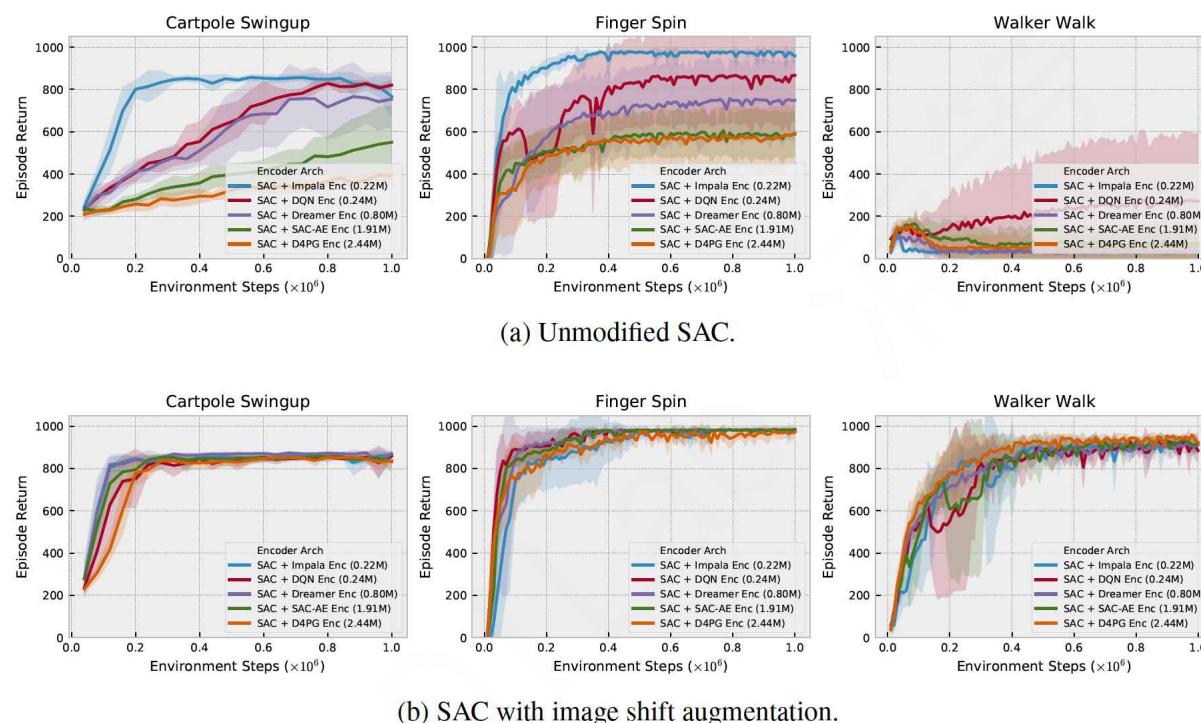
简介

- 能够直接从图像进行训练的样本效率高的深度强化学习算法将开辟控制和机器人领域的许多实际应用。然而，当环境交互是有限的、样本之间是强相关的和奖励信号是稀疏时，同时训练卷积编码器和策略网络具有挑战性。简单地尝试使用大容量编码器会导致严重的过度拟合(参见图 1a)，而较小的编码器会产生较差的表征，从而限制智能体的性能。
- 目前在机器学习社区中已经提出了大量的辅助损失函数来增强监督目标，例如权重正则化，噪声注入或各种形式的自动编码器。在 RL 领域中，也经常使用重建目标或替代任务。然而，这些目标往往与手头的任务无关，因此不能保证可以学到适合策略网络提升的表征。

- 数据增强方法已被证明在视觉和语音领域非常有效，其中具有输出不变性质的扰动 (output-invariant perturbations) 可以很容易地应用于有标签的输入样本。
- 令人惊讶的是，数据增强在 RL 社区中受到的关注相对较少，这是本文的重点。关键思想是使用标准的图像变换来扰动 (perturb) 输入观察，以及对 critic 学习的 Q 函数进行正则化，以便同一输入图像的不同变换具有相似的 Q 函数值。这样无需对标准的 Actor-Critic 算法进行修改，从而避免了其他一些方法中需要增加额外的损失函数，例如基于自动编码器，动力学模型或对比损失项的方法。

DrQ 实现

DrQ 工作的重点是为了提升样本效率，即寻求在有限的环境交互下更好地优化性能。在图6 (a) 中，作者展示了一个演示实验，该实验表明过度拟合是一个重大问题。环境使用 DMC 中的三个任务，算法使用 SAC，使用相同的策略网络架构进行训练，但使用不同的图像编码器架构，分别取自以下 RL 方法：NatureDQN，Dreamer，IMPALA，SAC-AE（与 CURL 图像编码器架构一样）和 D4PG。编码器的容量差异很大，总的参数从 220k 到 2.4M 不等。曲线显示性能随着参数总量的增加而降低，这明显地暗示存在过度拟合的问题。



(图6：在 DMC 的3个任务上，使用图像输入的 SAC 在使用不同网络容量的图像编码器上的性能曲线比较（图像编码器网络架构取自最近的RL算法）。(a) 原始的 SAC。可以看出，随着编码器容量的增加，任务性能会变得更差，表明存在过度拟合。对于最右边的 Walker Walk，所有架构都提供了很差的性能，这表明在困难的任务上，原始的 SAC 无法直接从图像输入的训练中得到合理的性能。(b): 使用图像随机偏移增强图像的 SAC。无论编码器容量如何，在所有架构上的任务性能都是相似的，与(a)相比，也有明显的性能改进，尤其是对于更具挑战性的 Walker Walk 任务，提升特别大。)

图像增强 (Image Augmentation)

在计算机视觉领域中已经开发了一系列成功的图像增强技术来对抗过度拟合。这些将变换应用于任务标签不变的输入图像，例如，对于物体识别任务，图像翻转和旋转不会改变语义标签。然而，RL 中的任务与视觉中的任务有很大不同，在许多情况下，这样的变换后的 obs 对应的奖励应该会变化。作者测试了论文附录 E 中的几种常见图像变换，并得出结论，**随机位移 (random shifts)** 在简单性和 RL 性能之间取得了良好的平衡，因此作者将增强方式限制在这种变换上。

- 图 6(b) 显示了 SAC 训练期间应用这种增强技术的结果。作者仅将数据增强应用于从 replay buffer 采样的图像，而不应用于样本收集过程，即只用于训练时损失函数的计算中，不用于与环境交互时产生动作的过程中。来自 DMC 的图像的大小是 84x84。作者先在每边填充 4 个像素（通过复制边界像素的方式），然后选择随机的 84x84 的裁剪图像，从而产生偏移了 4 个像素的图像。每次从 replay buffer 采样图像时，都会重复此过程。
- 图6(b) 显示过拟合大大减少，缩小了不同编码器架构之间的性能差距。仅使用随机偏移就使 SAC 能够实现有竞争力的性能，而无需辅助损失函数。

最优的具有不变性的状态变换 (Optimality Invariant Image Transformations)

虽然上述图像增强技术的使用方式是有效的，但它并没有充分利用 RL 任务内在的 MDP 结构。作者提出了一个通过变换输入状态来实现值函数正则化的一般框架。

理想情况下，最优的具有不变性的状态变换 (optimality invariant state transformation)

$f : \mathcal{S} \times \mathcal{T} \rightarrow \mathcal{S}$ ，应该保持该状态在应用该变换前后的 Q 值不变，即：

$$Q(s, a) = Q(f(s, \nu), a) \text{ for all } s \in \mathcal{S}, a \in \mathcal{A} \text{ and } \nu \in \mathcal{T}$$

其中 ν 是变换函数 $f(\cdot)$ 的参数，是从所有参数组成的集合 \mathcal{T} 中采样得到的。变换函数 $f(\cdot)$ 的一个具体实例是上一节的图像的随机偏移 (random image translations)。对于每个状态，变换允许生成具有相同 Q 值的多个代理状态 (surrogate states)，从而提供了一种减少 Q 函数估计方差的机制。

对于策略 π 和任意的 state 分布 $\mu(\cdot)$ ，在估计 Q 值时，

- 如果不用状态变换，相当于采样了一个状态 s^* 和一个动作 $a^* \sim \pi(\cdot | s^*)$ ：

$$\mathbb{E}_{\substack{s \sim \mu(\cdot) \\ a \sim \pi(\cdot | s)}} [Q(s, a)] \approx Q(s^*, a^*)$$

- 如果使用了状态变换，作者可以从一个状态的分布 $\mu(\square)$ 中采样 s ，并求得到期望的一个估计值。如下式所示，对 s^* 进行了 K 次变换，得到 K 个不同的 Q 值估计，然后求平均得到最终的估计值，这样就减少了 Q 值估计的方差。**注意对于每个变换后的状态，其对应的动作也不同。**

$$\mathbb{E}_{\substack{s \sim \mu(\cdot) \\ a \sim \pi(\cdot | s)}} [Q(s, a)] \approx \frac{1}{K} \sum_{k=1}^K Q(f(s^*, \nu_k), a_k) \text{ where } \nu_k \in \mathcal{T} \text{ and } a_k \sim \pi(\cdot | f(s^*, \nu_k))$$

