## 摘要

目前，持续学习的基准通常使用一种非常特殊的经验流，即每种经验只出现一次，且经验之间没有重叠。它们并不能反映现实世界中可以观察到的任意的非稳态性。在这项CVPR举办的比赛中，它们的目标是为一类持续学习问题设计高效策略;"带重复的类递增问题（CIR）"。CIR 包括各种流，有两个关键特征：(i) 以前观察到的类可以在新的体验中以任意重复模式再次出现；(ii) 并非所有类都必须出现在每次体验中。由于许多现有策略都是针对无重复的持续学习问题开发的，因此尚不清楚它们在 CIR 流中的表现和比较。为了探索重复的意义及其与开发新策略的相关性，它们使用了比赛方提供的一组由流生成器创建的 CIR 基准，该流生成器由三个参数控制，并有明确的解释。它们作为参赛者开发的策略在模型完成对整个数据流的训练后，能在测试集上达到较高的平均准确率，测试集包含数据流中所有类别的同等数量的未见示例。

## 简介

持续学习涉及一个具有挑战性的问题，即在非稳定的经验流中训练一个模型。目前，持续学习的基准通常使用一种非常特殊的经验流，即每种经验只出现一次，且经验之间没有重叠。在这种情况下，经验通常被称为 "任务"。尽管这些基准已被证明对学术研究有用，但它们并不能反映现实世界中可以观察到的任意的非稳态性。例如，这些基准不包含任何重复。

给定一个包含多个类别的静态数据集，本挑战赛使用的生成器通过四个可解释的控制参数创建随机数据流：

1.数据流长度 (N)：数据流中经验的数量。

2.经验大小 (S)：每种经验中的模式数量。默认情况下，经验大小平均分配给每个经验中的现有类别。

3.首次出现分布 (Pf)：数据流中经验的离散概率分布，决定了数据集类别在整个数据流中首次出现的概率。

4.重复概率（Pr）：每个类别的重复概率，用于控制每个类别在数据流中首次出现后再次出现的可能性。最简单的形式是一个概率值列表，每个类一个概率值。

本比赛有5中随机数据流，他们的上述参数互有差异。

## 现状

官方代码使用两个插件，分别是EWC和LwF插件。前者使⽤Fisher矩阵来评估⽹络中参数的重要性，通过正则化来更新任务。后者使⽤知识蒸馏来保留旧的知识，同时为新类别增加新输出头。天津大学提出了一种与集合学习策略相结合的参数隔离方法。具体来说，它们对分类器进行了调整，以保持模型在不同训练阶段获得的能量水平相同。为了解决样本效率问题，它们使用自监督学习来捕捉更具一般性和鉴别性的表征，从而提高泛化性能。此外，它们还建立了一个共享提示池，以促进不同任务和类别之间的互动，从而促进知识融合。格拉茨技术大学的核心思想是在选定的经验上学习一组特征提取器（Feature Extractors，FE），这些特征提取器应能提供有助于区分下游类别的强健特征。它们有一些启发式方法，可以在每次经验中决定是否要为当前类别学习新的特征提取器。它们不考虑少于 5 个类别的体验，在它们看到 85% 的类别后停止向集合中添加 FE，并始终在第一次体验中训练一个 FE。它们同时使用交叉熵头和对比损失来学习它们，对比损失在单独的头中强调硬负对。这两种损失都使用自适应阿尔法来平衡，阿尔法会根据每种损失的能量自动计算。从当前的经验中吸取教训后，头部会被移除，骨干会被冻结并添加到集合的其他部分。不过，由于没有任何一个 FE 对彼此有了解，它们需要使用伪特征投影来调整它们的表征，以便进行统一分类。因此，无论训练了哪一个 FE，它们都会更新在所有集合表征上训练的统一头。为了平衡统一表头，它们从 FeTrIL 中汲取灵感，扩展了伪特征投影，使其也能用标准偏差和所有集合的输出进行估计。当某些类别的平均值和标准偏差不可用时，它们会通过重新使用当前的经验表征来替代。

南京航空航天大学提出的方法是一种基于门控网络的动态架构。它们为每种体验设置独立的分支，并通过门控单元控制当前分支是否被激活。在训练过程中，门控单元会打开当前体验的分支，让网络进行学习，同时关闭之前体验的分支，冻结其参数。为了在训练过程中提高模型的泛化能力和鲁棒性，它们使用了大量数据增强技术，如增强混合（augmix）。在测试过程中，门控单元控制各分支执行顺序预测。不过，需要注意的是，像 DER 那样同时考虑所有分支的预测可能不会产生好的结果，因为大多数分支可能没有见过这个类别，可能会做出过于自信的判断。这个问题可以看作是一个开放集识别问题。为了解决这个问题，它们提出了一种基于熵、特征规范和类别数量的加权策略。具体来说，对于每个分支，它们都要计算预测概率的熵。如果熵值较高，则样本很可能是当前分支的开放集样本。同样，它们还会计算特征规范。如果特征规范高，样本也很可能是一个开放集样本。最后，它们认为经验的类别越多，模型的判断就越可靠。因此，它们也会根据经验的类数进行加权。

