《人工智能导论》探究报告
连续学习

姓名：王昱　　学号:2212046　　专业：信息安全
姓名：闫耀方　　学号:2212045　　专业：信息安全
姓名：赵熙　　学号:2210917　　专业：计算机科学与技术

作业正文

1 问题描述

1.1 概述

近年来深度学习在许多单项任务上取得了显著成绩，但这些模型通常是为固定任务设计的，缺乏动态适应能力。这些模型在处理单一任务时表现优异，但在面对不断变化的数据分布时，通常会遇到灾难性遗忘（catastrophic forgetting）的问题。当新的数据分布出现时，这些模型需要在整个历史数据上重新训练，然而在现实环境中，这种规模的训练是不切实际的。这种背景下，连续学习作为解决方案，逐渐受到研究者的关注。

对于生命智能体，知识技能的学习是一个终身的过程，虽然存在对以往知识的遗忘行为，但学习新知识时，对旧知识几乎不会产生灾难性遗忘。对于深度学习系统，连续学习使模型能够在一个连续序列的不同任务上学习，而不会遗忘已学知识。新任务通常与旧任务相关联，连续学习要求模型在学习新任务的同时，能够有效地维持在历史任务上的表现。这种能力对于实现真正的人工智能至关重要，因为它允许模型在动态变化的现实环境中快速适应。连续学习的挑战主要在于“稳定性-可塑性境”（stability-plasticity dilemma），即模型在学习新知识时保持旧知识的能力与模型融入新知识的能力之间的平衡。

1.2 概念比较

- **连续学习**：重点在迁移，即在任务一学习到的知识可以很好地适用于任务二上，关注任务间传递学习的效果，不重视对原任务一的表现。连续学习同样重视新任务和旧任务的学习效果。
- **多任务学习**：即把多个相关的任务放在一起学习，例如短视频推荐系统，观看时长、点赞、评论、转发情况来综合判断该视频是否符合观众的喜好。多任务学习是在同一时间上获取所有任务。
- **在线学习**：连续学习过程中任务是一个个出现的。重点在在线，虽然在线学习的过程中任务也是序列进行的，但重点是其高效性、强鲁棒性。通常需要每个样本只能使用一次，且数据都是来自于同一个任务。连续学习是终身的，目的是持续积累多个任务的推理知识下保存下来，会被多次地使用，类似人脑受到的训练。

许多研究常常将连续学习、终身学习和增量学习混为一谈，但 Masana M 等

人对这三个概念进行了明确区分。终身学习应被看作一个完整的系统，目标是建立一个能够在整个生命周期内持续学习的智能系统，而连续学习则是终身学习的一个方面。连续学习和增量学习的界限比较模糊，它们的主要目标都是在任务逐步增加时解决性能下降的问题，追求稳定性与可塑性的平衡。不同之处在于，增量学习更强调在训练过程中保留部分已有的经验，而不是每次都重新开始训练。

1.3 连续学习的情景

1. **任务增量学习** (Task-IL)：在任务增量学习中，模型在评估时能够知道当前输入来自哪个任务。Task-IL 是最早的连续学习情景之一，其研究重点是通过多头结构（multi-headed architecture）处理不同任务。每个任务有独立的输出层，其余的网络结构不随任务改变。

2. **类别增量学习** (Class-IL)：随着研究的深入，类别增量学习（Class-IL）成为一个重要方向。在 Class-IL 情景下，模型需要既能够完成给定任务，同时也要判别出任务的具体类别。iCaRL 方法即是在这种背景下提出的，通过存储旧任务的示例，并在新任务学习过程中重放这些示例，有效减少了灾难性遗忘的问题。

3. **领域增量学习（**Domain-IL)：领域增量学习（Domain-IL）是另一重要情景，模型无法在此情景下得知任务的类别，同时也不需要判断当前任务所属的具体类别。此情景下的任务都具有相同的结构，但有着变化的输入分布。例如，一个目标为学习在不同环境下生存的模型，每个环境下的结果都是生存和死亡两种，而模型不需要知道自己面对的具体环境。

1.4 连续学习的方法

积力之所举，则无不胜也；众智之所为，则无不成也。在研究者的共同努力下，目前连续学习已经取得了很好的成果，主要分为三类：

1. **基于重放的方法**：通过重放保存的训练数据来减轻遗忘问题。典型的重放方法包括 iCaRL 和 GEM。iCaRL 方法通过在学习新任务的同时重新训练保存的历史数据子集，有效减少了灾难性遗忘的问题。GEM 方法通过约束优化，限制新任务对旧任务的负面影响。

2. **基于正则化的方法**：通过在损失函数中添加正则项来保留模型对之前任务的知识。例如，LwF 通过计算蒸馏损失保留旧任务知识，EWC 使用 Fisher 信息矩阵估计参数重要性以减小重要参数的更新幅度。

3 **基于参数隔离的方法**：假设模型参数的不同子集在不同任务上发挥主要作用，通过冻结部分参数并更新其他参数来防止遗忘。PackNet 方法即是其中的代

表，通过在学习新任务后冻结部分参数，以保留旧知识。尽管这类方法有效防止了灾难性遗忘，但随着任务数量的增加，模型参数会迅速膨胀，导致计算资源的需求增加。

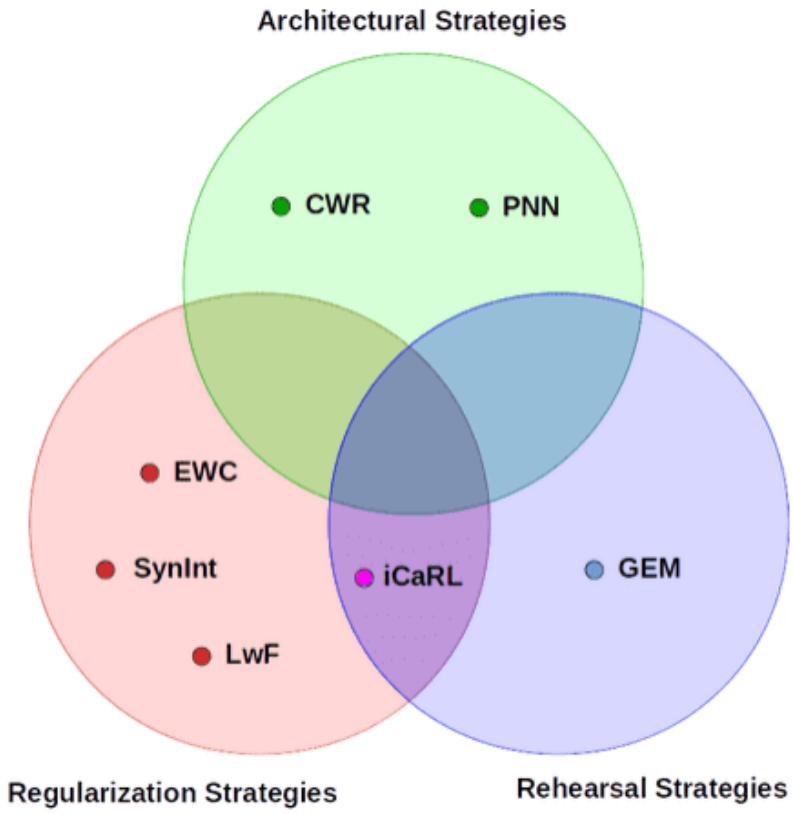


图 1: 不同类连续学习方法的 Venn 图

2 核心内容：

2.1 Learning Without Memory[1]

2.1.1 论文动机

在计算机视觉领域，增量学习旨在通过逐步增加模型可以识别的类别数量来增强模型的能力，即模型可识别的类的数量。这项任务的关键问题是需要存储与现有类别相关的数据，同时指导分类器学习新的类别。然而，在内存有限的边缘设备上实施是不切实际的，因为它在每一个递增的步骤中都会增加存储需求。同时，增量学习的另一个关键挑战在于如何在不遗忘已学知识的情况下学习新知识，当前的方法往往忽略了旧数据对模型的重要性，容易导致“灾难性遗忘”。