- 在 Q-learning 中可以有 2 种对 Q 函数进行正则化的方式：

- Target Q 增强：对于每个 transition tuple (s_i, a_i, r_i, s'_i) ，在计算 target 值时，可以应用上面的数据增强：对下一时刻状态进行 K 次变换得到 K 个 Q 值，然后取平均 Q 值作为最终的下一时刻状态值的估计：

$$y_i = r_i + \gamma \frac{1}{K} \sum_{k=1}^K Q_\theta(f(s'_i, \nu'_{i,k}), a'_{i,k}) \text{ where } a'_{i,k} \simeq \pi(\cdot | f(s'_i, \nu'_{i,k}))$$

- Current Q 增强：参数更新公式总体上不变。但是在 current Q 计算时，也可以对当前时刻状态进行 M 次变换得到 M 个 Q 值，然后取平均 Q 值作为当前时刻状态值的估计：

$$\theta \leftarrow \theta - \lambda_\theta \nabla_\theta \frac{1}{NM} \sum_{i=1, m=1} (Q_\theta(f(s_i, \nu_{i,m}), a_i) - y_i)^2$$

- 当同时使用这两种正则化方法时， $\nu_{i,m}$ 和 $\nu'_{i,k}$ 是独立选取的。

Our approach: Data-regularized Q (DrQ)

DrQ 的方法是上面介绍的三种独立正则化机制的联合体：

1. 采样后进行训练之前对输入图像进行一次变换。
2. 对下一时刻状态进行 K 次变换得到 K 个 Q 值，然后取平均 Q 值作为下一时刻状态值的估计。
3. 对当前时刻状态进行 M 次变换得到 M 个 Q 值，然后取平均 Q 值作为当前时刻状态值的估计。

伪代码

Algorithm 1 DrQ: Data-regularized Q applied to a generic off-policy actor critic algorithm.

Black: unmodified off-policy actor-critic.

Orange: image transformation.

Green: target Q augmentation.

Blue: Q augmentation.

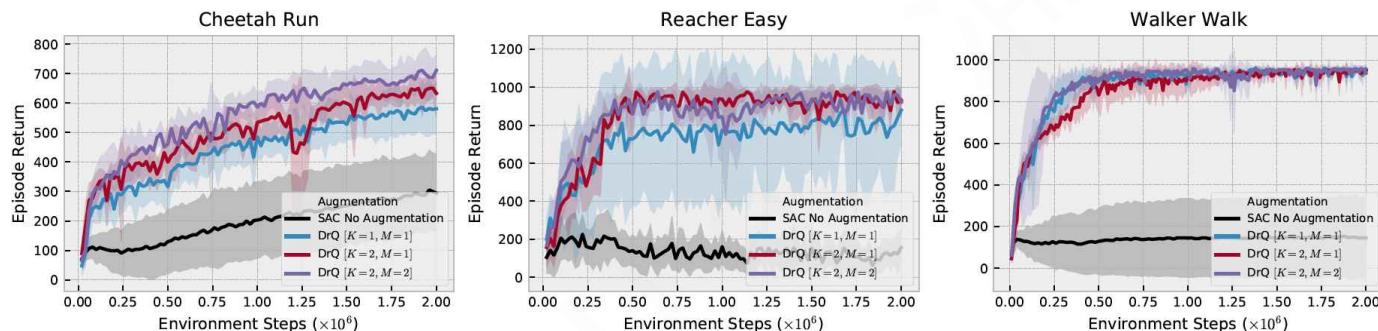
Hyperparameters: Total number of environment steps T , mini-batch size N , learning rate λ_θ , target network update rate τ , image transformation f , number of target Q augmentations K , number of Q augmentations M .

```

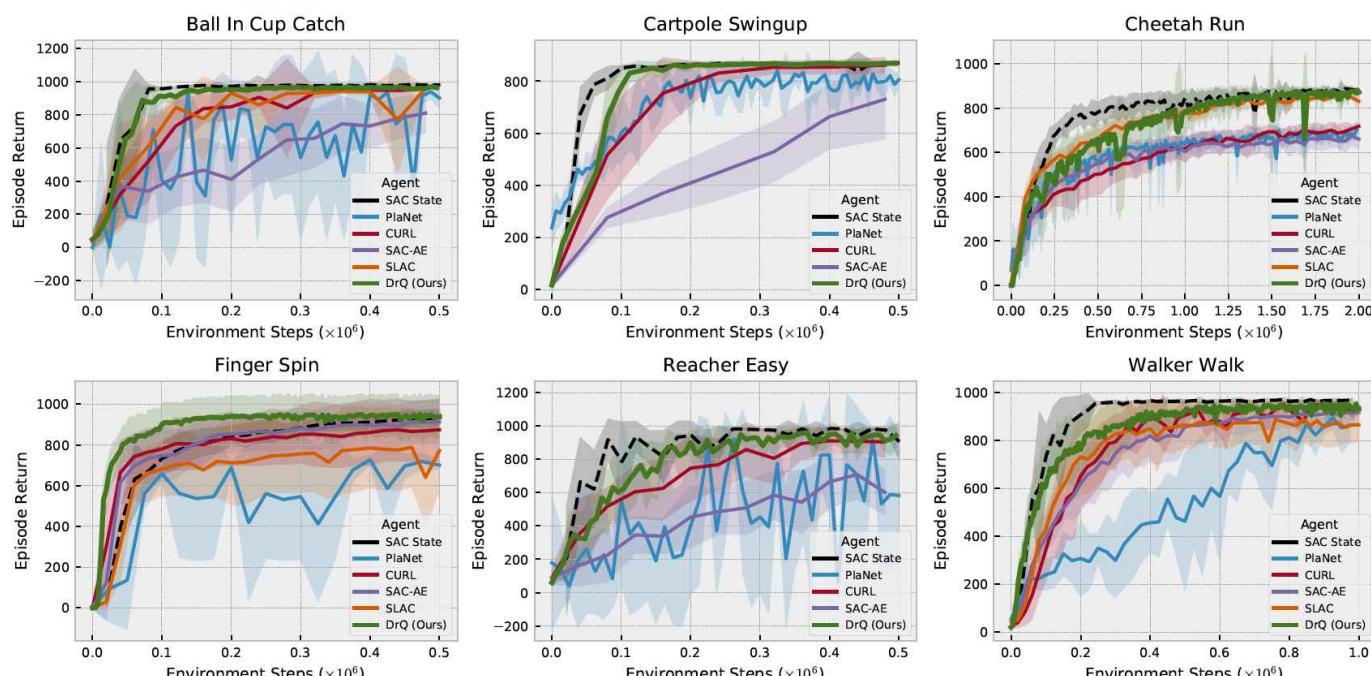
for each timestep  $t = 1..T$  do
     $a_t \sim \pi(\cdot|s_t)$ 
     $s'_t \sim p(\cdot|s_t, a_t)$ 
     $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, r(s_t, a_t), s'_t)$ 
    UPDATECRITIC( $\mathcal{D}$ )
    UPDATEACTOR( $\mathcal{D}$ )  $\triangleright$  Data augmentation is applied to the samples for actor training as well.
end for
procedure UPDATECRITIC( $\mathcal{D}$ )
     $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^N \sim \mathcal{D}$   $\triangleright$  Sample a mini batch
     $\{\nu'_{i,k} | \nu'_{i,k} \sim \mathcal{U}(\mathcal{T}), i = 1..N, k = 1..K\}$   $\triangleright$  Sample parameters of target augmentations
    for each  $i = 1..N$  do
         $a'_i \sim \pi(\cdot|s'_i)$  or  $a'_{i,k} \sim \pi(\cdot|f(s'_i, \nu'_{i,k}))$ ,  $k = 1..K$ 
         $\hat{Q}_i = Q_{\theta'}(s'_i, a'_i)$  or  $\hat{Q}_i = \frac{1}{K} \sum_{k=1}^K Q_{\theta'}(f(s'_i, \nu'_{i,k}), a'_{i,k})$ 
         $y_i \leftarrow r(s_i, a_i) + \gamma \hat{Q}_i$ 
    end for
     $\{\nu_{i,m} | \nu_{i,m} \sim \mathcal{U}(\mathcal{T}), i = 1..N, m = 1..M\}$   $\triangleright$  Sample parameters of Q augmentations
     $J_Q(\theta) = \frac{1}{N} \sum_{i=1}^N (Q_\theta(s_i, a_i) - y_i)^2$  or  $J_Q(\theta) = \frac{1}{NM} \sum_{i,m=1}^{N,M} (Q_\theta(f(s_i, \nu_{i,m}), a_i) - y_i)^2$ 
     $\theta \leftarrow \theta - \lambda_\theta \nabla_\theta J_Q(\theta)$   $\triangleright$  Update the critic
     $\theta' \leftarrow (1 - \tau)\theta' + \tau\theta$   $\triangleright$  Update the critic target
end procedure

