arXiv:2301.02464v1 [cs.LG] 6 Jan 2023

架构、正则化与重放（ARR）：一种灵活的持续学习混合方法

Vincenzo Lomonaco
计算机科学系 比萨大学
vincenzo.lomonaco@unipi.it

Lorenzo Pellegrini计算机科学与工程系 博洛尼亚大学 l.pellegrini@unibo.it

Gabriele Graffieti计算机科学与工程系 博洛尼亚大学 gabriele.graffieti@unibo.it

Davide Maltoni计算机科学与工程系 博洛尼亚大学 davide.maltoni@unibo.it

摘要

近年来，我们重新关注能够突破基本独立同分布假设、应对非平稳环境的机器学习方法，尤其是深度表征学习。这些环境会受到各种分布变化或样本选择偏差的影响。在此背景下，人们提出了多种基于架构先验、正则化器和重放策略的计算方法，其成效因开发和评估它们的具体场景而异。然而，要设计出全面的混合解决方案，使其能够灵活、通用地应用，并可调节效率与效果之间的权衡，似乎仍是一个遥远的目标。本文提出了架构、正则化与回放（ARR），这是著名的 AR1 算法及其变体的一种混合式推广。它不仅能在经典场景（例如类别增量学习）中取得最先进的结果，还能推广至由 CIFAR-100、CORe50 和 ImageNet-1000 等真实世界数据集生成的任意数据流。

1 引言

持续机器学习是一个具有挑战性的研究问题，具有深远的科学和工程意义 [22]。一方面，它动摇了依赖独立同分布（*iid*）假设的经典机器学习系统的基础；另一方面，它为构建高效且可扩展、能够像人类一样学习和思考，并能迅速适应外部世界不断变化特性的以人为中心的人工智能系统提供了路径。然而，尽管近年来机器学习和深度学习领域对这一主题的兴趣迅速高涨，且相关科学活动十分活跃，这一愿景仍远未实现。

尽管大多数持续学习算法显著减轻了特定场景下灾难性遗忘的影响，但很难将这些结果推广到它们未针对其进行设计的场景中（缺乏鲁棒性和普适性）。此外，它们大多

书籍章节预印本。

侧重于持续学习的纵向且单一的方法，这些方法基于正则化、回放或对底层预测模型进行架构调整。

本文总结了为持续学习制定混合策略所付出的努力，旨在使这些策略在真实应用场景中更稳健、适用范围更广且效果更佳。具体而言，我们将重点介绍“架构设计、正则化 与回放”（ARR）方法的定义：对著名的 AR1 算法 [30] 及其所有变体[23, 36], 进行通用化的重新表述与推广；可以说，它也是最早提出的混合持续学习方法之一 [34] （第 4 节）。
通过在 CIFAR-100、CORe50 和 ImageNet-1000 等先进基准上进行一系列实验，我们展示了所提方法相较于其他现有先进方法的效率和有效性（第 5 节）。随后，我们讨论了可调参数，它们能够轻松控制效果与效率之间的权衡，例如潜在重放层的选择（第 5.4 节）以及回放内存大小 （第 5.5 节）。最后，我们讨论了当前 ARR 实现在 Avalanche[27]中的移植情况（第 6 节）。

2 背景与问题表述

持续学习 (CL) 主要关注这样一种学习理念：从非平稳、短暂的数据流中学习。这些数据可以分多个计算步骤进行处理，除非被明确记忆，否则无法再次访问。在无先验持续学习场景中，数据以流式方式到达，形成一个（可能无限长的）序列 S，其中包含我们所称的学习经验 e，即 $S=e_1,\dots,e_n$。为简便起见，我们假设这是一个监督分类问题，每个经验阶段 e_i 由一个样本批次 $\mathcal{D}^i$ 构成，其中每个样本都是一个由输入数据和目标数据组成的元组 $\langle x_k^i, y_k^i\rangle$，标签 y_k^i 来自集合 $\mathcal{Y}^i$，该集合是类别全集 $\mathcal{Y}$的一个子集。不过，我们指出，这种表述很容易推广到不同的 CL 问题。通常， $\mathcal{D}^i$ 被划分为独立的训练集 $\mathcal{D}^i_{train}$ 和测试集 $\mathcal{D}^i_{test}$。持续学习算法 $\mathcal{A}^{CL}$ 是一个具有如下签名的函数 [19, 2]:

$$\mathcal{A}^{CL}\colon \langle f^{CL}_{i-1}, \mathcal{D}^i_{train}, \mathcal{M}_{i-1}, t_i\rangle \to \langle f^{CL}_i, \mathcal{M}_i\rangle \tag{1}$$

其中， f^{CL}_i是模型在经验阶段e_i上训练后学到的模型； $\mathcal{M}_i$ 是过去知识缓冲区（也可以为空），其中存储着来自先前经验阶段的内容，例如过去的样本或激活值，通常大小固定。 t_i 是一个任务标签，可用于识别正确的数据分布（或任务）。本文所有实验均假设最具挑战性的情形，即无法获取 t_i。通常，持续学习算法可使用的资源有限，因此在设计时应确保能够扩展到大量训练经历，同时不会随着时间推移增加内存或计算开销。持续学习算法的目标是最小化整个数据流 $\mathcal{L}_S$ 上的损失，其中数据流 S由 n 个不同的经验阶段组成:

$$\mathcal{L}_S(f^{CL}_n, n) = \frac{1}{\sum_{i=1}^{n} |\mathcal{D}^i_{test}|} \sum_{i=1}^{n} \mathcal{L}_{exp}(f^{CL}_n, \mathcal{D}^i_{test}) \tag{2}$$

$$\mathcal{L}_{exp}(f^{CL}_n, \mathcal{D}^i_{test}) = \sum_{j=1}^{|\mathcal{D}^i_{test}|} \mathcal{L}(f^{CL}_n(\boldsymbol{x}^i_j), y^i_j), \tag{3}$$

其中，损失 $\mathcal{L}(f^{CL}_n(\boldsymbol{x})\ y)$是在单个样本 $\langle \boldsymbol{x}, y\rangle$上计算的，例如分类问题中的交叉熵。因此，该表述的主要假设是，随着时间推移遇到的所有概念仍然相关（漂移仅为虚拟漂移），且不存在冲突证据。这是深度持续学习文献中一个相当常见的假设；相关研究更关注如何随着时间推移构建稳健且通用的表征，而非构建能够快速适应不断变化情形的系统。

3 探索混合持续学习方法

我们在图1中展示了部分最流行、最新的持续学习方法，并将它们归入上述类别及其组合。在图中，我们区分了采用回放（回放明确存储的训练样本）的方法与采用生成式回放（回放潜在表征