灾难性遗忘（catastrophic forgetting），即模型在学习新类别时完全忘记已学类别。这一现象在传统增量学习方法中非常普遍，因为新数据会覆盖旧数据的记忆。LwM 方法旨在解决这一问题，通过约束注意力图来保持模型对已学类别的记忆，从而减少灾难性遗忘的发生。因此，作者认为的理想增量模型具有以下特性：

1. 模型在所有新旧类别的分类任务中都应保持良好的性能。
2. 模型能够从不断更新的数据中学习新类别，同时保存其从原始数据中获得的知识。
3. 随着新类别和新样本的增加，模型的内存消耗应保持稳定。

本文的动机在于通过一种新的方法解决这一问题，使模型在不存储旧数据的情况下，仍能有效保留旧知识，并逐步学习新知识。由此，作者提出了一种新的方法，称为“无记忆学习”，也就是 LwM 方法。LwM 方法通过引入注意力蒸馏损失（Attention Distillation Loss, L_{AD}），在不存储任何已学类别数据的情况下保留这些类别的信息，使分类器能够逐步学习新类别。具体来说，LwM 方法通过约束学生模型和教师模型的注意力图，使学生模型在学习新类别时能够保持对已学类别的识别能力。

2.1.2 相关工作

在目标分类领域，增量学习（Incremental Learning, IL）是一个关键任务，其目标是通过逐步增加模型可以识别的类别数量，增强模型的能力。现有的方法可以根据其处理方式和所施加的约束条件进行分类：

- **任务增量（Task Incremental, TI）方法**：
  任务增量方法是指模型在一个特定的数据集上进行初步训练，然后逐步在新的数据集上进行训练。这类方法通常采用多头评估，在测试时分别评估模型在不同数据集上的表现，以避免类别混淆。这类方法包括使用蒸馏损失来保留基本类别的信息，通过网络权重的重要性调整模型权重，以及训练模型学会哪些信息不应该被遗忘。
- **类别增量（Class Incremental, CI）方法**：
  类别增量方法是指模型在特定数据集的特定类别上进行初步训练，然后逐步增加新的未见过的类别。这类方法通常采用单头评估，即在测试时同时评估模型在所有类别上的表现。相比于任务增量方法，类别增量方法更具挑战性，因为模型可能会将新类别与基本类别混淆。这类方法通过联合学习特征表示和分类器，并引入选择示例的方法，与蒸馏损失结合使用，以防止灾难性遗忘。

增量学习方法的一个关键因素是是否允许存储基本类别的数据。基于这一因素，现有方法可以分为两类：

- **使用基本类别数据的方法**：这些方法在训练模型学习新类别时，使用一部分基本类别的数据来缓解遗忘问题，但这会增加每个增量步骤的内存需求。
- **不使用基本类别数据的方法**：这些方法在训练模型逐步学习新类别时，不使用基本类别的数据，通过单头评估使用蒸馏损失来保留基本类别的知识。

Constraints	Use base class data	No base class data
CI methods	iCaRL	LwF-MC, LwM
TI methods	LwF	IMM, EWC, MAS

表 1: 对增量学习中最近相关工作的分类

从文章的相关工作中可以看出，传统的增量学习方法在处理灾难性遗忘问题时，主要依赖于存储部分基本类别的数据，这虽然有效，但增加了内存需求，尤其在资源有限的设备上不切实际。而不存储基本类别数据的方法，如 LwF-MC，通过知识蒸馏来保留基本类别的信息，但在实际应用中仍存在一定的性能局限。

LwM 方法创新性地引入了注意力图的概念，通过约束模型在注意力区域上的一致性，更好地保留了基本类别的信息。注意力图能够更精确地显示模型在图像中的关注区域，相比仅依赖于输出层的预测结果，提供了更丰富的上下文信息。通过这种方式，LwM 方法在不增加内存需求的情况下，有效地解决了灾难性遗忘问题，为增量学习领域提供了一种新的解决方案。

2.1.3 方法介绍

LwM 是一种增量学习方法，旨在不存储旧类别数据的情况下，通过引入注意力蒸馏损失（L_{AD}）来保留模型对旧类别的知识。该方法结合了分类损失（L_C）、知识蒸馏损失（L_D）和注意力蒸馏损失（L_{AD}）三种损失函数，以实现对新旧类别的平衡学习。作者首先给出如下定义：

- **教师模型**（Teacher model, Mt-1）：仅基于基本类别（base classes）训练的模型。
- **学生模型**（Student model, Mt）：在教师模型基础上，增量学习新类别数据得到的模型。
- **信息保留惩罚**（Information Preserving Penalty, IPP）：约束教师模型与学生模型之间差异性的损失函数。

- **增量学习任务**：增量学习的任务是在训练好教师模型 Mt-1 的基础上，使用新数据训练新模型 Mt。在初始化时，直接用 Mt-1 的参数初始化 Mt，此时二者差异不大，但随着训练的继续，二者的差异（即 IPP）逐渐增加。一个良好的增量学习系统应在最小化 IPP 的基础上，使损失函数最小，保证对新数据和旧数据的分类性能达到平衡。

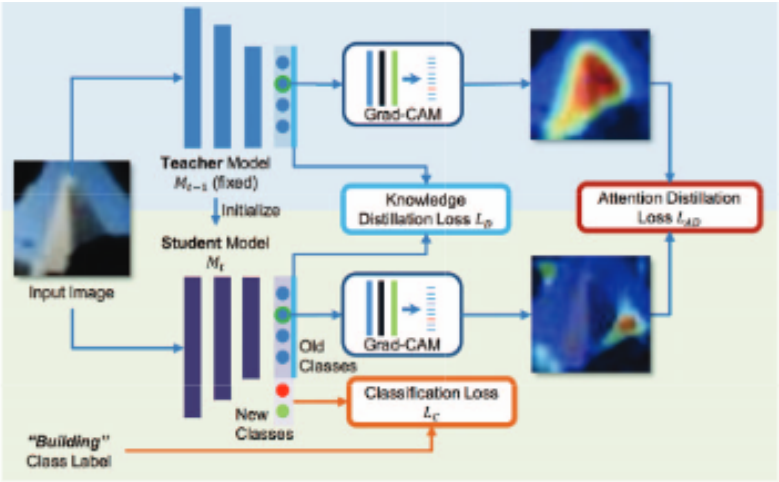


图 2: 基本的类增量学习框架

对于 LwF 模型而言，IPP 就是蒸馏损失函数（L_D），蒸馏损失函数的作用是保留模型 Mt 与 Mt-1 的预测结果差异。L_D 表现并不如预期的原因在哪里呢？作者认为，LwF 模型表现不佳的根本原因在于对旧模型学习的知识没有明确理解。作者提出，旧模型学习的知识是注意力图（attention map）。注意力图是一种对卷积神经网络进行可视化的方法，揭示了图像中哪些点对目标识别具有重要影响。例如，类别 A1 可能对应图像中的区域 B1，而类别 A2 对应区域 B2。传统的交叉蒸馏损失函数仅在输出层处理，忽略了卷积过程中的注意力图信息，从而影响了增量学习的性能。

为了改进这一问题，作者设计了如下的基于常规知识蒸馏和注意力知识蒸馏相结合的增量训练框架。其主体框架与 LwF 一致，也设计了常规的基于新类别数据的分类损失函数 L_C，针对旧类别数据的知识蒸馏损失函数（L_D）。除此之外，作者单独设计了基于旧类别数据建立的注意力蒸馏损失（L_{AD}）。具体介绍如下：

- **分类损失**（Classification Loss, L_C）：用于指导学生模型学习新类别数据，优化模型在新数据上的分类能力。

$$L_C(\mathrm{y}, \hat{\mathrm{y}}) = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{1}$$

- **知识蒸馏**（Knowledge Distillation）：通过蒸馏损失（L_D），LwM 方法在增量学习过程中保留了教师模型（M_{t-1}）和学生模型（M_t）在旧类别上的预测一

致性。

$$L_D(\mathrm{y},\hat{\mathrm{y}}) = -\sum_{i=1}^{N} y_i' \cdot \log(\hat{y}_i), \tag{2}$$

- **注意力蒸馏（**Attention Distillation**）**：引入注意力蒸馏损失（L_{AD}），通过约束教师模型和学生模型的注意力图，进一步增强模型对旧类别视觉信息的保留。