```

实验

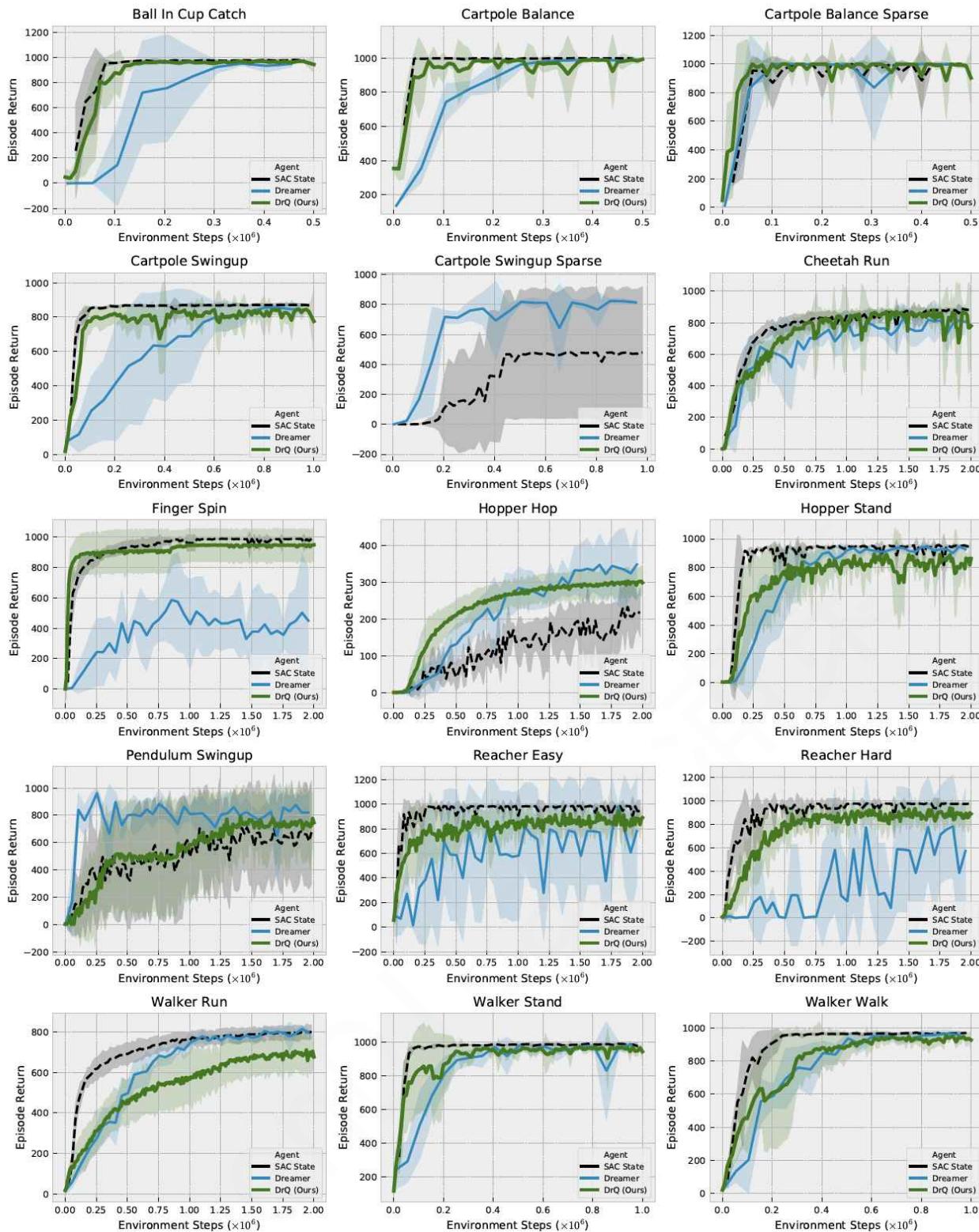


(图7: 在 DMC 任务上, 结合三种正则化技术的 DrQ 在K和M取不同组合时的性能曲线。黑色线: 标准 SAC。蓝色线: DrQ [K=1, M=1], 即只是输入图像使用随机偏移增强技术的 SAC。红色线: DrQ [K=2, M=1], 即输入图像随机偏移 + Target Q 增强。紫色线: DrQ [K=2, M=2], 即输入图像随机偏移 + Target Q + Current Q 增强。这三种正则化方法对应于算法1具有不同超参数K, M , 并且都优于未使用任何增强方法的 SAC。注意: DrQ [K=1, M=1] 相当于使用了特定超参数和数据增强类型的 RAD 算法。)

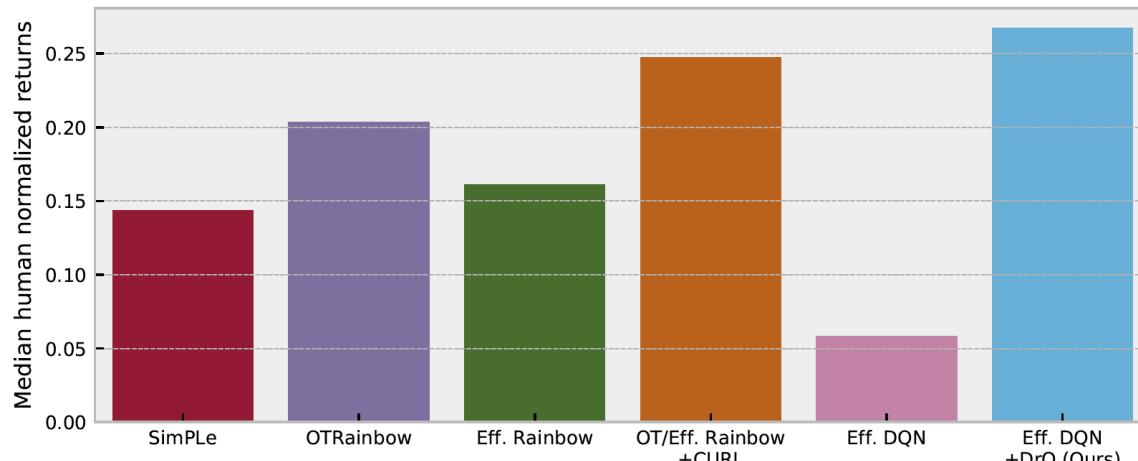


(图8: PlaNet 论文提到的 DMC 基准测试。作者的算法 (DrQ [K=2, M=2]) 优于其他方法, 并展示了最先进的性能。此外, 在其中几个任务上, DrQ 甚至能够与直接在内部状态(而不是图像)上训练的

SAC的上限性能相一致。最后，作者的算法不仅显示出相对于其他方法的采样效率更高，而且在绝对时间方面也更快。)



(图9：Dreamer 论文提到的 DMC 基准测试。作者的方法 (DrQ [K=2, M=2]) 再次证明了在 15 个选定任务中的 12 个任务中优于 Dreamer 的性能。在许多情况下，它还达到了直接从内部状态 (而不是图像) 上学习的 SAC 的上限性能。)



(图10：Atari 100k 基准测试。与当前先进的基线相比，作者的方法 (DrQ [K=1, M=1]，基线 RL 算法是 Efficient DQN) 虽然实现技术更简单，但实现了最先进的性能。特别注意，(Efficient DQN +

DrQ) 相对 Efficient DQN (蓝色线相比粉红色线) 的巨大改进。相比之下，利用 Data Efficient Rainbow 和 OTRainbow 技巧的 CURL 的收益比其基线 RL 方法要小得多。)

Does Self-supervised Learning Really Improve Reinforcement Learning from Pixels?

简介

- 作者研究了自监督学习 (Self Supervised Learning, SSL) 是否可以一致地改善图像输入的强化学习方法。
- 作者对联合优化 SSL 和 RL 损失的对比强化学习框架 (例如 CURL) 进行了扩展与推广，并对各种自监督损失进行了大量的实验。实验表明，在使用相同数量的数据和数据增强的情况下，现有的 SSL+RL 框架不能比仅利用图像增强的基线算法带来一致的有意义的改进。
- 作者通过演化搜索算法 (evolutionary searches)，以找到应用到 RL 上的多个自监督损失的最佳组合方式，但发现这样找到的最佳损失组合也不能一致地超过只利用精心设计的图像增强的基线方法。
- 作者在多个不同的环境 (DMC, Atari, 真实世界的机器人环境) 中评估了这些方法后，确认没有一个单一的自监督损失或图像增强方法可以在所有的环境上表现都好，并且目前的 SSL 和 RL 联合优化框架是有局限性的。最后，作者对框架中多个因素进行了消融研究，并分析了不同方法学到的表征的特性。

预备知识

强化学习

在本论文中，在具有连续动作空间和离散动作空间的环境上，分别采用用 SAC (Soft Actor Critic) [4] 和 Rainbow DQN [5] 作为基线 RL 算法。

- Soft Actor Critic 是一种 off-policy actor-critic 算法，它利用最大熵 RL 框架来鼓励 agent 在训练期间探索更多的状态。它维护一个策略网络 π_ψ 和两个 critic 网络 Q_{ϕ_1} 和 Q_{ϕ_2} 。 π_ψ 的目标是同时实现奖励的预期总和和 γ 折扣熵的最大化，其中熵是为了鼓励 agent 在学习期间进行探索。
- Rainbow DQN 是 DQN 的一个变种，结合了多种改进技术，包括: Double Q learning、优先采样、噪声网络、值分布 RL、dueling 网络和多步奖励等。