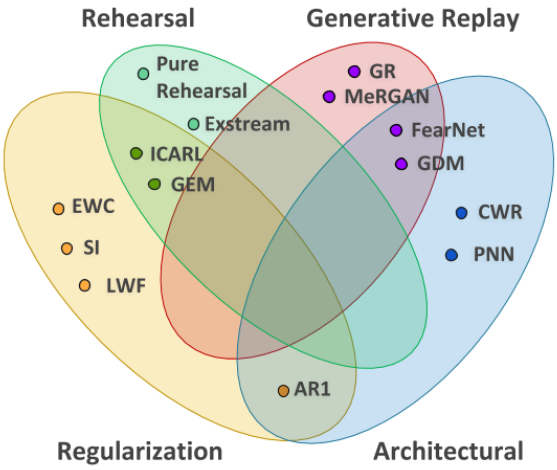

图 1： 一些最受欢迎的持续学习策略的维恩图：CWR [24], PNN [40], EWC [17], 突触智能[44], LWF [21], ICARL [38], GEM [29], FearNet [15], GDM [35], ExStream [13], 纯回放、GR [41], MeRGAN [42] 和 AR1 [31]。回放和生成式回放这两个上层类别可视为回放策略的子集。

或训练样本）的方法。至关重要的是，尽管提出的方法日益增多，但对于哪些训练方案和性能指标更适合评估持续学习模型，目前尚未达成共识。针对监督学习和无监督学习任务，研究者提出了不同的指标集来评估持续学习性能（例如 [12, 16, 10]）。由于缺乏标准化指标和评估方案，如何才算赋予一种方法持续学习能力，仍不明确。尤其是，许多持续学习模型仍需要大量计算资源和内存资源，这限制了它们从数据流中实时学习或以合理延迟进行学习的能力。

还值得注意的是，尽管针对各个主要类别已经提出了许多方法，但要找到能够灵活利用三种不同方法（即架构、正则化和回放）各自往往相互正交的优势的混合算法解决方案，仍然很困难；具体采用哪种方案取决于特定应用需求以及效率与效果之间的权衡。不过，有证据表明，有效的生物持续学习系统（例如人脑）会运用所有这些不同的功能。

本文认为，在终身学习机器的近期及长期未来中，我们将看到人们对开发混合持续学习算法的兴趣显著增长[28]，并提出 ARR，作为首批切实实现这一愿景的方法之一。

4 ARR：架构、正则化与回放

架构、正则化与回放算法，简称 ARR，是 AR1算法及其变体（CWR+、CWR*、AR1*、AR1Free）的灵活泛化 [26, 36]。通过对超参数进行适当初始化，ARR 可根据所需的效率与效能权衡，实例化为上述算法 [37]。按照该领域已形成的趋势 [7]，它可以使用预训练参数，也可以从随机初始化开始。伪代码 1 详细描述了 ARR，其基于三个主要组件：架构、正则化和回放。

4.1 架构组件

架构方法的核心理念是隔离 并保留一部分参数，同时添加新参数以容纳新知识。CWR+是 CWR [24]的演进版本，其伪代码见 [30] 中的算法 2。它为输出分类层维护两组权重：cw 是用于推理的巩固权重（用于保持稳定性），tw 是用于训练的临时权重（用于保持可塑性）；cw 在首次经验阶段开始前初始化为 0，之后迭代更新，而 tw 则在每次训练经历开始前重置为 0。

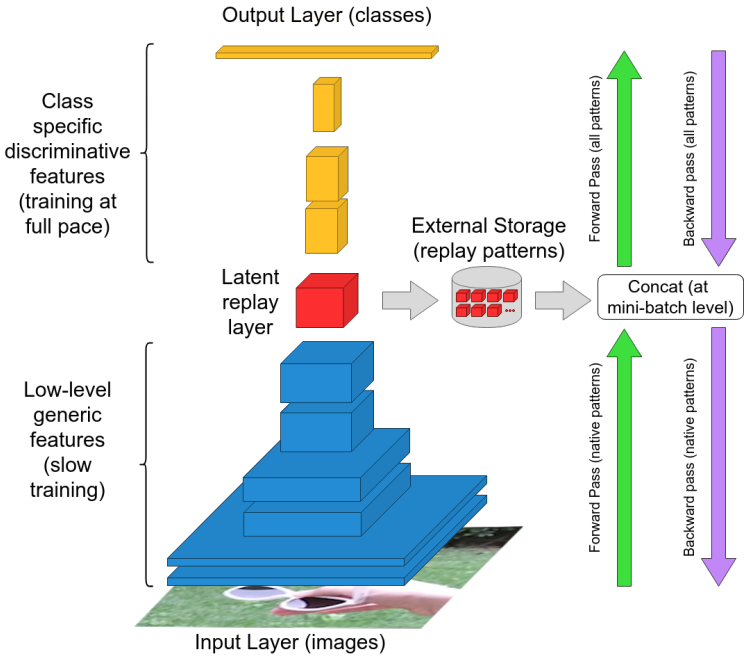

图2： ARR [36]的架构图。

在 [30], 中，作者提出了 CWR+ 的扩展版本 CWR*，可用于类别增量 [39] 和带重复的类别增量设置 [6]；具体而言，在带重复的类别增量设置下，后续经验阶段既包含新类别的样本，也包含已遇到类别的样本。对于已知类别，不再将权重重置为 0，而是重新载入巩固权重。此外，在巩固步骤中，现在使用加权和：第一项表示过去的权重，第二项表示当前训练经历的贡献。第一项所用的权重 $wpast_j$ 与比率 $\frac{past_j}{cur_j}$ 成正比，其中 $past_j$是过去经验阶段中遇到的类别 j 的样本总数，而 cur_j 是该类别在当前经验阶段中的样本数。如果存在大量规模较小的非独立同分布训练经历，最新经验阶段的权重可能会过低，从而阻碍学习过程。为避免这一问题，采用平方根对 $wpast_j$的最终值进行平滑处理。

4.2 正则化组件

广为人知的弹性权重巩固 （EWC）纯正则化方法 [17] 通过根据模型权重相对于先前遇到的数据分布和任务的重要性估计值，对其进行相应约束来控制遗忘。为此，在分类方法中，会在传统交叉熵损失中加入一个正则化项，其中模型的每个权重 θ_k都会被拉回其最优值 θ_k^*，拉回强度 F_k 与其对建模过去知识的重要性估计值成正比：

$$L = L_{cross}(\cdot) + \frac{\lambda}{2} \cdot \sum_k F_k \cdot (\theta_k - \theta_k^*)^2. \tag{4}$$