  训练 M_t 时，新类别的每一张图片分别输入到 M_{t-1} 与 M_t 中，选择 M_t 中旧类别值最高的分支，利用 grad-cam 对其进行可视化，得到一张注明了 attention 区域的图片，接着利用这两张图片构建 attention distillation loss。具体而言，对第 i 张图片，第 c 个类别，grad-cam 可视化的结果记为：

$$Q_{t-1}^{i,c} = \mathrm{vector}(\text{Grad-CAM}(i, M_{t-1}, c)) \tag{3}$$

$$Q_{t}^{i,c} = \mathrm{vector}(\text{Grad-CAM}(i, M_{t}, c)) \tag{4}$$

  vector 表示向量化，则 L_{AD}：

$$L_{AD} = \sum_{j=1}^{l} \left\| \frac{Q_{t-1,j}^{I_n,b}}{\left\| Q_{t-1}^{I_n,b} \right\|_2} - \frac{Q_{t,j}^{I_n,b}}{\left\| Q_{t}^{I_n,b} \right\|_2} \right\|_1 \tag{5}$$

最终的 loss 定义为：

$$L_{LwM} = L_C + \beta L_D + \gamma L_{AD} \tag{6}$$

可以看出，LwM 相比 LwF 模型，多了一个针对注意力蒸馏损失的约束项。那么加入这个约束项究竟有什么作用呢？作者通过一个实验说明了添加这一约束的必要性：蒸馏损失不受注意力区域退化的影响，而注意力蒸馏损失对注意力区域敏感。

如图所示，第一排图像在经过 n 个步骤的增量训练后，从 attention map 可以看出，在第 n 步训练后，注意力区域已经从电话的仪表盘处移到了下方。同时，知识蒸馏损失保持不变，约为 0.09，与第一步的 0.08 相差无几。然而，注意力蒸馏损失显著增加，从 0.12 上升到 0.82，变化明显。在第二排图像中，经过 n 个步骤的增量训练后，注意力区域没有发生变化，始终集中在仪表盘上。对应的知识蒸馏损失和注意力蒸馏损失都没有明显变化。这些实验结果表明，注意力蒸馏损失相比于知识蒸馏损失，更能准确地反映模型对关键知识的保留情况，因此需要对其进行约束以保证模型性能。

2.1.4 **实验结果**

1. 实验设计

图 3: 实验结果

为了验证 LwM（Learning without Memorizing）方法的有效性，作者在四个数据集（iILSVRC-small、iCIFAR-100、Caltech-101 和 CUBS-200-2011）上进行了实验。实验对比了三种配置：仅使用分类损失的配置 L_C、使用蒸馏损失和分类损失的 LwF-MC，以及在 LwF-MC 基础上增加注意力蒸馏损失的 LwM。每个增量步骤中，模型逐步学习新类别，以评估其在不存储旧类别数据情况下的性能。

Experiment ID\loss	L_C	L_D	L_{AD}
Finetuning	YES	NO	NO
LwF-MC	YES	YES	NO
LwM	YES	YES	YES

表 2: 不同方法用到的 loss

在现代机器学习领域，对数据集的深入理解对于开发有效的模型至关重要。各种数据集在类别数、图像数量以及评估指标上的不同，对模型的训练和验证过程具有重大影响。因此，通过对比不同数据集的特性，研究者可以更好地设计和调整其机器学习模型以适应特定的挑战。

2. 实验结果

论文给出了四次增量模型的可视化图，结果如下：

Dataset	iLSVRC-small	iCIFAR-100	CUB200-2011	Caltech-101
# classes	100	100	100	100
# training images	500	500	80% of data	80% of data
# testing images	100	100	20% of data	20% of data
# classes/batch	10	10, 20, 50	10	10
eval. metric	top-5	top-1	top-1	top-1

表 3: 使用到的数据集

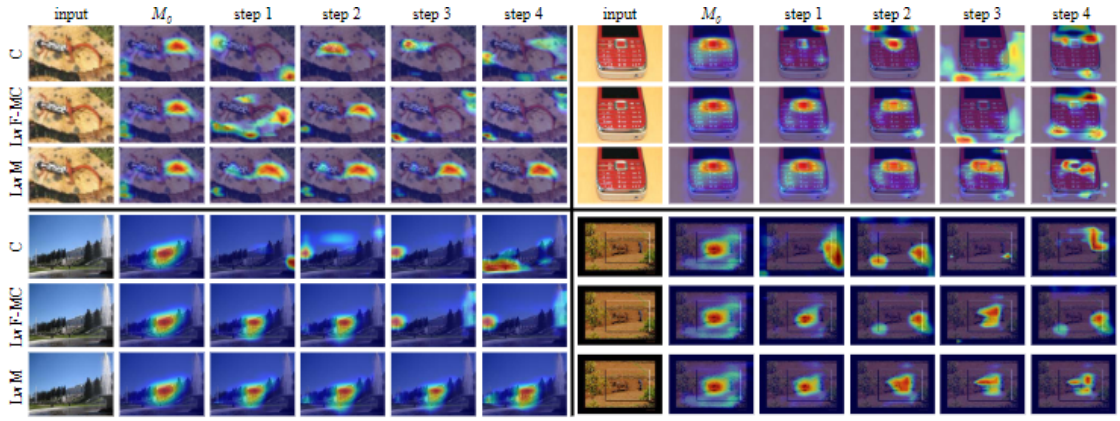


图 4: 多个增量步骤中的注意力图示例

图中展示了不同实验配置（L_C、LwF-MC 和 LwM）在多个增量步骤中的注意力图示例。图中每一行代表一种实验配置，列展示了每个增量步骤生成的注意力图，包括初始输入图像、初始教师模型（M0）的注意力图，以及四个增量步骤（step 1 到 step 4）中的注意力图。

实验结果证明了 LwM 方法在增量学习任务中的有效性。通过引入注意力蒸馏损失（L_{AD}），LwM 能够显著减少灾难性遗忘，帮助学生模型在学习新类别时保留对旧类别的视觉信息。相比之下，仅使用分类损失的 C 配置和使用蒸馏损失的 LwF-MC 配置在增量学习过程中都未能有效保留旧类别的信息，说明 L_{AD} 在增量学习中的关键作用。

图 4 展示了 LwM 方法与其他几种基线方法在不同数据集和增量学习条件下的性能对比。具体来说，图中比较了 LwM、LwF-MC、iCaRL 和仅使用分类损失进行微调（Finetuning）的方法在 iILSVRC-small 和 iCIFAR-100 数据集上的表现。图中的四个子图分别展示了这些方法在不同类别数下的 Top-1 和 Top-5 准确率。我们的方法、LwM 和基线之间的性能比较。LwM 在 iILSVRC-small 和 iCIFAR-100 数据集上优于 LwF-MC 和 “仅使用分类损失并进行微调”。鉴于 iCaRL 在访问基类数据方面具有不公平的优势,LwM 在 iILSVRC 小数据集上甚至优于 iCaRL。