Pairwise Learning

- 在计算机视觉中，具有以下原理的框架，作者称之为成对学习 (Pairwise Learning): 对于一个具有 dual-stream 结构的编码器，给定一个输入，会输出两个表征，利用这两个表征之间的语义不变性 (semantic invariance) 来学习视觉表征。
- 一般的成对学习方法：
 - 首先通过对输入样本进行一系列的随机图像增强来生成多个增强的视图 (multiple augmented views)，然后在表征空间中对具有相同语义的视图进行聚类。在基于这种框架的方法中，我们可以选择使用对比损失来排斥具有不同语义的样本。
- 在本文中，作者主要关注四个代表性的成对学习方法：MoCo [7], BYOL [8], SimSiam [9] 和 DINO [10]。作者在原论文附录A.1中对这些方法进行了详细的阐述和比较。

基于图像输入的 RL 中的表征学习

最近的 RL 工作中，探讨了怎样从图像观测中学习到更好的表征，以加速策略的学习。包括2个方向：

- 一个方向是使用图像增强进行策略学习，其中 RAD 和 DrQ 使用简单的图像增强技术实现了显著的性能提升。

- 另一个方向是将 SSL 与 RL 相结合，其中有两个代表性的方法，SAC+AE 和 CURL。

RAD (Reinforcement Learning with Augmented Data) [6]：研究了不同类型的图像增强对 image/state 输入的影响。通过对输入图像进行随机平移或随机裁剪，RAD 仅通过图像增强就能显著提高数据效率，没有使用任何辅助的损失函数。

DrQ (Data-regularized Q) [2]：进一步研究了图像增强对于策略学习的影响。DrQ 对输入图像进行两次图像增强，并将对这两个增强后的图像的 Q 值进行平均后的结果，作为原始输入图像的 Q 值。

DrQ v2 [19] 是 DrQ 的后续扩展版本，改用 DDPG (深度确定策略梯度) 作为 RL 方法，带来了规划好的探索噪声 (scheduled exploration noise)，以控制不同学习阶段的探索水平，并开源了图像增强和 replay buffer 的快速实现代码。

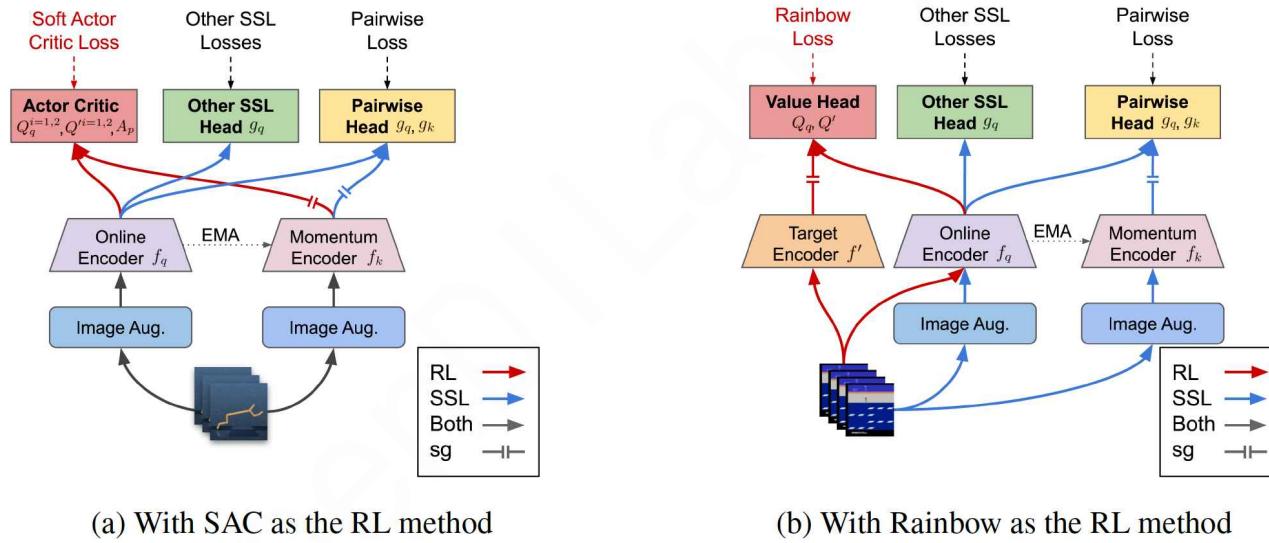
SAC+AE：利用 RAE (确定性正则化自动编码器) 来取代 β -VAE，以提高学习稳定性。RAE 与 SAC 联合训练，在一个 batch 中交替执行 SAC 更新和 RAE 更新。

CURL：通过在图像编码器的末端引入一个额外的对比学习网络头，将对比学习与在线 RL 算法相结合。与前面提到的 SAC+AE 类似，这里的对比损失和强化学习损失在训练时交替应用。

强化学习中的自监督技术

为了有效地评估不同的自监督损失对强化学习策略学习的影响，作者通过在 RL 框架中加入通用的自监督学习头，扩展了以前论文中广泛使用的联合学习框架。作者保持与 CURL 中相同的 RL 方法，即在具有连续动作空间的任务中使用 SAC，在具有离散动作空间的任务中使用 Rainbow DQN。

通用联合学习框架 (General Joint Learning Framework)



(图11：SSL + RL 的通用联合学习框架。红色箭头代表 RL 数据流，蓝色箭头代表 SSL 数据流，黑色箭头代表两者共享的数据流，sg 代表没有梯度传播。)

结合 SAC

Algorithm 1 Update SAC with Self-supervised Losses
 Green: additional operations for SSL; Orange: only for BYOL and DINO.

```

procedure UPDATESACWITHSSL( $s$ : current state,  $s'$ : next state,  $a$ : action,  $r$ : reward,  $d$ : done signal,
step: model update step counter,  $f_q$ : online encoder,  $f_k$ : target/momentum encoder,  $A_p$ : actor head,  $Q_q^i$ : online critic head,  $Q'^i$ : target critic head,  $\tau$  : target/momentum network update rate,  $g_q$ : online SSL head,  $g_k$ : momentum SSL head)
   $s_a, s'_a \leftarrow \text{IMAGEAUGMENTATION}(s), \text{IMAGEAUGMENTATION}(s')$ 
   $s_p, s'_p \leftarrow \text{IMAGEAUGMENTATION}(s), \text{IMAGEAUGMENTATION}(s')$ 
   $f_q, Q_q^{i=1,2}, A_p \leftarrow \text{UPDATESOFTACTORCRITIC}(s_a, s'_a, a, r, d)$ 
   $f_k, Q'^{i=1,2} \leftarrow \tau(f_q, Q_q^{i=1,2}) + (1 - \tau)(f_k, Q'^{i=1,2})$   $\triangleright$  EMA update of SAC
   $g_k \leftarrow \tau g_q + (1 - \tau)g_k$   $\triangleright$  EMA update of the momentum SSL head
   $f_q, g_q \leftarrow \text{UPDATESSL}(s_a, s'_a, s_p, s'_p, a, r)$ 
end procedure
  
```

图11 展示了一个通用的联合学习框架，图11 (a) 使用 SAC 作为基线 RL 方法。

网络结构：

- 未经修改的 SAC 算法包含：一个在线编码器 (online encoder) f_q ，一个目标(或称为动量)编码器 (target (or momentum) encoder) f_k ，以及一个 actor head A_p 。每个编码器后面都有两个 critic head (或 target critic head)。
- 为了加入SSL 损失：在在线编码器之后附加了一个 self-supervised head g_q 。如果需要计算 Pairwise Learning 的损失，可以在目标编码器之后串联一个 momentum SSL head g_k 。

训练过程：

- 对于每次采样的 minibatch，作者首先对当前状态 s 和下一时刻状态 s' 应用图像增强，并使用 **增强后的图像更新SAC模型** ($f_q, Q_q^{i=1,2}, A_p$)。请注意，出于稳定性的考虑，作者在更新 actor head A_p 时不更新在线编码器 f_q 的参数。然后，通过指数移动平均法 (EMA) 更新 target critic head。
- 随后，如果需要的话，也对 momentum SSL head g_k 进行 EMA 更新。最后，通过最小化自监督损失更新在线编码器 f_q 和 self-supervised head g_q 。通过在每个 minibatch 中交替执行 RL 和 SSL，共同训练框架中的所有组件。

算法1中提供了 SAC+SSL 的伪代码。

结合 Rainbow DQN

Algorithm 2 Update Rainbow with Self-supervised Losses Green: additional operations for SSL; Orange: only for BYOL and DINO.

```

procedure UPDATERAINBOWDQNWITSSL( $s$ : current state,  $s'$ : next state,  $a$ : action,  $r$ : reward,  $d$ : done,
step: model update step counter,  $f_q$ : online encoder,  $f'$ : target encoder,  $Q_q$ : online value head,  $Q'$ : target value
head,  $f_k$ : momentum networks,  $\tau$  : momentum network update rate,  $g_q$ : online SSL head,  $g_k$ : momentum
SSL head,  $w_{SSL}$ : weights of self-supervised losses)
     $s_a, s'_a \leftarrow \text{IMAGEAUGMENTATION}(s), \text{IMAGEAUGMENTATION}(s')$ 
     $s_p, s'_p \leftarrow \text{IMAGEAUGMENTATION}(s), \text{IMAGEAUGMENTATION}(s')$ 
     $\mathcal{L}_{SSL} \leftarrow \text{CALCULATESSLLOSS}(s_a, s'_a, s_p, s'_p, a, r)$ 
     $\mathcal{L}_{Rainbow} \leftarrow \text{CALCULATERAINBOWLOSS}(s, s', a, r, d)$ 
     $\mathcal{L} \leftarrow \mathcal{L}_{Rainbow} + w_{SSL} \mathcal{L}_{SSL}$ 
     $f_q, Q_q, g_q \leftarrow \text{ONLINENETWORKSUPDATE}(\mathcal{L})$ 
     $f', Q' \leftarrow f_q, Q_q$                                  $\triangleright$  Copy parameters from online networks to target networks
     $f_k, g_k \leftarrow \tau(f_q, g_q) + (1 - \tau)(f_k, g_k)$        $\triangleright$  EMA update of momentum networks and SSL head
end procedure