突触智能 (SI) [44]也是一种广为人知的 EWC 轻量级变体。它不在每次经验结束时更新费 Fisher 信息 F[1]，而是通过整合 F_k 来获得。

[1]本文中，对于 EWC 和 ARR 的实现，我们采用一个随时间更新的单一费 Fisher 矩阵，遵循 [30]中所述的方法。

它利用梯度下降过程中已有的信息，沿权重轨迹对损失进行积分。对于这两种方法，与方程 4 对应的权重更新规则为：

$$\theta_k^{'}=\theta_k-\eta\cdot\frac{\partial L_{cross}(\cdot)}{\partial\theta_k}-\eta\cdot F_k\cdot(\theta_k-\theta_k^*) \tag{5}$$

其中，η 是学习率。这个方程有两个缺点。首先，λ 的值必须经过仔细校准：事实上，如果其值过高，某些参数的最优值可能会被越过，从而导致发散（参见 [30]第2节中的讨论）。其次，必须维护两份完整的模型权重副本，分别存储 θ_k 和 θ_k^*，导致每个权重的内存消耗翻倍。为了解决上述问题，[26]的作者建议用以下规则替换方程5中的更新规则：

$$\theta_k^{'}=\theta_k-\eta\cdot(1-\frac{F_k}{max_F})\cdot\frac{\partial L_{cross}(\cdot)}{\partial\theta_k} \tag{6}$$

其中，max_F 是权重重要性的最大值（我们将大于 max_F的 F_k 值裁剪至 max_F）。基本上，对于重要性最高的权重（$F_k=max_F$），学习率会降至0（即完全冻结）；对于 $F_k=0$的权重，学习率则保持为 η。值得注意的是，这两种更新规则的作用方式不同：前一种规则仍会使高 F_k 的权重沿着与梯度相反的方向移动，然后再朝过去的（最优）值迈出一步；后一种规则则倾向于完全冻结高 F_k的权重。然而，在 [26], 进行的实验中，两种方法得出的结果相似，因此第二种方法更可取，因为它解决了上述缺点。既可以对低级通用特征，也可以对类别特定判别特征施加学习参数正则化，如 AR1* 中所实现的那样。不过，为了简化起见，在 ARR 中，我们仅考虑将此类正则化项应用于后一类特征，因为冻结或缓慢微调低级通用特征已被证明是一种有效策略。

4.3 回放组件

在 [36, 33] 中，研究表明，一种非常简单的回放实现方式（下文称为原生复习）并不逊于 iCaRL 等更复杂的方法。该方式会针对每次训练经历，从该经历的样本中随机抽取一个子集，加入外部存储，以替换外部记忆中一个同样随机选取的子集。因此，在 [36] 中，作者选择了简化方案，并比较了在 CORe50 NICv2 – 391 [26]上采用和不采用回放训练的 MobileNetV1[2] 中 CWR* 和 AR1* 的学习趋势。他们采用了 [25] 中介绍的相同协议和超参数，并使用了包含 1,500 个样本的回放记忆。研究结果清楚地表明，即使是适度规模的外部记忆（约占训练集总量的 1.27%），也能有效提高这两种方法的准确率，并缩小其与累积上界之间的差距；对于该模型，累积上界为 ∼85%。

在深度神经网络中，靠近输入端的层（通常称为表征层）通常负责低级特征提取；在大型数据集（如 ImageNet）上经过适当的预训练后，这些层的权重往往相当稳定，可在不同应用中重复使用。另一方面，较高层倾向于提取类别特定判别特征，调整这些层通常有助于最大限度地提高准确率。

一种潜在重放方法（见图2） [36] 可据此提出：我们无需在外部记忆中以原始数据的形式保存输入样本副本，而是可以存储给定层的激活量（该层称为潜在重放层）。为保持表征稳定并确保存储的激活值有效，我们建议减缓潜在重放层以下所有层的学习速度，同时让其上方的层以全速学习。在极限情况下，如果较低层完全冻结（即减速至 0），潜在重放在功能上等同于从输入端进行回放，但由于只需让较少的样本在整个网络中进行前向和反向传播，且网络通常会在较高层压缩信息，因此能够节省计算量和存储空间。

[2]该网络在 ImageNet-1k 上进行了预训练。

算法 1 ARR 伪代码： Θ 是表征层的类别共享参数；记号 $cw[j]$ / $tw[j]$ 用于表示对应类别 j的已巩固权重组/临时权重组。请注意，该版本在新类别（NC）设置下仍然适用；这里将其视为新类别与实例（NIC） [24]的一种特殊情况。事实上，由于 NC 中当前体验里的类别此前从未出现过，第 7 行的步骤会为 $\mathcal{D}^i$ 中的类别加载 0 值，因为 cw_j 初始化为 0；而在巩固步骤（第 13 行）中， $wpast_j$ 的值始终为 0。外部随机记忆 RM 会在各训练体验中逐步填充。请注意，为了保持不同训练体验的贡献大致均衡，逐步减少要添加的样本数量 h ，但不会强制执行任何类别平衡约束。λ 是正则化强度， α 是重放层。若省略，三个输入参数的默认值均为 0。

1：过程 ARR(RM_{size}, λ, α)2 $RM=\varnothing$ $cw[j]=0$ $past_j=0\ \forall j$：，并且：从头随机初始化或使用预训练模型初始化（例如在 ImageNet 上训练的模型）3 $\bar{\Theta}$ 4：对于每段训练经历 e_i：5：$tw[j]=\begin{cases} cw[j], & j \quad \mathcal{D}^i \text{ 若类别} \\ 0 \end{cases}$ 属于，否则 6：$mb_e=\begin{cases} \frac{|\mathcal{D}^i|}{|\mathcal{D}^i|+RM_{size}}() & /mb_{size} \quad e_i>e_1 \text{，若，} \\ mb_{size} \end{cases}$ 否则 7：$mb_r=mb_{size}-mb_e$8：对于每个轮次：9：从 $\mathcal{D}^i$ 中采样 mb_e 个样本，并从 RM中采样 mb_r 个样本 10：在采样数据上训练模型（在第 α层注入回放数据）：11 $e_i=e_1$ Θ tw：若同时学习 和：否则学习 和，并使用 控制遗忘。12 tw $\bar{\Theta}$ λ 13：对于 $\mathcal{D}^i$中的每个类别 j：14：$wpast_j=\sqrt{\frac{past_j}{cur_j}}$，其中 cur_j是 $\mathcal{D}^i$中类别 j的样本数 15：$cw[j]=\frac{cw[j]\cdot wpast_j+(tw[j]-avg(tw))}{wpast_j+1}$ 16：$past_j=past_j+cur_j$ 17：使用 Θ和 cw测试模型 18：$h=\frac{RM_{size}}{i}$ 19：从 $\mathcal{D}^i$中随机采样 h 个样本 20：$R_{replace}=\begin{cases} \varnothing \text{ 若} \end{cases}$ $i==1$，则随机采样 h个样本；否则从 RM 中随机采样 21：$RM=(RM-R_{replace})\cup R_{add}$