## 方法

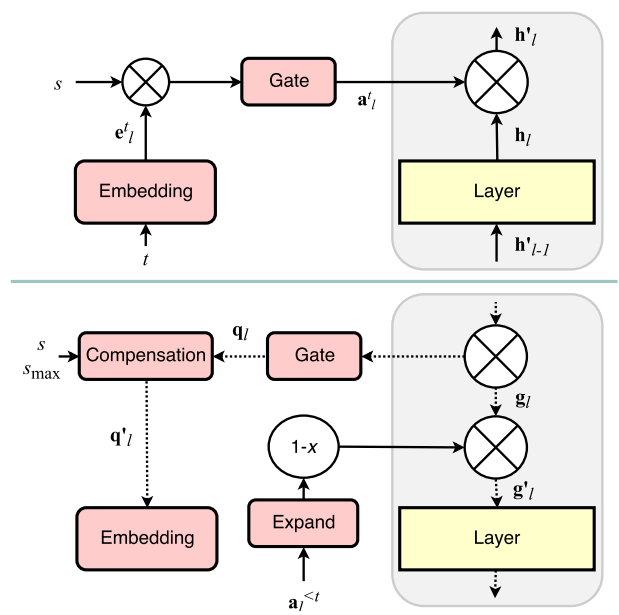
为了应对当前任务 t，我们采用了一种分层注意机制（Hard Attention to the Task, HAT[1]）（图 1）。给定第 l 层单元的输出 hl，我们按元素顺序乘以 文本

描述已自动生成。然而，与普通注意力机制的一个重要区别在于，相乘因子并非形成概率分布，而是单层任务嵌入图片包含 文本

描述已自动生成的门控版本。

图片包含 文本

描述已自动生成



其中，σ(x) ∈ [0, 1] 是一个门控函数，s 是一个正的缩放参数。实验中使用了 sigmoid ，但也可以使用其他机制。所有层 l = 1。 除了最后一层 L，其他层的操作是相同的。其中 L 是二进制硬编码。第 L 层的操作等同于多头输出。这在灾难性遗忘中是经常使用的。

公式 1 的门控机制背后的理念是形成硬的、可能是二元的注意力掩码，这些掩码就像 "抑制性突触 "一样，可以激活或停用每一层单元的输出。通过这种方式，与 PathNet（费尔南多等人，2017 年）类似，我们可以动态地创建和破坏跨层路径，这些路径可以在以后学习新任务时保留下来。不过，与 PathNet 不同的是，HAT 中的路径不是基于模块，而是基于单个单元。因此，我们不需要预先分配模块大小，也不需要为每个任务设置模块的最大数量。给定一定的网络架构后，HAT 会学习并自动调整单个单元的路径，最终影响单个层的权重。此外，HAT 不使用遗传算法在单独阶段学习路径，而是使用反向传播和 SGD 与网络的其他部分一起学习路径。

同时，我们也是用基于碎片与集成的重复技术。前者是每个碎⽚针对不同体验处理⼀个体验流。后者是对于同⼀个体验，有多个模型，在预测时，将他们的预测结果平均。

图示

描述已自动生成

最后，我们也尝试了使用Replay保存部分数据。虽然这违反了官方比赛的规定，但是我们的实验结果证明了它的相对于基准的有效性。

## 实验和总结

本次实验由一台RTX4090与RTX3090完成。由于时间与设备限制，我们主要在前两个随机数据流中进行对照试验。下表中的换插件是在官方示例代码上更换插件，效果并不明显。使用Replay保存部分数据相比前者有较大提升。不带任何重复操作的HAT平均正确率大约50%，而基于重复的HAT高达56%左右。



## 作者贡献

1. 阅读HAT文献，理解期工作机制。
2. 撰写小组汇报ppt。
3. 在官方示例基础上尝试不同插件，并运行改进版HAT代码，计算出结果。

## 参考文献

[1] Joan Serrà, Dídac Surís, Marius Miron, Alexandros Karatzoglou, Overcoming catastrophic forgetting with hard attention to the task. ICML 2018.