```

图11(b) 展示了如何将 SSL 应用于 Rainbow DQN。

网络结构：

- 未经修改的 Rainbow DQN 保持着一个在线编码器 (online encoder) f_q ，一个目标编码器 (target encoder) f' ，然后是两个 state value heads Q_q, Q' 。作者按照 CURL 的建议，引入一个额外的动量编码器 (momentum encoder) f_k 和self-supervised head g_q, g_k 。

训练过程：

- 对于每次采样的 minibatch，自监督损失是使用增强的图像计算的，而 **RL 损失是使用原始数据计算的**。最后，通过最小化自监督损失，更新在线编码器 f_q 和self-supervised head g_q 。

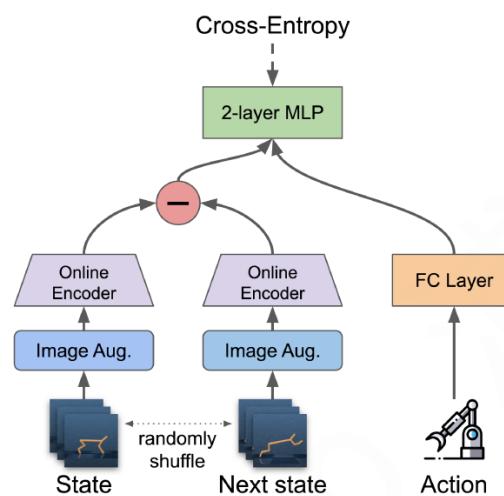
算法1中提供了 Rainbow DQN+SSL 的伪代码。

自监督学习的损失函数

本论文研究的自监督损失大致分为四类：

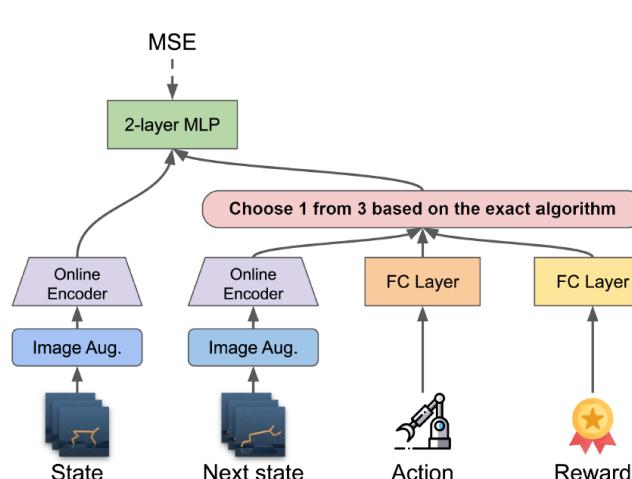
- Pairwise Learning：
 - 方法的介绍参考章节【预备知识-成对学习】。文章除了测试 CURL, BYOL, DINO 和 SimSiam 这四种表征学习方法，还测试了两种在成对学习框架引入 RL 特定变量的方法：CURL-w-Action 和 CURL-w-Critic。CURL-w-Action 是基于 CURL 的，区别在于不仅将对比损失应用在图像表征上，而是应用在图像表征和 actor 网络输出的拼接向量上。同样地，CURL-w-Critic 将对比损失应用在现有的图像表征与 critic 网络的输出的拼接向量上。

- Transformation Awareness:
 - 通过在学习过程中，让从表征中能够感知 (awareness) 到不同的变换（如旋转、拼图和时间排序），这个思想改善了计算机视觉中的许多下游任务，如图像分类和动作识别。通常情况下，这种意识可以通过显式地要求分类器能够从图像表征中识别出对图像应用了何种变换来获得，本文研究了下面2种简单的分类损失函数：
 - RotationCLS：一种鼓励能从表征中判别出空间转换情况的方法 (spatial transformation awareness)。受RotNet 和 E-SSL 的启发，作者对输入图像进行 0° 、 90° 、 180° 和 270° 的旋转。让分类器从视觉表征中预测旋转角度，并通过交叉熵损失进行训练。
 - Shuffle Tuple：鼓励编码器通过预测两个帧是否依次出现来培养对动作因果关系的认识。作者通过随机打乱 (Shuffle) 状态转换元组中的当前时间状态图像和下一时刻状态图像，并预测其顺序是否是被打乱的。分类器的输入也包括动作，因为有些transition是可逆的，即 $s -> s'$, $s' -> s$ 都是合理的 transition。ShuffleCLS 损失函数计算的整体结构如图12所示。



(图12：ShuffleCLS)

- Reconstruction：
 - 用沙漏结构 (hourglass architecture) 重构输入图像已被证明是学习图像表征的有效方法。对于本文实验的方法 SAC+AE 中，作者通过改变输入和重建目标为增强的图像对其进行了扩展。RAE 中的重建损失和正则化项都没有变动。
 - 最近 Masked AutoEncoder (MAE) 研究了基于 patch 的 Vision Transformers 的重建任务。MAE 的目标包括从输入的 masked image patches 中重建整个图像。受此启发，作者将 SAC+AE 改编为 SAC+MAE，将增强的输入图像替换为 masked version image，只对 masked patches 的重建误差进行惩罚。
- RL Context Prediction：



(图13：通用的 RL context 预测。)

	Extract-A	Extract-R	Guess-A	Guess-F	Predict-F	Predict-R	Extract-AR	Guess-AF	Predict-FR
Rep. of s'	Input	Input	-	Output	Output	-	Input	Output	Output
Action a	Output	-	Output	-	Input	Input	Output	Output	Input
Reward r	-	Output	Input	Input	-	Output	Output	Input	Output

(表1: RL context prediction losses 输入输出的可能组合方式。注意: 当前时刻状态 s 是一个固定输入, 这里的 Input 表示额外的输入。其中 Extract 代表额外输入为 next state, Guess 代表额外输入为 reward, Predict 代表额外输入为 action。)

- 对于一个 transition tuple (s, a, s', r) 。使用当前时刻状态 s 的表征作为 SSL Head 的一个固定输入, 并选择 s', a, r 中的几个元素作为 SSL Head 的额外输入, 然后使用两层MLP预测 transition tuple 中的其他元素, 预测过程参见图13, 具体的可能组合参见表1。
- 对于连续输出, 采用均方误差 (MSE) 损失, 而对于离散目标 (例如离散动作空间中的动作) 采用交叉熵损失。
 - Balanced RL Context Prediction: 文章附录 A.2.3 指出, 如果有预测包括两个元素的方法 (即 Extract-AR、Guess-AF、Predict-FR) 之前是先将两个输出拼接为一个向量, 再使用监督学习来计算 loss。以 Extract-AR 为例, 先将两个输出 (Action, Reward) 拼接后再计算 loss 的方法会导致两个 Output (Action, Reward) 中有更高维度的 target (Action) 相较于维度较低的 target (Reward) 获得了更多的 penalty。因此, 文章提出 “Balanced” 的方法, 即分别对 Action 和 Reward 计算 Prediction loss, 然后总的 loss 取它们的平均值, 这样就缓解了上面提到的问题。作者也实验证明了, 通过调整组合权重, "Balanced" 技巧带来了明显的性能改进。这表明, 我们需要仔细设计两种损失的组合方式, 随着组合损失数量的增加, 这种设计也变得越来越棘手。

Evolving Multiple Self-supervised Losses

除了单一的自我监督损失或两个损失的手工组合, 作者还研究了在 RL+SSL 联合学习框架中, 多个自监督损失的权重是如何影响策略学习的。作者借鉴了 ELo (Evolving Losses) [17] 的思想, 使用演化搜索方法, 来搜索多个 SSL loss 的权重, 以期望搜索到最佳的 SSL loss 组合, 具体地, 使用的是粒子群优化 (Particle Swarm Optimization, PSO) 算法。优化过程可以用下面的公式描述:

$$\underset{m_{j=1,2}, w_{i=1,\dots,N_l}}{\operatorname{argmax}} \text{IQM}(\mathcal{R}_{\text{env}}^{\text{seed}=1,\dots,5}(m_{j=1,2}, w_{i=1,\dots,N_l}))$$

其中 $m_{j=1,2}$ 表示 online encoder 和 momentum encoder 对于图像的增强程度, 实际是指 random crop 前的图像大小。 $w_{i=1,\dots,N_l}$ 表示 N_l 个 SSL 损失的权重。IQM (interquartile mean) 表示四分平均数, 即对于一个向量, 仅使用第一个和第三个 四分位数 之间的数据求得的平均值。