在表征层并未完全冻结的一般情况下，存储在外部记忆中的激活值可能会受到老化效应 的影响（也就是说，随着时间推移，它们会逐渐偏离同一模式从输入层前向传播时产生的激活值）。然而，如果这些层的训练速度足够慢，老化效应就不会造成干扰，因为外部记忆有足够的时间用新获得的样本进行更新。使用小批量随机梯度下降进行潜在重放训练时：*(i)* 在前向步骤中，在重放层（沿小批量维度）执行拼接，将来自输入层的样本与来自外部存储的激活值合并；*(ii)* 对于重放样本，反向步骤会在重放层之前停止。

5 实证评估

为了通过实证评估 ARR 的整体质量和灵活性，我们评估了它在计算机视觉分类任务中三个常用持续学习基准测试上的表现：CIFAR-100（第 5.1 节）、CORe50（第 5.2 节）和 ImageNet-1000（第 5.3 节）。随后，我们深入分析潜在重放层选择（第 5.4 节）以及以记忆激活量衡量的内存大小（第 5.5 节）所产生的影响。

5.1 CIFAR-100

CIFAR-100 [18] 是一个广为人知且广泛使用的数据集，用于小尺寸（32×32）自然图像分类。它包含 100 个类别，每个类别有 600 张图像（500 张用于训练， + 100 张用于测试）。通过将 100 个类别划分为若干组，可以将默认分类基准转换为类别增量 场景（ [38] 将其称为 iCIFAR-100）。本文中，我们将类别分为每组 10 个，从而得到 10 个增量经验阶段。

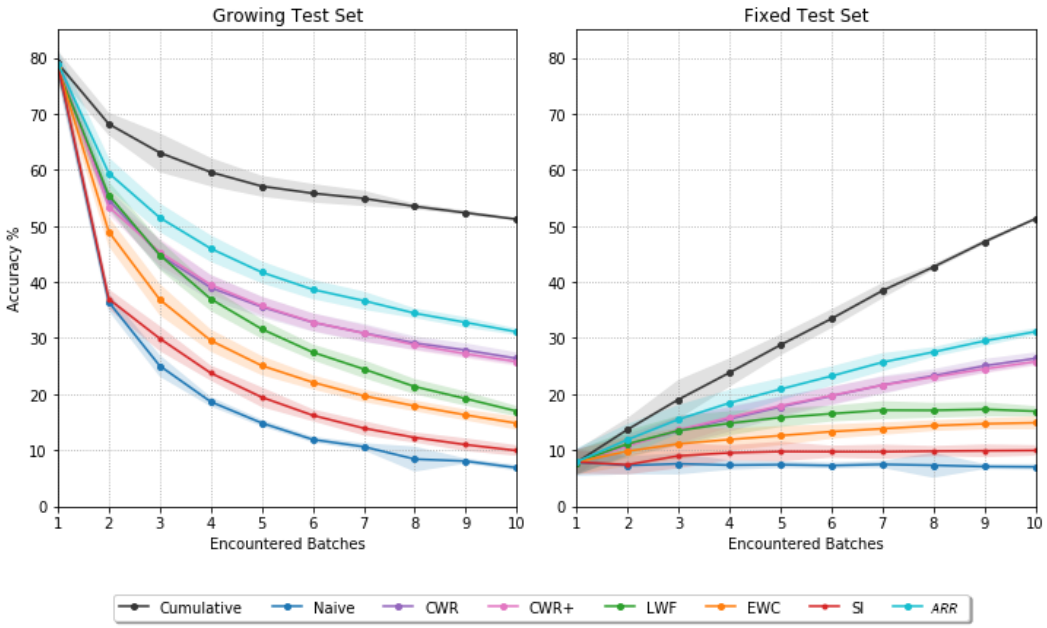

图 3： 在包含 10 个经验阶段（每个阶段 10 个类别）的 iCIFAR-100 上的准确率。结果为 10 次运行的平均值：所有策略的超参数均在第 1 次运行中调优，并在其余运行中保持不变。右侧实验遵循 CORe50 测试协议，采用包含全部 100 个类别的固定测试集；左侧则仅将迄今为止遇到的类别纳入测试集（与 [38] 中报告的结果类似）。彩色区域表示各条曲线的标准差。建议彩色查看。 [30]

本实验使用的卷积神经网络模型与 [44] 用于 CIFAR-10/100 Split 实验的模型相同 [30]。该模型由 4 个卷积层和全连接层组成；详情见 [44] 的附录 A。该模型在 CIFAR-10 [18] 上进行了预训练。图 3 比较了不同方法在 iCIFAR-100 上的准确率。结果表明：

- 与朴素方法不同，无遗忘学习 (LWF) [21] 和弹性权重巩固 (EWC)能够在一定程度上抵御遗忘，即使在这一增量场景中，它们的表现并不理想。单独使用突触智能（SI）时，其表现相当不稳定，且不如 LWF 和 EWC。
- CWR+ 相较于 CWR 的准确率提升在这里非常小，因为各经验阶段的数据是均衡的（因此不需要权重归一化），而且用于最后一层权重的卷积神经网络初始化值本来就非常接近 0（我们采用了作者的默认设置，即标准差为 $= 0.005$ 的高斯分布）。
- ARR (λ=$4.0e5$)始终优于所有其他方法。

值得注意的是，图 3 中报告的两组实验（即测试集不断扩大的实验（左）和测试集固定的实验（右），从左到右）在各方法的相对排名方面得出了相同结论。不过，我们认为，固定测试集更有助于观察增量学习趋势及其特点（饱和、遗忘等），因为分类复杂度（与类别数量成正比）在各经验阶段中保持不变。例如，从右图可以看出，SI、EWC 和 LWF 的学习能力在经历 6–7 个经验阶段后趋于饱和，而 CWR、CWR+ 和 ARR 则继续提升；左图中由于问题复杂度不断增加所造成的整体负面趋势，这一信息并不明显。