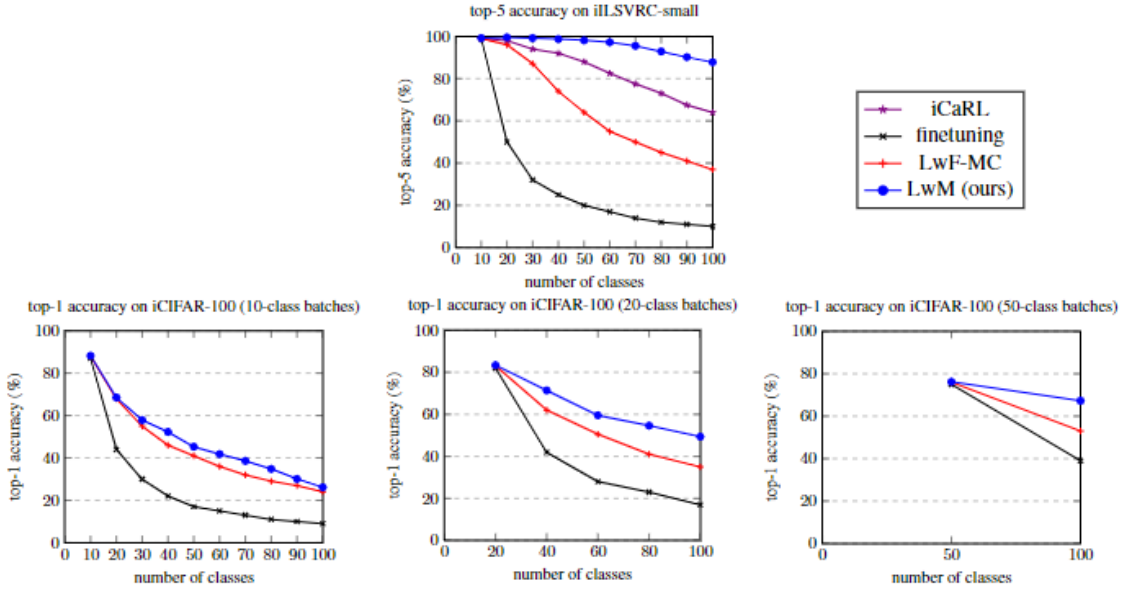


图 5: LwM 方法与其他几种基线方法在不同数据集和增量学习条件下的性能对比

实验结果清晰地展示了 LwM 方法在增量学习任务中的优越性能。通过结合分类损失（L_C）、知识蒸馏损失（L_D）和注意力蒸馏损失（L_{AD}），LwM 方法在不存储任何旧类别数据的情况下，显著提升了模型在增量学习中的表现。无论是在 iILSVRC-small 还是 iCIFAR-100 数据集上，LwM 方法都表现出色，特别是在类别数目较多的情况下，其性能优势更加明显。这些结果证明了 LwM 方法在实际应用中的广泛适用性和有效性。

通过实验和定性分析，作者证明了 LwM 方法在增量学习任务中的有效性和优越性。LwM 方法通过结合分类损失（LC）、知识蒸馏损失（LD）和注意力蒸馏损失（LAD），在不存储任何旧类别数据的情况下，显著提升了模型的性能。实验结果表明，无论是在 iILSVRC-small、iCIFAR-100、Caltech-101 还是 CUBS-200-2011 数据集上，LwM 方法都表现出色，特别是在类别数目较多的情况下，其性能优势更加明显。LwM 方法的提出为增量学习任务提供了新的思路和解决方案，具有广泛的应用前景。

2.2 Masked Autoencoders are Efficient Class Incremental Learners[2]

2.2.1 **论文动机**

过去十年来，深度学习对大多数计算机视觉任务产生了广泛而深刻的影响。然而，有一个问题始终存在于这些深度学习任务中。现实世界是非常动态的，导致数据分布随着时间的推移而变化，而深度模型在学习新任务时往往会灾难性地忘记旧任务。考虑到人类在一生中不断学习的方式，我们很自然地期望模型也能够积

累知识并基于过去的经验来逐步适应新任务。

类增量学习 (CIL) 就是这样一种方法，它旨在顺序学习新的分类任务，同时避免灾难性遗忘。能够在连续不断地接收新类数据的同时，保持对先前类的记忆。然而，通常类增量学习过程中只允许使用固定大小的内存。因此，这限制了存储来自过去任务的样本。其他研究利用生成网络（例如 GANs）来合成旧任务的样本以进行重放。尽管它们可以生成重放数据以减轻遗忘，但一个典型的缺点是生成图像的质量，并且生成的模型本身也可能会发生遗忘。

而本文的作者，选择了引入了掩码自编码器（MAEs）作为重放的基础模型。它仅需要一小部分图像块即可重建整个图像，从而实现高效的样本存储。因此，我们可以在与其他基于样本的方法相同的有限内存量下存储更多的样本。与之前的生成方法相比，MAE 的重放更加稳定，因为它使用部分线索来推断全局信息，这是与任务无关的，并且在跨任务时遗忘更少。这缓解了 GANs 在跨任务中由于图像块静止而产生的不稳定生成效果。

掩码自动编码器 (MAE) 最初被提出是为了在自监督学习场景中学习更好的特征表示。在 MAE 中，输入数据的一部分会被随机掩码，即将这些部分的数据置为零或其他特定值。然后，编码器只接收未被掩码的部分数据。但解码器可以从编码器输出的低维表示中重建整个输入数据，包括被掩码的部分。这种掩码机制可以强制模型学习到输入数据的全局结构和特征，而不仅仅是依赖于局部信息。

由此，本文的作者有了如下想法：利用掩码自动编码器（MAE）的自监督学习特性和掩码机制，增强模型的泛化能力和记忆能力，从而在 CIL 任务中取得更好的表现，解决 CIL 任务中灾难性遗忘的问题。

基于这一想法，在本项工作中，作者将 MAEs 视为高效的增量学习器，其融合了自监督重建和回放数据生成的优势，可以在 CIL 中实现高效的样本重放。同时，为了进一步提高重建图像的质量和学习效率，作者还设计了一种新颖的双边 MAE，以更好的重建图像和正则化表示。

2.2.2 相关工作

- **增量学习**：
  在过去的几年里，人们提出了各种增量学习方法，最近的工作可以粗略地分为三类：基于回放的方法、基于正则化的方法和知识蒸馏方法。

  1. 基于回放的方法
     经验回放策略保存部分旧类数据或其生成样本，并在学习新类时一并训练模型，从而减轻遗忘。

2. 基于正则化的方法
   正则化方法通过在损失函数中添加正则项，限制模型参数的变化，从而保持旧知识。
3. 知识蒸馏方法
   知识蒸馏方法通过蒸馏先前模型的知识到当前模型中，使得新模型能够保持对旧类的记忆。

- **自我监督学习**:
  自监督学习可以通过设计预训练任务，使模型能够从未标注数据中学习有效的特征表示，帮助模型学习到更具泛化性的特征，这使得将其应用于增量学习成为一个自然的考虑。早期的工作使用了诸如图像块排列或旋转预测等预训练任务。对比学习方法则通过建模样本之间的成对相似性和不相似性来实现。相比之下，掩码自动编码器（MAEs）能通过从掩码版本的输入中重建图像来学习特征表示。

2.2.3 方法介绍

在文章中，作者首先向我们介绍了类增量学习问题和基本 MAE 模型，接下来描述了他们的增量学习框架以及它所基于的双向 MAE 架构。

类增量学习问题（Class Incremental Learning, CIL）旨在顺序学习带有新类别的任务，同时防止或减轻旧任务的遗忘。在学习任务 $t \in \{1, 2, ..., T\}$ 的特定阶段，模型训练只能利用当前任务的数据 $\{(x_i^t, y_i^t)\}$，其中 x_i^t 表示任务 t 中的第 i 张图像，y_i^t 是其对应的类别标签。CIL 模型通常由特征提取器 F_θ 和一个随新任务增长的通用分类器 G_ϕ 组成。在任务 $t+1$ 时，有 C_{t+1} 个新类别被添加到 G_ϕ 中。特征提取器 F_θ 首先将输入 x 映射到一个深度特征向量 $z = F_\theta(x) \in \mathbb{R}^d$，其中 d 是输出特征表示的维度，然后统一分类器 $G_\phi(z) \in \mathbb{R}^{C_1:t}$ 产生类别 $C_1 : t$ 上的概率分布，用于对输入 x 进行预测。

MAE（Masked Autoencoder）框架用于分类任务。MAE 首先将输入图像 x 切分为不重叠的图像块，我们将完整图像 x 中的图像块数量记为 N_f。在切分之后，MAE 随机遮盖 N_f 图像块中的一部分，其比例为 $r \in [0, 1]$，剩下的图像块数量为 $N = N_f \times (1 - r)$。然后，这些采样的 $K \times K$ 像素图像块通过多层感知器（MLP）映射到维度为 D 的视觉嵌入。连接一个类别标记后，结果是一个大小为 $\mathbb{R}^{(N+1)\times D}$ 的张量。在对原始图像块位置进行位置编码后，该输入被传递到 MAE Transformer 编码器中。此操作保持嵌入的形状不变。输出的类别标记嵌入可以通过交叉熵损失 L_{cls}^t 用于分类。