实验

实验分为3个部分, 即总体表现与消融分析、关于学习得到的表征的分析、不同学习框架的影响, 后面2个部分在论文附录中:

- 第1部分 (总体表现与消融分析): 在 DMC, Atari, 和一个 Real Robot 上测试算法表现, 展示了不同的自监督 loss 对 RL 的影响; 在 DMC 上进行消融实验, 经验地分析了 SSL+RL 中几个关键因素的影响;
- 第2部分 (表征分析, 附录A.6): 分析了学习到的表征的多个 metric, 并给出了不同 metric 与 agent 得分之间的关系;
- 第3部分 (学习框架, 附录A.7): 探索了不同于 joint learning framework 的另一种预训练学习框架。

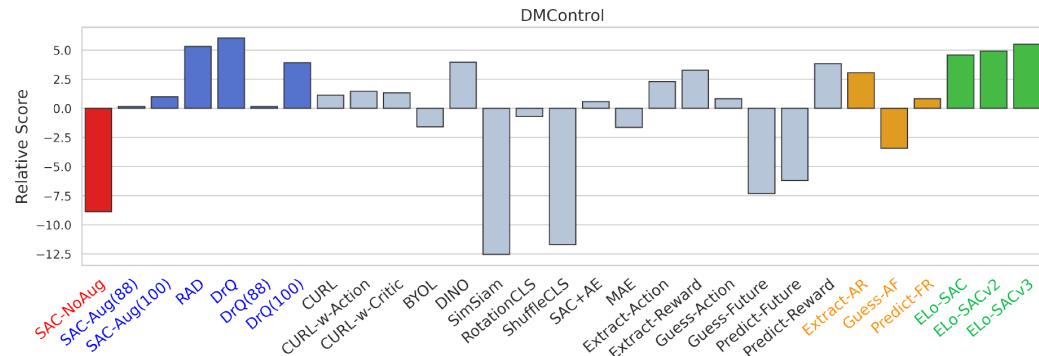
主要评估指标是 Relative Score: 给定一个 agent (即一种算法), 首先计算每个环境上多次运行的的 IQM 分数, 然后根据所有环境的 IQM 分数的 mean 和 std 进行标准化, 再对所有环境取均值。

$$S_{\text{Relative}}^A = \sum_{e \in E} (\text{IQM}^{A,e} - \text{mean}(\text{IQM}^e)) / \text{std}(\text{IQM}^e)$$

- 其中 $\text{IQM}^{A,e}$ 表示 agent A 在环境 e 上取得的四分平均数。

总体表现与消融分析

DMC 实验



(图14：六个 DMC 任务的相对分数，环境步长=100k，batch size=512，种子数=10。)

SAC-NoAug 表示没有使用图像增强，而所有其他方法都使用了图像增强；**蓝色的方法（如DrQ）** 只利用了图像增强，没有任何SSL；**黑色的方法（如CURL）** 应用了一个自监督损失；**橙色的方法（如 Extract-AR）** 手动结合了两个自监督损失；**绿色的方法** ELo-SAC、ELo-SACv2 和 ELo-SACv3 结合了多个自监督损失和演化搜索的特定权值。

从这个图中可以看出，现有的基于 SSL 的联合学习框架的方法都没有达到比 DrQ 更好的性能，而 DrQ 只使用精心设计的图像增强。ELo-SAC方法取得了比所有自监督方法更高的相对分数，但它的表现仍然比 DrQ 和 RAD 差，只有 ELo-SACv3 例外，它比 RAD 略好。

注意：方法括号中的数值表示先将原始84*84的图像变换到这个图像大小，例如 SAC-Aug(88) 指先将原图像 resize为88*88，再 random crop 到84*84。)

实验设置

环境：DMC 包含许多具有挑战性的视觉连续控制任务，这些任务被最近的论文广泛利用。选取了常见于以前论文中的**六个环境**。

基线算法：

- 作者对章节【自监督学习的损失函数】中介绍的所有自监督方法以及两个重要的基线 SAC-NoAug 和 SAC-Aug(100) 进行了评估。
- 其他只利用图像增强的方法，如 RAD 和 DrQ 也被作为基准进行比较。
- 在 SAC+AE 的情况下，作者提供了增强的图像以进行公平的比较，这是与原论文不同的配置。

请参考原论文 [3] 附录A.2.5，了解各个算法的详细比较和它们应用的确切数据增强。

作者主要遵循 CURL 中报告的超参数和测试环境，为了简单起见，作者在所有环境中使用相同的学习率 10^{-3} 。所有的方法都是在10个随机种子下，以训练批次大小为512的100k环境步骤为基准，并且它们共享相同的策略网络容量。

结果分析

每种测试算法在DMC上的相对得分如图14所示。作者也强烈建议读者查看表11的全部结果和表12的另外两个困难环境下的结果，以了解全貌。

- 从图14可以看到，**没有一个基于 SSL 的方法比 DrQ 和 RAD 取得更好的性能**，这两个方法是精心设计的充分利用了特定图像增强的优势。与基线 SAC-Aug(100) 相比，19个具有自监督损失的方法中的11个比基线强化学习差。一些 SSL方法（如SimSiam, ShuffleCLS）破坏了策略学习，导致性能甚至比 SAC-NoAug 更差，这表明不适当使用自我监督损失会破坏图像增强带来的好处。

- 然后，关于组合损失，Guess-AF 和 Predict-FR 是手动设计的，**将两个单独的损失组合在一起并不比单一的自监督损失好**（参考图14中的 Guess-Action 和 Predict-Reward）。
- ELo-SAC 和 ELo-SACv2 通过在一个任务 ("cheetah run") 中搜索找到所需的组合参数。这样的组合参数可以推广到DMC上的其他环境，因为整体性能比搜索空间中的任何单一方法都要好。在进行**搜索的任务 "cheetah run" 上**，他们在所有基于 SSL 的方法中获得了最佳结果。这证明了 ELo-SAC 的可行性，并意味着通过演化搜索获得的组合可以推广到 DMC 的其他环境。然而，在 "finger, spin" 和 "reacher, easy" 中较弱的表现使 ELo-SAC 平均而言还是比（不使用任何自监督技术的）DrQ 差。
- 有趣的是，尽管 ELo-SAC 和 ELo-SACv2 有不同的搜索空间，它们之间有类似的性能模式 (similar performance pattern)。相比之下，ELo-SACv3 通过在六个任务上同时搜索，找到了一个总体上**更好的组合**。尽管它在 "walker, walk "和 "reacher, easy" 中取得了最高分，但它在 "cartpole, swingup" 和 "cheetah, run" 中的表现比 ELo-SAC 和 ELo-SACv2 差。这样的实验结果是判断不同任务和自监督方法性质的一个窗口。