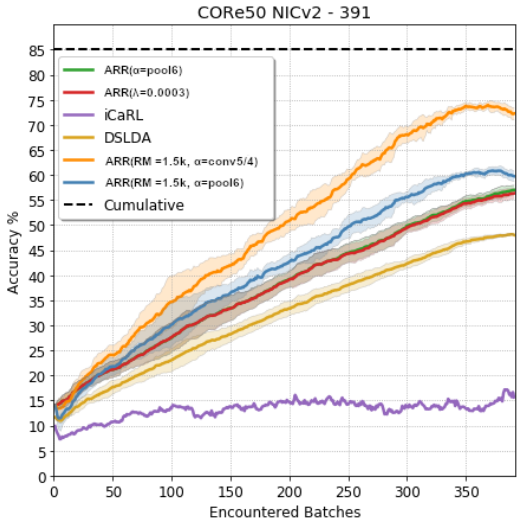

图 4： ARR(α=$pool6$)、ARR(λ=0.0003)、DSLDA、iCaRL、ARR(RM_{size}=1500, α=$conv5_4$) 和 ARR(RM_{size}=1500, α=$pool6$) 在 CORe50 NICv2 – 391 基准上的准确率结果。结果为 10 次运行的平均值，每次运行中经验阶段的顺序均经过随机打乱。彩色区域表示各曲线的标准差。例外的是，鉴于运行时间较长（~14 天），iCaRL 仅在单次运行中进行了训练。

最后请注意，iCIFAR-100 上的绝对准确率不能直接与 [38] 进行比较，因为 [38] 中使用的 CNN 模型是 ResNet-32，其准确率要高得多：在完整训练集上，本文使用的模型准确率约为 51%，而 ResNet-32 约为 68.1%。

5.2 CORe50

上一节已经讨论了所提出方法相较于最先进的无回放技术在准确率方面的提升；此外，将其与 CORe50 上其他最先进的持续学习技术进行比较，也有助于更好地认识其在真实世界的持续学习场景以及更长经验序列中的实际影响和优势。具体而言，虽然 ARR 和 ARR（α=$pool6$）已被证明在 NICv2 - 391 基准上显著优于 LWF 和 EWC [26], ，但仍值得将其与 iCaRL[39], ——最优秀的基于回放的技术之一——进行比较。

遗憾的是，iCaRL 是为类别增量 场景设计的，将其移植到带重复的类别增量 场景（其经验阶段也包含已知类别的样本）并非易事。为避免主观修改， [26] 的作者从 iCaRL 原作者分享的代码着手，并通过以下方式模拟带重复的类别增量 设置：*(i)* 始终根据新经验阶段中的样本创建新的虚拟类别；*(ii)* 评估准确率时合并虚拟类别。例如，假设在经验阶段 2 中遇到类别 5 的 300 个样本，并在经验阶段 7 中又遇到该类别的另外 300 个样本；虽然 iCaRL 在训练期间会创建两个虚拟类别，但在评估准确率时，这两个类别都对应真实类别 5。我们使用这种 iCaRL 实现，并为其配置包含 8000 个样本的外部记忆（远多于所提出的潜在重放所用的 1500 个样本，但符合原论文[38]提出的设置），在 NICv2 - 391 上进行了测试，但未能取得令人满意的结果。图 4 展示了 iCaRL 随时间变化的准确率，并将其与 ARR(RM_{size}=1500, α=$conv5_4/dw$) 、 ARR(RM_{size}=1500, α=$pool6$) 以及此前介绍的表现最佳的三种无回放策略进行比较：ARR(α=$pool6$)、ARR(λ=0.0003) 和 DSLDA。尽管 iCaRL 的表现优于 LWF 和 EWC（如 [25]所报告），但与 DSLDA、ARR(α=$pool6$) 和 ARR(λ=0.0003)相比仍相去甚远。

此外，当算法必须处理数量如此庞大的类别（包括虚拟类别）和训练体验时，其效率会变得很低（ [30] 中也报告了这一点）。 [26] 的表1报告了总运行时间（训练和测试）、内存开销，以及与累积上界相比的准确率差异。我们认为 ARR(RM_{size}=1500, α=$conv5_4/dw$) 体现了

方法	最终准确率
微调（朴素）	27.4
EWC-E [17]	28.4
RWalk [5]	24.9
LwM [9]	17.7
LWF [20]	19.8
iCaRL [39]	30.2
EEIL [3]	25.1
LUCIR [14]	20.1
IL2M [1]	29.7
BiC [43]	32.4
ARR [30]	**33.1**

表1： 按照 [32] 的基准，在 ImageNet-1000 上进行 25 个经验阶段、每个阶段包含 40 个类别时的最终准确率。每种方法均使用大小为 20,000 个样本的回放记忆库（训练结束时每个类别 20 个样本）。其他方法的结果引自 [32]。

在效率与效果之间的良好权衡：计算和内存开销有限，且与累积上界之间的准确率差距仅为 $\sim$13%。ForiCaRL 的总训练时间为 $\sim$14 天，而在单个图形处理器上，其他学习算法的训练时间均不到 $\sim$1 小时。

5.3 ImageNet-1000

为了进一步验证 ARR算法的可扩展性， [11] 的作者在 ImageNet-1000 这一具有竞争力的基准上进行了测试，并遵循 [32], 提出的类别增量 基准。该基准由 25 个经验阶段组成，每个阶段包含 40 个类别。由于类别数量庞大（1,000 个）、任务具有增量性质（包含 25 个经验阶段），且数据维度为 224×224 （与 ImageNet协议相同），因此该基准极具挑战性。

在这项实验中， [11] 将 ARR 与基于正则化的方法 [8, 17, 21] 以及基于回放的方法 [1, 3, 4, 14, 39, 43] 进行了比较。所有受测方法使用相同的分类器（ResNet-18）和相同的内存大小（20,000 个样本，每个类别 20 个）；对于基于正则化的方法，还将回放作为一种额外机制加入。

对于 ARR，他们使用随机梯度下降优化器训练模型。在第一个经验阶段，算法采用较大的学习率 0.1、动量 0.9 和权重衰减 10^{-4} 进行调优。随后，每经过 15 个训练轮次，初始学习率就乘以 0.1。模型总共训练 45 个轮次，批量大小为 128。对于之后的所有经验阶段，特征提取器的参数 ϕ 使用学习率为 $5 \cdot 10^{-3}$ 的随机梯度下降，分类器的参数 ψ 使用学习率为 $5 \cdot 10^{-2}$ 的随机梯度下降。每个经验阶段训练 32 个轮次，并采用学习率调度器，随着经验阶段数量增加而降低学习率。这样做是为了在旧知识比新知识更丰富时，保护旧知识不受新知识影响。与第一个经验阶段一样，批量大小设为 128，其中包含当前体验的 92 个样本，以及从回放记忆库中随机抽取（不放回）的 36 个样本。
结果如表1所示。基于回放的方法表现最佳，iCaRL 和 BiC 的最终准确率均超过 30%。ARR(RM_{size}=1500, α=$pool6$) 优于所有基线方法（33.1%），在这一具有挑战性的基准上取得了最先进的性能，证明了灵活的混合持续学习方法的优势。然而，考虑到 ResNet-18 在整个数据集上训练时的 ImageNet top-1准确率为 69.76%[3], , 即使是最佳方法，在持续学习设置下的准确率差距仍然非常大。这表明，持续学习，尤其是在类别数量众多且数据维度高的复杂场景中，远未得到解决，因此仍需投入更多研究。