对于 MAE 解码器，可学习的遮盖标记被插入嵌入中，以替代被遮盖的图像

块，MAE 编码器的输出形状从 $\mathbb{R}^{(N+1)\times D}$ 变为 $\mathbb{R}^{(N_f+1)\times D}$。虽然解码器不用于分类，但它帮助网络反向传播图像级重建监督到嵌入级别。这稳定了图像嵌入并有助于优化过程。此外，解码后的重建图像提供了更丰富、更高质量的回放数据。输入图像 x 和重建图像 $\hat{x}$ 之间的均方误差被用作重建损失函数 $L^t_{\text{rec}}(x, \hat{x})$。

接下来，在此基础上，作者提出了一种双向 MAE 学习全局和细节分类以及图像重建知识，以进一步提高重建质量和嵌入多样性。其整体框架如图 6 所示：

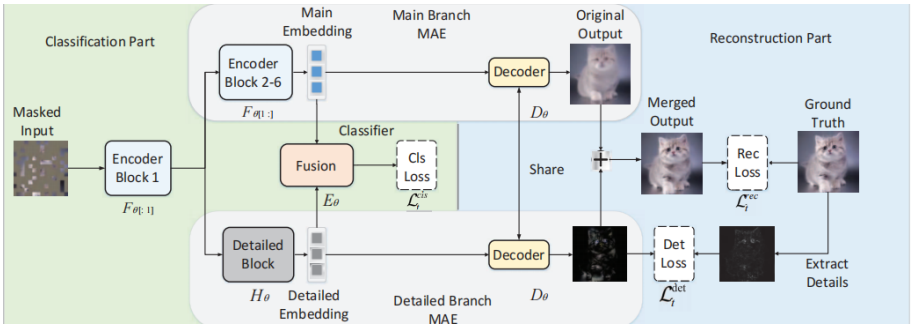


图 6: 双向 MAE 的整体框架

该双边 MAE 将掩码输入通过两个分支处理。在嵌入级别进行双边融合用于分类，在图像级别进行融合用于重建学习，产生高质量的重放数据，并为 CIL 提供稳定的自监督。可以从少量输入图像块生成完整图像，并将重建的图像用于重放。

使用 $F_\theta[:1]$ 和 $F_\theta[1:]$ 分别表示 Transformer 编码器的第一个和后续的块。令 H_θ 和 E_θ 分别表示图 6 中的详细块和嵌入融合模块，它们分别是标准的 MLP 层和注意力块。则分类损失计算如下：

$$f = F_\theta[:1](\text{mask}(x, r)) \tag{1}$$

$$z = E_\theta(F_\theta[1:](f), H_\theta(f)) \tag{2}$$

$$L^t_{\text{cls}}(x, y) = L_{\text{ce}}(G_\phi(z), y) \tag{3}$$

其中，$\text{mask}(x, r)$ 表示对图像 x 应用比率为 r 的随机掩码，f 是由第一个编码器块提取的嵌入，它是我们双边 MAE 的两个分支的输入，$G_\phi(z)$ 是用于交叉熵损失的估计类别分布。

对于详细的损失，作者发现，在频域中工作使网络更容易关注高频细节，而这正是详细分支应该重建的内容。因此，作者定义了一个频率掩膜函数 $M(\cdot)$，该函

数将其参数（一个图像块）转换到频域，然后使用围绕原点的圆形掩膜屏蔽低频。如图 6 所示，MAE 解码器由模型的两个分支共享，因为它们具有相似的重建任务以及输入和输出形状。设 D_θ 为这个共享解码器，那么两个分支的图像级输出和重建损失 L^t_{rec} 为：

$$f = F_\theta[:1](\text{mask}(x, r)) \tag{4}$$

$$\hat{x} = D_\theta(F_\theta[1:](f)) \tag{5}$$

$$\hat{x}^d = \text{ifft2}(M(D_\theta(H_\theta(f)))) \tag{6}$$

$$\hat{x} = \hat{x} + \hat{x}^d \tag{7}$$

$$L^t_{\text{rec}} = L_{\text{mse}}(x, \hat{x}) \tag{8}$$

(7)

其中 $\hat{x}$ 和 $\hat{x}^d$ 分别是主输出和残差详细输出，ifft2 是逆快速傅里叶变换。详细损失 (L^t_{det}) 也借助频率掩膜函数 (M) 来比较详细分支的输出和输入图像两个 MAE 重建之间的差异：

$$\hat{x}_1 = D\theta(F_\theta(\text{mask}(x, r1))) \tag{9}$$

$$\hat{x}_2 = D\theta(F_\theta(\text{mask}(x, r2))) \tag{10}$$

$$L^t_{\text{det}} = |M(D_\theta(H_\theta(f))) - M(\hat{x}_2 - \hat{x}_1)|_1 \tag{11}$$

其中 $\hat{x}_1$ 和 $\hat{x}_2$ 是使用不同掩膜比率 $(r1)$ 和 $(r2)$ 的两个重建图像（从梯度计算图中分离）。残差差异 $(\hat{x}_2 - \hat{x}_1)$ 为频域中用于详细分支的监督损失 (L^t_{det})。如图 7 所示意。

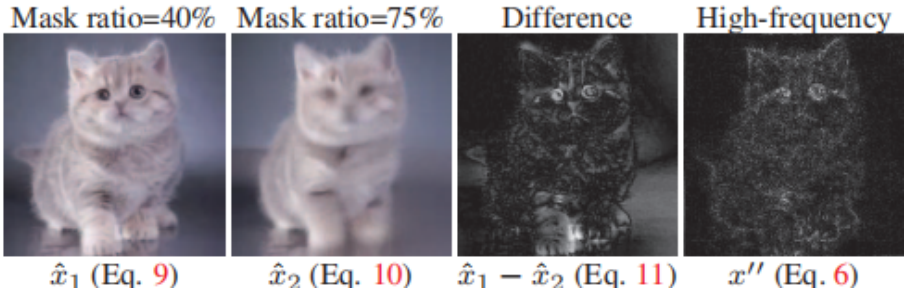


图 7: 使用不同遮掩比 r1 和 r2 重建图像的示例

训练的总体损失是分类损失 (L^t_{cls})、重建损失 (L^t_{rec}) 和详细损失 (L^t_{det}) 的加权和：

$$L_t = \lambda_{\text{cls}} L_{\text{cls}}^t + \lambda_{\text{rec}} L_{\text{rec}}^t + \lambda_{\text{det}} L_{\text{det}}^t \tag{12}$$

整个双向 MAE 算法的伪代码如算法 1 所示：

Algorithm 1 Pseudocode of our Bilateral MAE.

Input: The number of task T, training samples $D_t = \{(x_i, y_i)\}_t$ of task t, model Θ^0, replay buffer ϵ, and masking ratios r, r_1, r_2.
Output: model Θ^T

for $t \in \{1, 2, ..., T\}$ **do**
  $\Theta^t \leftarrow \Theta^{t-1}$
  $R_t \leftarrow$ ReconstructOldSamples (ϵ_t, r)
  while not converged **do**
    $(x, y) \leftarrow$ Sample (R_t, D_t)
    $(\mathcal{L}_t^{\text{cls}}, \mathcal{L}_t^{\text{rec}}) \leftarrow$ BilateralMAE (x, y)
    $(\hat{x}_1, \hat{x}_2) \leftarrow$ MaskAndReconstruct (x, r_1, r_2)
    $\mathcal{L}_t^{\text{det}} \leftarrow$ ComputeDetailLoss $(\hat{x}_1, \hat{x}_2)$
    train Θ^t by minimizing $\mathcal{L}_t$ from Eq. 12
  end while
end for

图 8: 算法 1

2.2.4 实验结果

作者在三个数据集上进行实验：CIFAR-100、ImageNet-Subset 和 ImageNet-Full ，以评估该方法的性能。对于 CIFAR-100 和 ImageNet-Subset，作者测试了 10、20 和 50 任务的场景，（所有任务中的类别数相等）。同时在 ImageNet-Full 上评估了 10 任务场景，其中每个任务包含 100 个新类。为了衡量训练过程中所有任务完成后的总体准确率，作者报告了每个任务后的已学习任务的平均准确率，以及增量学习结束时所有任务的准确率。