DMC 上的消融实验

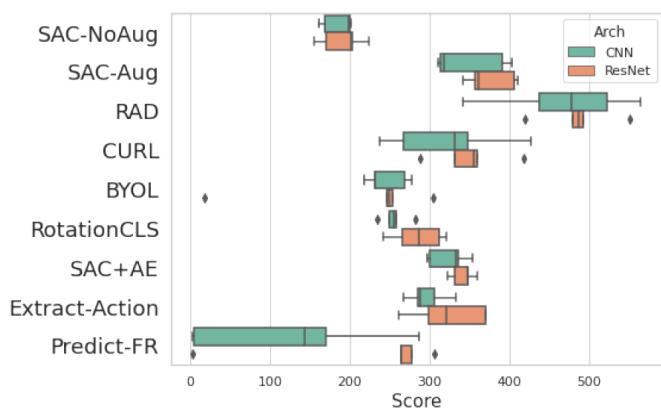


图15：主干编码器的消融实验

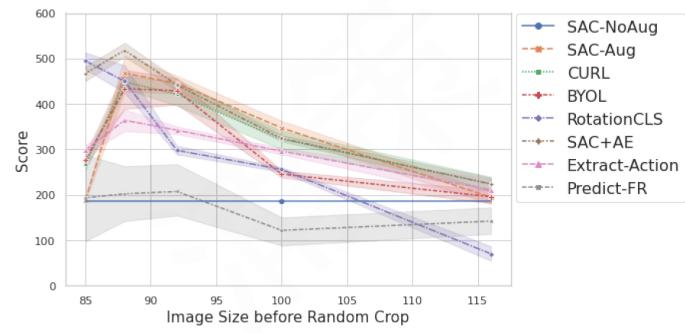


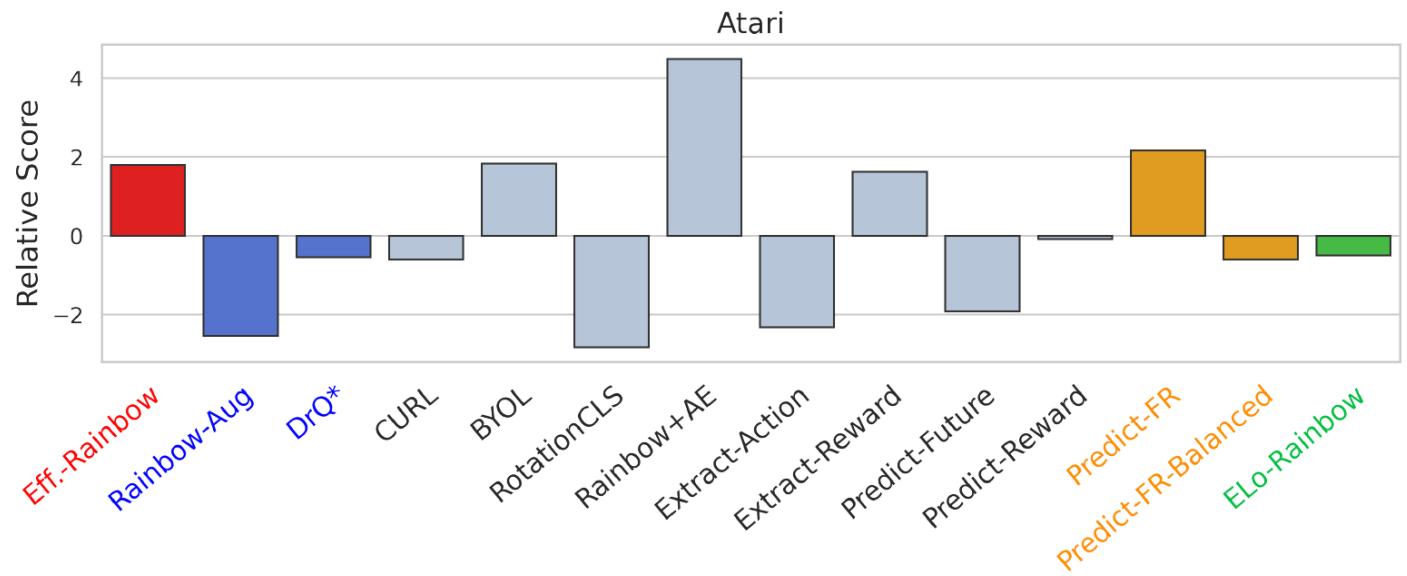
图16：随机裁剪增强 (random crop augmentation) 的消融实验

作者在 SAC-Aug(88)、SAC-Aug(100) 和 RAD 上的实验结果表明了图像增强对应的超参数的重要性，这三种方法之间唯一的区别是应用的增强方式不同。

作者对 cheetah run 中的图像增强方法 random crop 进行了消融研究。除了环境步长被设置为400k和 batch size 被减少到128之外，所有的超参数参考原论文表2。

- 图16显示了随机裁剪和平移 (random crop and translate) 的大小对 agent 取得的分数的贡献。在使用固定的裁剪后尺寸时，随机裁剪前的图像尺寸与图像增强程度呈线性关系：图像尺寸越大，增强的程度越大 (the larger the image size, the stronger the augmentation)。有一个趋势是，随着图像增强的加大，得分首先增加，然后减少。总之，在设计一个有或没有SSL的RL系统时，仔细设计图像增强是至关重要的。
- 然后，在图15中，作者研究了一个不同的视觉编码器主干 ResNet，即用一个 residual block 替代最后两个卷积层，该 residual block 的层数和通道数与 CNN 的基线相同。在所有这些方法上，使用 ResNet 骨干网络稍微改善了性能。
- 作者也鼓励读者在原论文 [3] 附录A.3 中查看更多关于图像增强 (如random translate)，学习率、编码器层和激活函数的消融实验。

Atari 实验



(图17：不同算法在七个 Atari 游戏上的 Relative Score，环境步长=400k, batch size=32, 种子数=20。一个方法的颜色反映了其类别，与图4相同。总的结果显示，**使用图像增强技术的 RL 算法并没有给Atari游戏的策略学习带来好处，这与DMC有很大不同**。即使给 SSL head 更多计算和额外的模型容量，大多数结合自监督损失的方法也未能带来改进。只有 **Rainbow+AE** 优于 Efficient Rainbow，这与 SAC+AE 的效果不一致。ELO-Rainbow 取得的结果甚至比在搜索空间中的一些基于 SSL 的单一方法更差，如 BYOL 和 Rainbow+AE。不同游戏之间的高方差和图像领域的差距使得 ELO-Rainbow 在寻找通用于所有环境的组合损失方面面临极大的挑战。)

实验设置

- 环境：Atari 2600 游戏也是具有挑战性的基准，具有离散的动作空间。作者在这个基准中选择了七个游戏作为选定的方法。
- 基线算法：所有的方法都使用 Efficient Rainbow 作为 RL 方法，它是 Rainbow 的一个提高数据效率后的变体。请注意，Efficient Rainbow 作为一个基线，没有利用图像增强的优势。因此，作者也以 Rainbow-Aug 为基准，它本质上是 Efficient Rainbow+augmented images。作者对所有适用的方法使用了 CURL 报告的相同的图像增强和超参数。为了进行公平的比较，DrQ* 的增强也采用了 CURL 的方法，这与原 DrQ 论文的建议不同。作者把作者的设定称为 DrQ*，以区别于原始的 DrQ。同样地，Rainbow+AE 也采用了增强的图像。对于每个游戏，作者运行 20 个随机种子，并以 400K 个环境步骤（100K 个模型步骤，frame skip 为 4）。作者报告了与 DMC 相同的四分位数、标准偏差和相对分数（见表 13）。

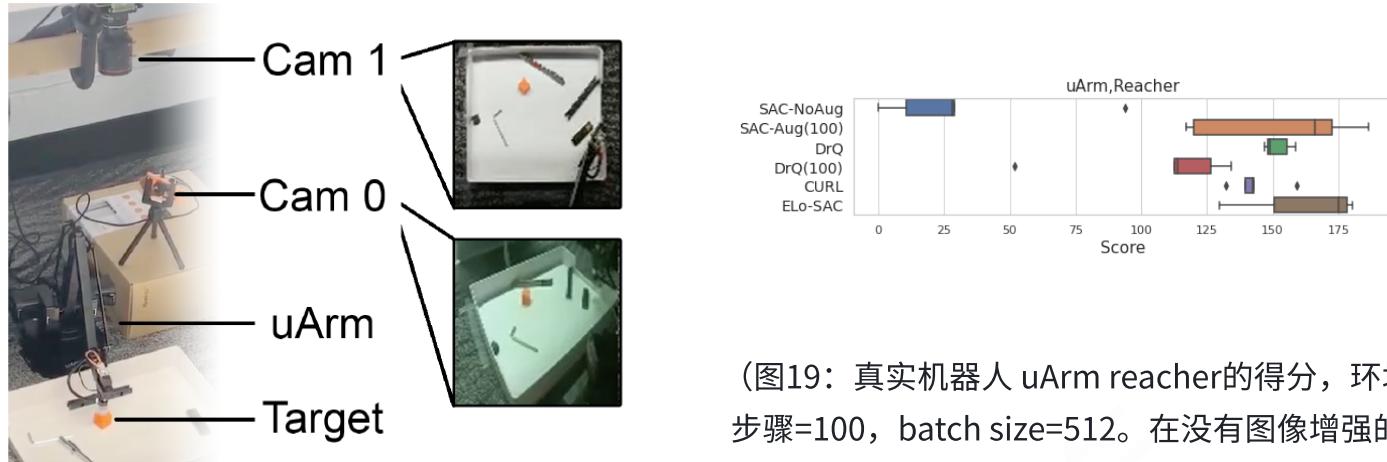
结果分析

图17显示了各个算法在七个不同任务的 Relative Score 的对比图。

- 首先，与没有任何图像增强或自我监督学习的基线 Efficient Rainbow 相比，**Rainbow-Aug 在为 RL 增加图像增强的情况下整体表现更差**。这表明 CURL 中用于自监督学习的图像增强技术很难迁移到其他算法上。
- 同样，DrQ* 取得的性能比 Efficient Rainbow 差，表明在 Atari 上 Rainbow+ 图像增强 并不像 DMC 上的 SAC+ 图像增强 那样有利于策略学习。
- 基于图像增强技术对性能影响的不一致，RL 算法在 Atari 上应用图像增强技术时，需要进一步的研究。
- 对于自监督损失，BYOL、Rainbow+AE、Extract-Reward 和 Predict-Reward 获得了比 CURL 更好的性能。然而，相比基线 Efficient Rainbow 算法，只有 Rainbow+AE 表现出明显的改善，并且超过了所有其他方法，有趣的是，这与 SAC+AE 在 DMC 上的表现不一致。

- 在 DMC 上显示出相当大的改进的 Predict-FR-Balanced 方法 (通过手动结合两个自监督损失), 在 Atari 上却未能超过 PredictReward 方法。在 Frostbite 中搜索的 ELo-Rainbow, 只在 demon attack 和 frostbite 环境中比基线算法好。Atari 基准上各个环境的高方差使得演化搜索非常困难。此外, Atari 游戏之间图像存在巨大的语义差距, 这使得 ELo-Rainbow 更难在 Atari 上的多个游戏中泛化。

真实机器人实验



4(图18: 真实机器人环境设置。)

(图19: 真实机器人 uArm reacher 的得分, 环境步骤=100, batch size=512。在没有图像增强的情况下, agent 未能学习到有效的策略。)

实验设置

- 作者进一步在真实世界的机器人环境中进行实验, 即使用 uArm reacher。目标是将执行器尽可能快地移到目标物体附近。作者的自主训练环境设置和实验结果分别显示在图18和图19中 (环境设置细节请查看原论文附录A.5)。作者用五个不同的随机种子对所有的方法进行基准测试, 除了在表3中报告的修改的超参数外, 使用与 DMC 实验相同的超参数, 结果如图19所示。