[3]准确率取自 torchvision 官方页面：https://pytorch.org/vision/stable/models. html

5.4 重放层选择

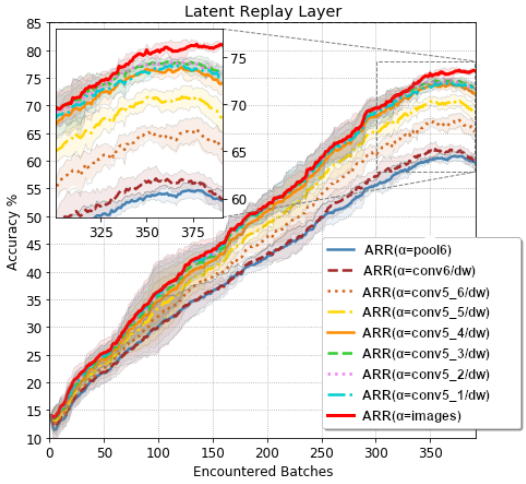

图5： 采用潜在重放（RM_{size}=1500）的 ARR 在选择不同潜在重放层时的表现。将重放层设为"图像"层，相当于原生复习。最后几个训练经验阶段呈现的饱和效应源于 NICv2 – 391 中的数据分布（见 [25]）：具体而言，某些类别缺少新实例（这些类别的数据已全部引入），这减缓了准确率的变化趋势，并加剧了激活老化效应。

图5展示了在 CORe50 实验中，ARR(RM_{size}=1500, α=...) 采用不同回放层 α 时的准确率。不出所料，回放层位置越靠后，相应的准确率越高，这证明持续调整表征层十分重要。不过，在 conv5_4/dw 之后，准确率开始趋于饱和，模型准确率不再提升。与原生复习相比，残余差距（~4%）并非由网络较低部分的权重冻结造成，而是由上文所述的老化效应引起。只需采用一种"中间"方法即可证明这一点：始终从输入端馈入回放模式，并在 conv5_4处停止反向传播；这种中间方法在训练结束时取得的准确率与原生复习（使用原始数据）非常接近。我们认为，通过更好地调整批量重归一化（BRN）超参数，和/或引入一种调度策略，使全局矩移动窗口随着持续学习的推进而逐渐变宽（即早期具有更强的可塑性，后期则更加稳定），可以进一步减小老化效应导致的准确率下降；不过，这种精细优化取决于具体应用，超出了本研究的范围。

为更好地评估潜在重放相对于原生复习的表现，我们在表2中报告了相关维度：*(i)* 计算量指部分前向传播（从潜在重放层开始）相对于从输入层开始的完整前向步骤，其运算量所占的百分比；*(ii)* 模式大小指需存储在外部记忆中的模式维度（考虑到我们使用 MobileNetV1，并采用 128×128×3 输入以匹配 CORe50 图像大小）；*(iii)* 准确率和Δ 准确率分别表示训练结束时的绝对准确率，以及与原生复习相比的差距。例如，conv5_4/dw 呈现出有意思的权衡：计算量约为原生复习的 32%，存储量降至 66%（关于这一点的更多内容见第 5.5 节），且准确率下降幅度较小（5.07%）。ARR(RM_{size}=1500, α=$pool6$) 相较于原生复习，计算开销可忽略不计（0.027%），同时相较于非复习情形，准确率仍提升了 ~4%（分别为 ~60% 和 ~56%，如图5和图6所示）。

5.5 回放记忆大小选择

为了了解外部记忆大小的影响，我们使用不同的RM_{size} 值重复了实验：500、1,000、1,500、3,000。结果如图6所示：值得注意的是，增加复习记忆可以提高所有算法的准确率，但1,500与3,000之间的差距并不大。我们认为，对于这个数据集，1,500是一个不错的折中选择。ARR(RM_{size}=

表2： 在 NICv2 – 391 上使用 RM_{size}=1500 持续训练的 MobileNetV1 卷积网络中，不同层采用潜在重放时的计算量、存储和准确率权衡。

层	计算量 % 相对于原生复习	样本大小	最终准确率 %	Δ准确率 % 与原生复习相比
图像	100.00%	49152	77.30%	0.00%
conv5_1/dw	59.261%	32768	72.82%	-4.49%
conv5_2/dw	50.101%	32768	73.21%	-4.10%
conv5_3/dw	40.941%	32768	73.22%	-4.09%
conv5_4/dw	**31.781%**	**32768**	**72.24%**	**-5.07%**
conv5_5/dw	22.621%	32768	68.59%	-8.71%
conv5_6/dw	13.592%	8192	65.24%	-12.06%
conv6/dw	9.012%	16384	59.89%	-17.42%
pool6	0.027%	1024	59.76%	-17.55%

...) 在提供足够数量的回放样本时，效果略优于 ARR(RM_{size}=...,λ=0.003)；但正如预期的那样，回放较少（即 RM_{size}=500）或不进行回放时，准确率会更低。

值得注意的是，图6中表现最佳的 ARR 配置，即 ARR(RM_{size}=3000)，仅比累积上界差 5%；更好的参数化方式以及对复习记忆的更充分利用，可以进一步缩小这一差距。

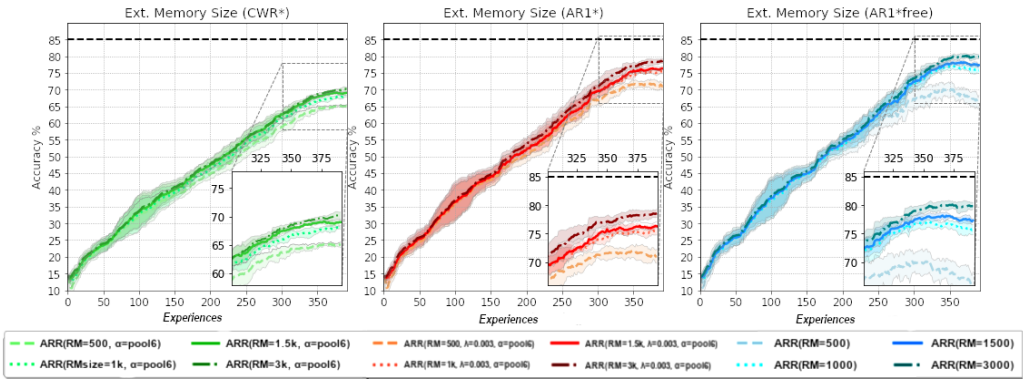

图6： 在 CORe50 NICv2 – 391 上，对比采用不同外部记忆大小（RM_{size}=500，1000，1500 和 3000 个样本）的主要 ARR 配置。