实验过程中的一些参数设置如下：对所有数据集使用相同的网络。为了防止数据泄露，模型从头开始训练，批量大小为 1024，使用 Adam 优化器，初始学习率为 1e-4，并采用余弦衰减。公式 12 中的损失权重设置为 cls = 0.01， rec = 1.0 和 det = 1.0。掩码比例设置为 r = 0.75，r1 = 0.75 和 r2 = 0.4。每个任务训练 400 个周期。

接下来，作者分别在这三个数据集上将他们的方法与与最新的技术进行了比较，包括 DER 和 DyTox 。具体结果如下

- CIFAR-100：
  图 9 中报告了平均准确率（Avg）、最后一个任务后的准确率（Last）和平均遗忘率（F）的结果。显然，在每种设置下，作者的方法都以较大优势超过了其他方法。对于较长的任务序列，这种双边 MAE 受益于自监督重建和更丰富的重放数据，与其他方法相比，它的遗忘程度要小得多。整体准确率曲线如图 10 所示。使用相同量的重放存储，本文的方法在所有三种情景下，在最后一个任务后的准确率上都比 DyTox 高出约 6

Method	N=10			N=20			N=50		
	Avg ↑	Last ↑	F ↓	Avg ↑	Last ↑	F ↓	Avg ↑	Last ↑	F ↓
iCaRL [29]	65.27	50.74	31.23	61.20	43.75	32.40	56.08	36.62	36.59
UCIR [12]	58.66	43.39	35.67	58.17	40.63	37.75	56.86	37.09	38.13
BiC [36]	68.80	53.54	28.44	66.48	47.02	29.30	62.09	41.04	34.27
PODNet [8]	58.03	41.05	41.47	53.97	35.02	36.70	51.19	32.99	40.42
DER w/o P [40]	75.36	65.22	15.02	74.09	62.48	23.55	72.41	59.08	26.73
DER [40]	74.64	64.35	15.78	73.98	62.55	23.47	72.05	59.76	26.59
DyTox [9]	75.47	62.10	15.43	75.10	59.41	21.60	73.89	57.21	24.22
Ours	**79.12**	**68.40**	**12.17**	**78.76**	**65.22**	**14.39**	**76.95**	**63.12**	**18.34**

图 9: 在 10 任务、20 任务和 50 任务场景下，CIFAR-100 的平均准确率、最后阶段准确率和遗忘率的结果

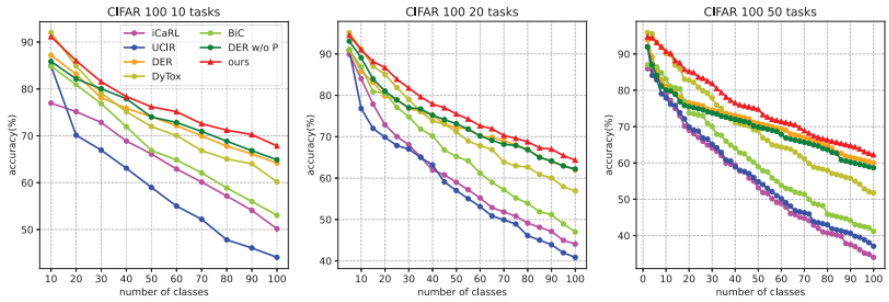


图 10: 在 CIFAR-100 数据集上，随着增量任务的增加，在 10 任务、20 任务和 50 任务场景下的性能演变

- ImageNet-Subset 与 ImageNet-Full：图 11 和图 12 中分别报告了 ImageNet-Subset 和 ImageNet-Full 上的性能结果。本文的方法在最后一个任务后的准确率上比 DyTox 分别高出 1.19%、2.53% 和 2.85%（在 10 任务、20 任务和 50 任务设置下的绝对增益）。在每个阶段更高的平均准确率和更低的遗忘率证明了该方法在缓解遗忘方面的有效性。
  图 13 中展示了 ImageNet-Subset 上的性能演变。作者的方法在第一个任务中的准确率与 DyTox 相似，但在后续任务中，这一方法超越了所有其他方

法，尤其是在长任务序列中。在更大规模的 ImageNet-Full 上，双向 MAE 在所有指标上显著超越其他方法，约高出 3%。

Method	N=10			N=20			N=50		
	Avg ↑	Last ↑	F ↓	Avg ↑	Last ↑	F ↓	Avg ↑	Last ↑	F ↓
BiC [36]	64.96	55.07	31.32	59.40	49.35	34.70	53.75	44.56	40.23
PODNet [8]	63.44	51.75	35.63	55.11	45.37	41.70	51.72	42.94	44.65
DER w/o P [40]	77.18	66.70	14.86	72.70	61.74	20.76	70.44	58.87	24.20
DER [40]	76.12	66.06	15.09	72.56	61.51	20.46	69.77	58.19	25.35
DyTox [9]	77.15	69.10	14.66	73.13	61.87	17.32	71.51	60.02	20.54
Ours	**79.54**	**70.29**	**12.04**	**75.20**	**64.40**	**14.89**	**74.42**	**62.87**	**17.22**

图 11: 在 10、20 和 50 任务场景下，ImageNet-Subset 的平均准确率 (%)、最后阶段准确率（%）和遗忘率 F（%）的结果。

Method	top-1		top-5	
	Avg ↑	Last ↑	Avg ↑	Last ↑
iCaRL [29]	38.40	22.70	63.70	44.00
Simple-DER	66.63	59.24	85.62	80.76
DER w/o P [40]	68.84	60.16	88.17	82.86
DER [40]	66.73	58.62	87.08	81.89
DyTox [9]	71.29	63.34	88.59	84.49
Ours	**74.76**	**66.15**	**91.43**	**87.13**

图 12: ImageNet-Full 在 10 个增量任务上的结果

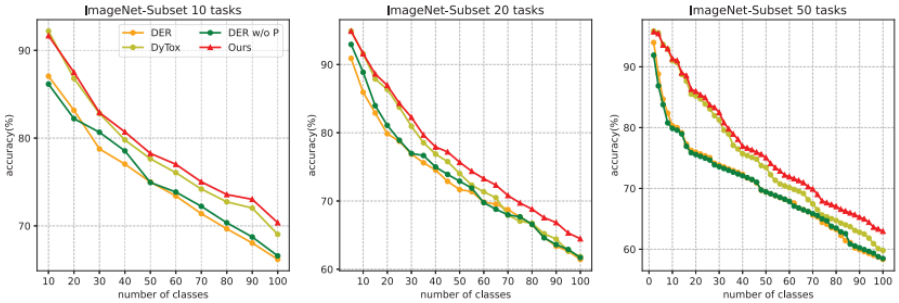


图 13: 在 ImageNet-Subset 上的增量任务中的性能演变

双边 MAE 方法在 CIFAR-100、ImageNet-Subset 和 ImageNet-Full 三个不同数据集上的不同 CIL 设置下实现了最先进的性能。

2.3 iCaRL: Incremental Classifier and Representation Learning[3]

2.3.1 论文动机

传统机器学习在持续接收新类别数据时确实容易遗忘旧类别知识，这会导致性能下降。因此，作者提出了一种结合分类器学习和表示学习的方法，旨在解决这个问题。他们通过引入类别记忆和样本重播机制，实现了持续学习和更新模型而不忘记旧类别知识的目标。这种方法能够有效应对增量学习中的遗忘问题和新类别学习挑战，为持续学习系统提供了一种有效的解决方案。

在这篇论文中，作者介绍了一种新的训练策略，即 iCaRL，它允许以类增量的方式进行学习：只需要保留少量的旧数据，就可以逐步学习新类别。iCaRL 不仅仅是一个分类器，它还同时学习强分类器和数据表示法。这种综合的学习方法使得模型能够在接收到新类别数据时不仅学习新知识，还能够有效地保留旧知识，从而实现了持续学习的目标。