结果分析

- 令人惊讶的是, 在这个现实世界的环境中, agent 未能在没有任何图像增强的情况下学习有效的策略。ELo-SAC 算法 (在 cheetah run 上进行参数搜索得到的最佳参数组合), 实现了显著优于 DrQ 和 CURL 的性能, 这与作者之前在 DMC 上的实验结果有很大不同。单纯的图像增强 (即 SAC-Aug(100)) 就足以超越使用自监督的 CURL 方法。图19中的 DrQ 和 DrQ(100) 的性能有很大不同说明需要一些工程设计才能使图像增强技术发挥作用。

表征分析

作者参考论文 [11], 通过四个 metric 分析在 DMC cartpole swingup 环境上学习到的表征, 四个 metric 分别是 Dynamics Awareness, Diversity, Orthogonality 和 Predict State from Visual Representation。这里给出四个 metric 与 relative score 的皮尔逊相关系数 (Pearson correlation coefficient), 其他实现细节与结果详见原论文。

Dynamic Awareness	Orthogonality	Diversity	Prediction MSE
-0.284	0.435	0.111	-0.625

(表14: relative score 和 metric 之间的皮尔逊相关系数。)

- 主要结论如下:
 - 根据上图中的皮尔逊相关系数的绝对值大小, Prediction MSE (对应 Predict State from Visual Representation) 与 agent 的表现有较强的相关性, 即较低的 Prediction MSE 通常会带来较高的score。这表明一个好的表征应该能够较为精确地捕捉环境状态信息。
 - Orthogonality 这一 metric 与 score 之间有一定的正相关, 这表明好的表征要对不同状态有一定的区分度。

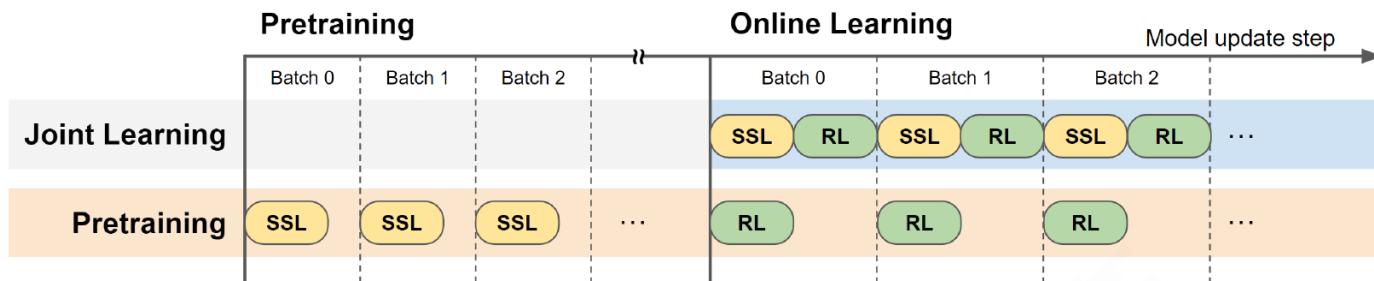
学习框架

作者将 CV 中的 two-stage 预训练框架推广到 RL+SSL 的方法中：

注解：two-stage 预训练框架，即先使用自监督损失进行预训练，然后使用下游任务 (downstream task) 损失微调 (finetune) 网络。

具体来说，第一阶段，先在训练好的一个 SAC-Aug(100) agent 产生的数据上，利用 SSL loss 更新 encoder；第二阶段，使用第一阶段得到的 encoder 的初始化参数，再结合 RL loss 来训练一个 SAC-Aug(100) agent。

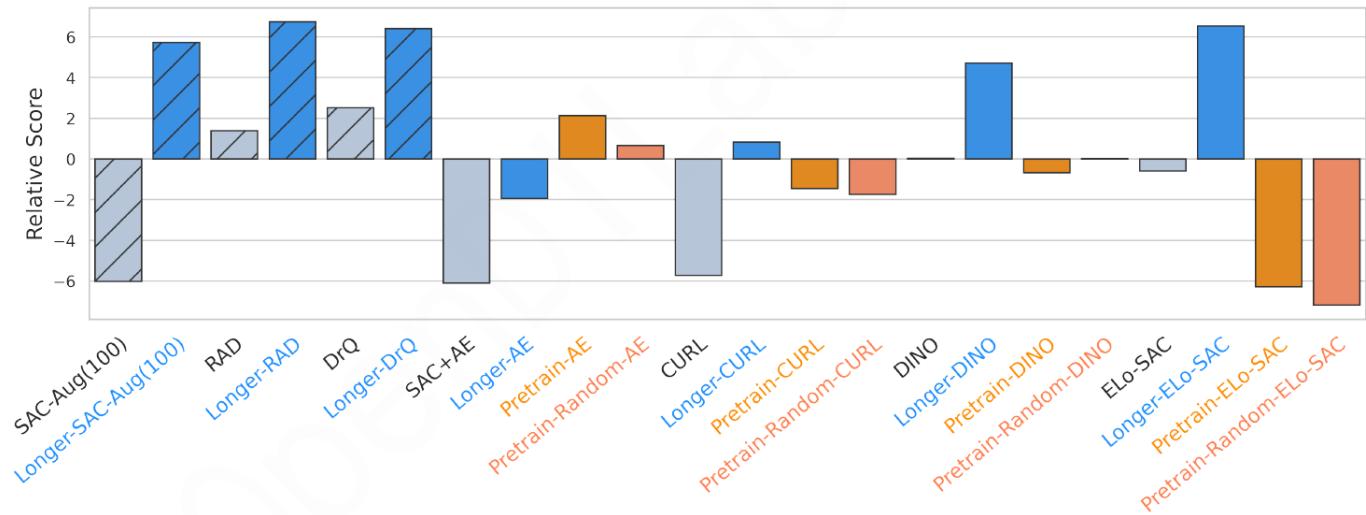
作者提出的预训练框架与 SSL+RL 中常见的 joint learning framework 的示意图如下所示：



(图20：SSL + RL 的两个学习框架，圆角矩形表示使用里面标注的损失来更新模型一步。)

由上图可以看到，在模型的更新次数一样的情况下，Pretraining 比 Joint learning 多了一倍的交互数据，为了公平比较，作者额外增加了另一个带有 "Longer" 前缀的 Joint learning 算法设置，即将与环境交互的数据总数与采用 Pretraining 框架的 agent 对齐。

作者在 DMC 的6个环境上测试了这两种框架的性能：



(图21：将SSL与RL相结合的两个学习框架的 relative score。带对角线的条形图代表只使用图像增强技术而没有任何自监督损失的方法。)

实验结论如下：

- 在模型的更新次数相同的情况下，Pretraining framework 的性能优于Joint Learning framework；
- 当 Joint learning framework 给定相同量的交互数据时，即带有 “Longer” 前缀的方法，通常比 Pretraining framework 性能更好，这说明了在DMC上，预训练的优势来源于额外的数据，而不是框架本身。

参考文献

- [1] CURL: Contrastive Unsupervised Representations for Reinforcement Learning.
- [2] DrQ: Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels.
- [3] Does Self-supervised Learning Really Improve Reinforcement Learning from Pixels?
- [4] SAC: Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the International Conference on Machine Learning (ICML), pages 1861–1870. PMLR, 2018.
- [5] Efficient Rainbow: Hado P van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? Advances in Neural Information Processing Systems (NeurIPS), 32, 2019.
- [6] RAD: Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. Advances in Neural Information Processing Systems (NeurIPS), 33:19884–19895, 2020.
- [7] MoCo: Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9729–9738, 2020.
- [8] BYOL: Jean-Bastien Grill, Florian Strub, Florent Altch., Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. Advances in Neural Information Processing Systems (NeurIPS), 33:21271–21284, 2020.
- [9] SimSiam: Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 15750–15758, 2021.
- [10] DINO: Mathilde Caron, Hugo Touvron, Ishan Misra, Herv. J.gou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9650–9660, 2021.
- [11] Han Wang, Erfan Miah, Martha White, Marlos C Machado, Zaheer Abbas, Raksha Kumaraswamy, Vincent Liu, and Adam White. Investigating the properties of neural network representations in reinforcement learning. arXiv preprint arXiv:2203.15955, 2022.
- [12] Skew-Fit: State-Covering Self-Supervised Reinforcement Learning. ICML 2020
- [13] PlaNet: Hafner, Danijar, et al. "Learning Latent Dynamics for Planning from Pixels." arXiv preprint arXiv:1811.04551 (2018).
- [14] Dreamer: Hafner, Danijar, et al. "Dream to control: Learning behaviors by latent imagination." arXiv preprint arXiv:1912.01603 (2019).
- [15] CPC: van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.
- [16] SimCLR: Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations, 2020.
- [17] AJ Piergiovanni, Anelia Angelova, and Michael S Ryoo. Evolving losses for unlabeled video representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and

Pattern Recognition (CVPR), 2019.

- [18] Weng, Lilian. (May 2021). Contrastive representation learning. Lil' Log. <https://lilianweng.github.io/posts/2021-05-31-contrastive/>.
- [19] DrQ v2: Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In Proceedings of the International Conference on Learning Representations (ICLR), 2022.