6 Avalanche 中的 ARR 实现

本文提出的 架构化、正则化与回放（ARR）方法，是对过去几年提出的不同变体及改进进行全面重新形式化的成果，始于 [24, 30]。这类方法（CWR、CWR+、CWR*、AR1、AR1* 以及采用潜在表征重放的 AR1*）的原始实现均已在 Caffe 和 PyTorch 中提供。然而，鉴于这些方法各不相同，要从一种实现迁移到另一种实现，并将其应用于与提出时所针对的设定和场景稍有不同的情况，都相当困难。

为了充分发挥 ARR 方法的通用性和灵活性，我们决定直接在 Avalanche [27] 中重新实现该方法。*Avalanche* 是一个基于 PyTorch 的持续学习开源（采用麻省理工学院许可证）端到端库，旨在提供共享、协作的代码库，以便快速进行持续学习算法的原型开发、训练和评估。

得益于 Avalanche 的可移植实现（即将集成到该库的下一个稳定版本中），ARR 可配置为复现本文所介绍的实验（图 7），遵循先前提出的策略（如 AR1*、CWR* 等），并且已准备好接受

```
strategy = ARR(model, optimizer, criterion, mem_size, lambd, alpha)

for experience in benchmark.train_stream:

    strategy.train(experience)
    strategy.eval(benchmark.test_stream)
```

图 7： Avalanche 中的 ARR 实现。给定一组超参数，即可实例化并正确配置 ARR，以便在 Avalanche 中已有的大量基准测试上进行测试。

测试，可使用 Avalanche 中已有的大量基准测试，也可轻松将其添加到该库中。

7 结论

本文展示了 ARR 是一种灵活、有效且高效的技术，即使面对规模较小且非独立同分布的经验阶段，也能持续学习新类别以及已知类别的新实例。采用潜在重放的 ARR 确实能够高效学习，同时取得的准确率也与累积上界相差不远（某些情况下约为 5%）。计算量、存储和准确率之间的权衡可根据目标应用和可用资源进行设定，因此，即使是没有图形处理器的边缘设备也能持续学习。此外，ARR 可以轻松扩展，以支持更复杂的重放记忆管理策略（也可用于抵消老化效应），甚至可以与在训练过程中一同训练的生成模型结合；该模型能够按需提供伪激活体积，正如 [11] 中最初展示的那样。

参考文献

参考文献

[1] Eden Belouadah 和 Adrian Popescu。IL2M：利用双重记忆进行类别增量学习。载于*IEEE/CVF*国际计算机视觉大会论文集，第 583–592 页，2019年。[2] Antonio Carta、Andrea Cossu、Vincenzo Lomonaco 和 Davide Bacciu。Ex-model：从训练模型流中进行持续学习。*arXiv* 预印本 *arXiv:2112.06511*，2021年。[3] Francisco M Castro、Manuel J Marín-Jiménez、Nicolás Guil、Cordelia Schmid 和 Karteek Alahari。端到端增量学习。载于欧洲计算机视觉大会（*ECCV*）论文集，第 233–248 页，2018年。[4] Arslan Chaudhry、Puneet K Dokania、Thalaiyasingam Ajanthan 和 Philip HS Torr。用于增量学习的黎曼游走：理解遗忘与固执性。载于欧洲计算机视觉大会（*ECCV*）论文集，第 532–547 页，2018年。[5] Arslan Chaudhry、Puneet K Dokania、Thalaiyasingam Ajanthan 和 Philip HS Torr。用于增量学习的黎曼游走：理解遗忘与固执性。载于*ECCV*，2018年。[6] Andrea Cossu、Gabriele Graffieti、Lorenzo Pellegrini、Davide Maltoni、Davide Bacciu、Antonio Carta 和 Vincenzo Lomonaco。类别增量是否足以实现持续学习？《人工智能前沿》，第 5 卷，2022年。

[7] Andrea Cossu、Tinne Tuytelaars、Antonio Carta、Lucia Passaro、Vincenzo Lomonaco 和 Davide Bacciu。持续预训练可减轻语言与视觉领域中的遗忘。*arXiv* 预印本 *arXiv:2205.09357*，2022年。[8] Prithviraj Dhar、Rajat Vikram Singh、Kuan-Chuan Peng、Ziyan Wu 和 Rama Chellappa。无记忆学习。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第 5138–5146 页，2019年。[9] Prithviraj Dhar、Rajat Vikram Singh、Kuan-Chuan Peng、Ziyan Wu 和 Rama Chellappa。无记忆学习。载于*IEEE*计算机视觉与模式识别会议（*CVPR*），2019年6月。[10] Natalia Díaz-Rodríguez、Vincenzo Lomonaco、David Filliat 和 Davide Maltoni。别忘了，遗忘之外还有更多：持续学习的新指标。载于持续学习研讨会，*NeurIPS 2018*（神经信息处理系统，蒙特利尔，加拿大，2018年12月。[11] Gabriele Graffieti、Davide Maltoni、Lorenzo Pellegrini 和 Vincenzo Lomonaco。用于持续学习的生成式负重放。*arXiv* 预印本 *arXiv:2204.05842*，2022年。[12] T. L. Hayes、R. Kemker、N. D. Cahill 和 C. Kanan。持续学习的新指标与实验范式。载于*2018*年 *IEEE/CVF*计算机视觉与模式识别研讨会（*CVPRW*），第 2112–21123 页，2018年6月。[13] Tyler L. Hayes、Nathan D. Cahill 和 Christopher Kanan。用于流式学习的内存高效经验重放。*2019*年国际机器人与自动化会议（*ICRA*），第 9769–9776 页，2018年。[14] Saihui Hou、Xinyu Pan、Chen Change Loy、Zilei Wang 和 Dahua Lin。通过再平衡逐步学习统一分类器。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第 831–839 页，2019年。[15] Ronald Kemker 和 Christopher Kanan。Fearnet：受大脑启发的增量学习模型。载于国际学习表征会议，2018年。[16] Ronald Kemker、Marc McClure、Angelina Abitino、Tyler L. Hayes 和 Christopher Kanan。衡量神经网络中的灾难性遗忘。载于*AAAI*，2017年。[17] James Kirkpatrick、Razvan Pascanu、Neil Rabinowitz、Joel Veness、Guillaume Desjardins、Andrei A Rusu、Kieran Milan、John Quan、Tiago Ramalho、Agnieszka Grabska-Barwinska 等。克服神经网络中的灾难性遗忘。《美国国家科学院院刊》，2017年。[18] Alex Krizhevsky、Geoffrey Hinton 等。从微小图像中学习多层特征。技术报告，Citeseer，2009年。[19] Timothée Lesort、Vincenzo Lomonaco、Andrei Stoian、Davide Maltoni、David Filliat 和 Natalia Díaz-Rodríguez。机器人持续学习：定义、框架、学习策略、机遇与挑战。信息融合，2019年12月。[20] Xilai Li、Yingbo Zhou、Tianfu Wu、Richard Socher 和 Caiming Xiong。学会增长：一种克服灾难性遗忘的持续结构学习框架。*CoRR*，abs/1904.00310，2019年。[21] Zhizhong Li 和 Derek Hoiem。无遗忘学习。*IEEE*模式分析与机器智能汇刊，2017年。[22] Vincenzo Lomonaco。深度架构下的持续学习。博士论文，博洛尼亚大学，2019年。[23] Vincenzo Lomonaco、Karan Desai、Eugenio Culurciello 和 Davide Maltoni。在三维非平稳环境中的持续强化学习。*arXiv* 预印本 *arXiv:1905.10112*，2019年。