2.3.2 相关工作

增量学习是机器学习领域的一个重要问题，它涉及到在系统已经学习过一部分数据后，如何有效地继续学习新的数据。在这方面，有一些与 iCaRL 相关的工作：

- **类别增量学习方法**：许多先前的研究工作着眼于类别增量学习，即系统需要逐步接收新的类别并保持对先前类别的知识。传统的方法包括使用原型存储、特征选择、迁移学习等技术。然而，这些方法往往面临着遗忘先前类别知识、灾难性遗忘等问题。
- **存储和选择原型**：
  一些方法尝试在增量学习过程中存储一些代表性样本或原型，并利用它们来保持对先前类别的知识。这些方法通常涉及到原型选择的问题，即如何从新样本中选择出最具代表性的样本。iCaRL 中的 herding-based 步骤就是一个基于聚类的原型选择方法。
- **表示学习与增量学习的结合**：
  一些研究尝试将表示学习与增量学习相结合，通过在学习过程中同时学习特征表示和分类器来提高性能。这些方法通常利用深度神经网络进行端到端的训练，以实现特征的自动学习和分类器的优化。

iCaRL 在这些方面都有所贡献，并通过将分类器和表示学习相结合，以及使用原型存储和选择等技术，提供了一种有效的增量学习方法。其对于解决遗忘问

题和实现长期增量学习具有重要意义

2.3.3 方法介绍

在 iCaRL 论文中，作者进行了一系列详细的实验来验证他们提出的增量学习方法的有效性。首先，他们使用了 CIFAR-100 数据集，该数据集包含 100 个类别的图像，每个类别有 600 张训练图像和 100 张测试图像。作者将数据集按照类别分为 10 个不相交的子集，每个子集包含 10 个类别。在实验中，作者使用卷积神经网络作为基础模型，并在每个阶段使用随机采样的训练数据进行模型训练。

iCaRL 通过计算每个类别的样本在模型中的嵌入向量的均值，然后选择距离这个均值最近的 topk 个样本存储下来，用于维护一个旧数据集。这样做的目的是为了在后续的模型训练中，可以直接使用这些存储的旧数据样本，以帮助模型保持对旧类别的识别能力。iCaRL 通过存储每个类别的样本的平均表示，然后选择距离平均表示最接近的一些样本，来构建一个旧数据集。这个旧数据集可以帮助模型在学习新类别的同时，保持对旧类别的知识，从而实现增量学习的效果。这种简单直观的数据重播方法是 iCaRL 方法的一个关键特点，有效地解决了增量学习中遗忘问题和新旧类别学习的平衡。

在基准阶段，模型只接收前 50 个类别的数据进行训练，然后在增量学习阶段，模型接收剩余 50 个类别的数据进行增量学习。作者提出了一种类别记忆和样本重播的策略，用于维护旧类别的知识，并引入了一种损失函数来平衡新旧类别的学习。在每个阶段结束时，作者评估了模型在测试集上的准确率，包括对旧类别和新类别的分类准确率。

实验结果表明，iCaRL 方法在增量学习任务中表现出色，能够有效地保持对旧类别的识别能力，同时快速学习新类别的知识。与其他增量学习方法相比，iCaRL 在遗忘问题和新类别学习方面取得了更好的性能。

Algorithm 1 iCaRL CLASSIFY 描述了 iCaRL 方法中的分类过程。该算法首先计算每个类别示例集合中样本的嵌入向量均值，然后通过计算待分类图像 x 的嵌入向量与每个类别均值的距离，选择距离最近的类别作为输出的类别标签。Algorithm 2 iCaRL INCREMENTALTRAIN 描述了 iCaRL 方法中的增量训练过程。给定一组训练样本 Xs,...,Xt 和内存大小 K，当前模型参数 Θ 以及当前的示例集合 P = (P1,...,Ps−1)。该算法首先通过更新表示（UPDATEREPRESENTATION）函数来更新模型参数 Θ，然后根据每个类别的内存大小 m，对旧的示例集合进行缩减（REDUCEEXEMPLARSET）操作，以及构建新的示例集合（CONSTRUCTEXEMPLARSET）操作。最后更新示例集合 P 为新的示例集合。

Algorithm 1 iCaRL CLASSIFY

input x // image to be classified
require $\mathcal{P} = (P_1, \ldots, P_t)$ // class exemplar sets
require $\varphi : \mathcal{X} \to \mathbb{R}^d$ // feature map
  for $y = 1, \ldots, t$ **do**
    $\mu_y \leftarrow \frac{1}{|P_y|} \sum_{p \in P_y} \varphi(p)$ // mean-of-exemplars
  end for
  $y^* \leftarrow \underset{y=1,\ldots,t}{\operatorname{argmin}} \|\varphi(x) - \mu_y\|$ // nearest prototype
output class label y^*

Algorithm 2 iCaRL INCREMENTALTRAIN

input $X^s, \ldots, X^t$ // training examples in per-class sets
input K // memory size
require Θ // current model parameters
require $\mathcal{P} = (P_1, \ldots, P_{s-1})$ // current exemplar sets
  $\Theta \leftarrow$ UPDATEREPRESENTATION$(X^s, \ldots, X^t; \mathcal{P}, \Theta)$
  $m \leftarrow K/t$ // number of exemplars per class
  for $y = 1, \ldots, s-1$ **do**
    $P_y \leftarrow$ REDUCEEXEMPLARSET(P_y, m)
  end for
  for $y = s, \ldots, t$ **do**
    $P_y \leftarrow$ CONSTRUCTEXEMPLARSET(X_y, m, Θ)
  end for
  $\mathcal{P} \leftarrow (P_1, \ldots, P_t)$ // new exemplar sets

图 14: 算法 1，2

Algorithm 3 iCaRL UPDATEREPRESENTATION

input $X^s, \ldots, X^t$ // training images of classes $s, \ldots, t$
require $\mathcal{P} = (P_1, \ldots, P_{s-1})$ // exemplar sets
require Θ // current model parameters
  // form combined training set:

$$\mathcal{D} \leftarrow \bigcup_{y=s,\ldots,t} \{(x, y) : x \in X^y\} \cup \bigcup_{y=1,\ldots,s-1} \{(x, y) : x \in P^y\}$$

  // store network outputs with pre-update parameters:
  for $y = 1, \ldots, s-1$ **do**
    $q_i^y \leftarrow g_y(x_i)$ for all $(x_i, \cdot) \in \mathcal{D}$
  end for
  run network training (*e.g.* BackProp) with loss function

$$\ell(\Theta) = -\sum_{(x_i, y_i) \in \mathcal{D}} \Big[\sum_{y=s}^{t} \delta_{y=y_i} \log g_y(x_i) + \delta_{y \neq y_i} \log(1 - g_y(x_i)) + \sum_{y=1}^{s-1} q_i^y \log g_y(x_i) + (1 - q_i^y) \log(1 - g_y(x_i)) \Big]$$

  that consists of *classification* and *distillation* terms.

图 15: 算法 3

Algorithm 3 iCaRL UPDATEREPRESENTATION 描述了 iCaRL 方法中的更新表示过程。该算法接受训练图像集合 Xs,...,Xt（代表类别 s 到 t 的训练图像）、示例集合 P = (P1,...,Ps−1)（代表类别 1 到 s-1 的示例集合）以及当前模型参数 Θ 作为输入。算法首先将训练集合和示例集合合并为一个组合训练集合 D，然后使用预更新参数来存储网络输出。

Algorithm 4 iCaRL CONSTRUCTEXEMPLARSET

input image set $X = \{x_1, \ldots, x_n\}$ of class y
input m target number of exemplars
require current feature function $\varphi : \mathcal{X} \to \mathbb{R}^d$
  $\mu \leftarrow \frac{1}{n} \sum_{x \in X} \varphi(x)$ // current class mean
  for $k = 1, \ldots, m$ **do**
    $p_k \leftarrow \underset{x \in X}{\operatorname{argmin}} \left\| \mu - \frac{1}{k} [\varphi(x) + \sum_{j=1}^{k-1} \varphi(p_j)] \right\|$
  end for
  $P \leftarrow (p_1, \ldots, p_m)$
output exemplar set P