[24] Vincenzo Lomonaco 和 Davide Maltoni。CORe50：用于连续目标识别的新数据集与基准。收录于 Sergey Levine、Vincent Vanhoucke 和 Ken Goldberg 编，第一届机器人学习年会论文集，机器学习研究论文集第78卷，17–26页。PMLR，2017年11月13日至15日。[25] Vincenzo Lomonaco、Davide Maltoni 和 Lorenzo Pellegrini。细粒度持续学习。*arXiv*预印本 *arXiv: 1907.03799*，1–14页，2019年。[26] Vincenzo Lomonaco、Davide Maltoni 和 Lorenzo Pellegrini。针对小型非独立同分布批次的无复习持续学习。收录于*CVPR*研讨会，第1卷，第3页，2020年。[27] Vincenzo Lomonaco、Lorenzo Pellegrini、Andrea Cossu、Antonio Carta、Gabriele Graffieti、Tyler L Hayes、Matthias De Lange、Marc Masana、Jary Pomponi、Gido M Van de Ven 等。Avalanche：用于持续学习的端到端库。收录于*IEEE/CVF*计算机视觉与模式识别会议，3600–3610页，2021年。[28] Vincenzo Lomonaco、Lorenzo Pellegrini、Pau Rodriguez、Massimo Caccia、Qi She、Yu Chen、Quentin Jodelet、Ruiping Wang、Zheda Mai、David Vazquez 等。CVPR 2020计算机视觉持续学习竞赛：方法、结果、当前挑战与未来方向。人工智能，303：103635，2022年。[29] David Lopez-Paz 和 Marc-Aurelio Ranzato。用于持续学习的梯度情景记忆。收录于 I. Guyon、U. V. Luxburg、S. Bengio、H. Wallach、R. Fergus、S. Vishwanathan 和 R. Garnett 编，神经信息处理系统进展第*30*卷，6467–6476页。Curran Associates, Inc.，2017年。[30] Davide Maltoni 和 Vincenzo Lomonaco。单增量任务场景中的连续学习。神经网络，116：56–73，2019年8月。[31] Davide Maltoni 和 Vincenzo Lomonaco。单增量任务场景中的连续学习。神经网络，116：56–73，2019年。[32] Marc Masana、Xialei Liu、Bartlomiej Twardowski、Mikel Menta、Andrew D Bagdanov 和 Joost van de Weijer。类别增量学习：图像分类综述与性能评估。*arXiv*预印本 *arXiv:2010.15277*，2020年。[33] Gabriele Merlin、Vincenzo Lomonaco、Andrea Cossu、Antonio Carta 和 Davide Bacciu。基于回放的持续学习方法的实用建议。*arXiv*预印本 *arXiv:2203.10317*，2022年。[34] German I Parisi 和 Vincenzo Lomonaco。序列上的在线持续学习。收录于数据学习最新趋势，197–221页。Springer，2020年。[35] German I. Parisi、Jun Tani、Cornelius Weber 和 Stefan Wermter。通过双记忆循环自组织终身学习时空表征。*Frontiers inNeurorobotics*，12：78，2018年。[36] Lorenzo Pellegrini、Gabrile Graffieti、Vincenzo Lomonaco 和 Davide Maltoni。用于实时持续学习的潜在重放。*arXiv*预印本 *arXiv:1912.01100*，2019年。[37] Leonardo Ravaglia、Manuele Rusci、Alessandro Capotondi、Francesco Conti、Lorenzo Pellegrini、Vincenzo Lomonaco、Davide Maltoni 和 Luca Benini。RISC-V极边缘节点上持续学习的内存—延迟—精度权衡。收录于*2020*年*IEEE*信号处理系统研讨会（*SiPS*），1–6页。IEEE，2020年。[38] S. Rebuffi、A. Kolesnikov、G. Sperl 和 C. H. Lampert。iCaRL：增量分类器与表征学习。收录于*2017*年*IEEE*计算机视觉与模式识别会议（*CVPR*），5533–5542页，2017年7月。[39] Sylvestre-alvise Rebuffi、Alexander Kolesnikov、Georg Sperl 和 Christoph H Lampert。iCaRL：增量分类器与表征学习。收录于*IEEE*计算机视觉与模式识别会议（*CVPR*），檀香山，夏威夷，2017年。

[40] A. A. Rusu、N. C. Rabinowitz、G. Desjardins、H. Soyer、J. Kirkpatrick、K. Kavukcuoglu、R. Pascanu 和 R. Hadsell。渐进式神经网络。*ArXiv*电子预印本，2016年6月。[41] Hanul Shin、Jung Kwon Lee、Jaehong Kim 和 Jiwon Kim。通过深度生成式回放实现持续学习。收录于神经信息处理系统进展，2990–2999页，2017年。[42] Chenshen Wu、Luis Herranz、Xialei Liu、yaxing wang、Joost van de Weijer 和 Bogdan Raducanu。记忆回放生成对抗网络：学习生成新类别而不遗忘。收录于 S. Bengio、H. Wallach、H. Larochelle、K. Grauman、N. Cesa-Bianchi 和 R. Garnett 编，神经信息处理系统进展第*31*卷，5962–5972页。Curran Associates, Inc.，2018年。[43] Yue Wu、Yinpeng Chen、Lijuan Wang、Yuancheng Ye、Zicheng Liu、Yandong Guo 和 Yun Fu。大规模增量学习。收录于*IEEE/CVF*计算机视觉与模式识别会议，374–382页，2019年。[44] Friedeman Zenke、Ben Poole 和 Surya Ganguli。通过突触智能实现持续学习。收录于 Doina Precup 和 Yee Whye Teh 编，第*34*届国际机器学习会议论文集，机器学习研究论文集第70卷，3987–3995页，悉尼国际会议中心，澳大利亚，2017年8月6日至11日。PMLR。