Algorithm 5 iCaRL REDUCEEXEMPLARSET

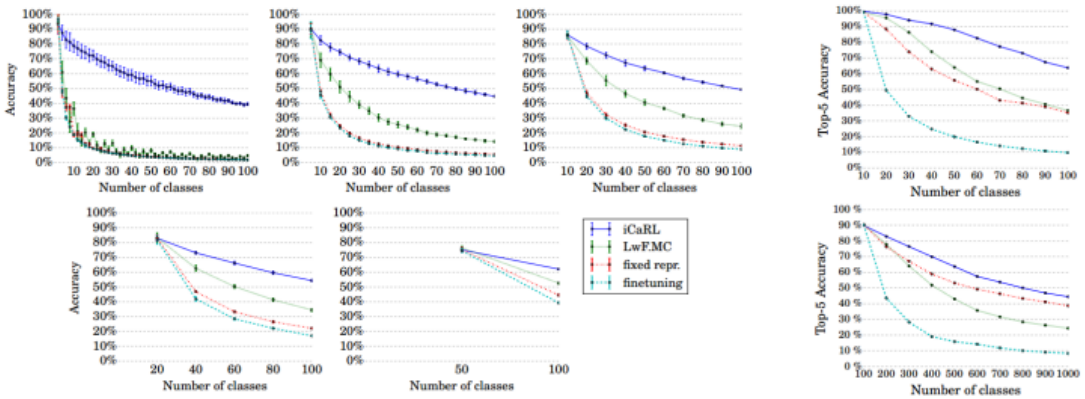
input m // target number of exemplars
input $P = (p_1, \ldots, p_{|P|})$ // current exemplar set
  $P \leftarrow (p_1, \ldots, p_m)$ // *i.e.* keep only first m
output exemplar set P

图 16: 算法 4，5

Algorithm 4 CONSTRUCTEXEMPLARSET 用于构建新类别的示例集，其中输入包括类别 y 的图像集合 X 和目标示例数量 m。该算法首先计算当前类别的平均特征向量 ，然后选择与平均特征向量距离最小的 m 个图像作为示例。最后输出构建的示例集 P。Algorithm 5 REDUCEEXEMPLARSET 用于减少示例集的大小至 m，输入为目标示例数量 m 和当前示例集 P，输出为减少后的示例集 P。

$$\mathcal{L}_{kdl}^{e} = \sum_{l} \gamma_l \left\| D_T(\mathrm{x}_\mathrm{t}^1)_l - E_S(\mathrm{x}_\mathrm{t}^1)_l \right\|_1 \tag{8}$$

2.3.4 **实验结果**



(a) Multi-class accuracy (averages and standard deviations over 10 repeats) on iCIFAR-100 with 2 (top left), 5 (top middle), 10 (top right), 20 (bottom left) or 50 (bottom right) classes per batch.

(b) Top-5 accuracy on iILSVRC-small (top) and iILSVRC-full (bottom).

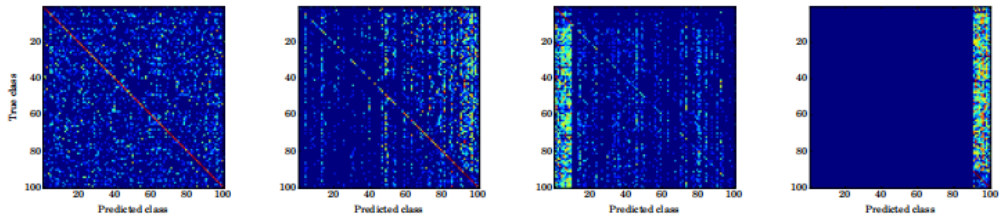
图 17: 实验结果

在 iCIFAR-100 和 iILSVRC 上进行类增量训练的实验结果：报告了在某个时间

点上观察到的所有类别的多类准确性。在这种设置下，iCaRL 明显优于其他方法。在训练了第一批数据后固定数据表示（fixed repr.）的效果比基于蒸馏的 LwF.MC 差，除了 iILSVRC-full。没有防止灾难性遗忘的微调网络（finetuning）取得了最差的结果。

除了 iCaRL 外，作者实现并测试了三种备选的类增量方法。微调方法学习了一个普通的多类网络，没有采取任何措施来防止灾难性遗忘。它也可以解释为通过微调先前学习的多类分类网络来学习新进入的类别的多类分类器。固定表示方法也学习了一个多类分类网络，但以一种方式防止灾难性遗忘。它在处理了第一批类别并处理了相应类别后冻结了特征表示，对于后续的类别批次，只训练新类别的权重向量。最后，我们还将其与一种网络分类器进行比较，该分类器试图通过在学习过程中使用蒸馏损失来防止灾难性遗忘，就像 iCaRL 一样，但不使用示例集。对于分类，它使用网络输出值本身。这本质上是无遗忘学习方法，但应用于多类分类，因此我们将其称为 LwF.MC。

结果如图所示。可以看出，iCaRL 明显优于其他方法，而且在增量设置更多的情况下表现得更好（即同时处理的类别越少）。在其他方法中，基于蒸馏的网络训练（LwF.MC）始终排名第二，除了在 iILSVRC-full 中，此时最好是在处理完前 100 类后固定表示。微调始终取得最差的结果，证实了灾难性遗忘确实是类增量学习中的一个主要问题。



(a) iCaRL (b) LwF.MC (c) fixed representation (d) finetuning

作者在该图表中进一步提供了对不同方法行为的见解。它展示了使用每次处理 10 个类别的批次进行训练后在 iCIFAR-100 上的 100 类分类器的混淆矩阵（可以在补充材料中找到更大的版本）。可以看到非常特征化的模式：iCaRL 的混淆矩阵在所有类别上都呈现出均匀的模式，无论是对角线条目（即正确的预测）还是非对角线条目（即错误）。这表明 iCaRL 对学习过程中早期或晚期遇到的类别没有固有的偏好或偏离。特别是，它不会受到灾难性遗忘的影响。

该图比较了 iCaRL、混合分类器和 NCM 分类器在不同内存预算下的效果。这些方法都使用与 iCaRL 相同的数据表示，但它们的分类规则不同。实验结果表明，所有方法都受益于更大的内存预算，这表明 iCaRL 的表示学习步骤确实受益于更

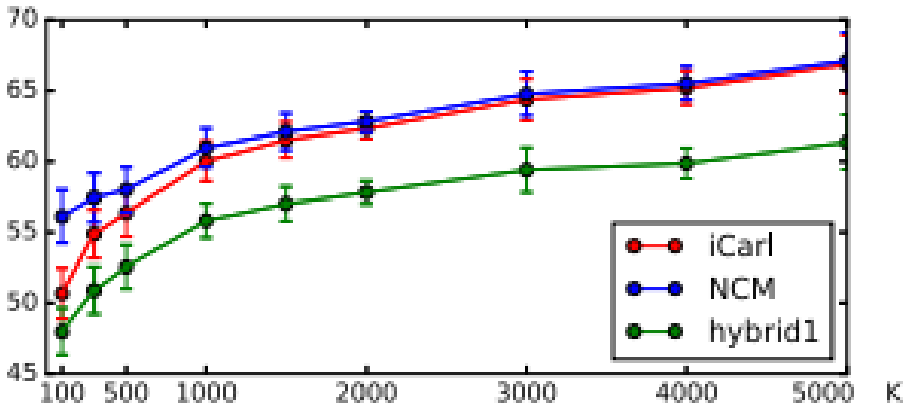


图 18: 对于不同的内存预算 K，在每批次包含 10 个类别的 iCIFAR-100 数据集上的平均增量准确率

多的原型。

当有足够多的原型(至少 1000 个)时,iCaRL 的范例均值分类器的性能与 NCM 分类器相似，而通过网络输出进行分类的方法则不具有竞争力。这说明 iCaRL 在具有足够大的内存预算时，其分类性能与其他基于原型的方法相当，甚至更优。

2.4 End-to-End Incremental Learning[4]

2.4.1 **论文动机**

”End-to-End Incremental Learning” 的实验动机主要来自于对传统增量学习方法存在的一些限制和挑战的认识。传统的增量学习方法往往在处理长期增量学习任务时面临着以下问题：

- **遗忘问题**：传统的增量学习方法在学习新类别的同时往往会遗忘先前类别的知识。这会导致模型性能下降，无法保持对先前类别的准确分类能力。
- **固定内存限制**：传统方法通常依赖于固定大小的内存来存储先前类别的信息，这会导致内存资源的浪费或者对新类别的学习能力受限。

基于这些挑战，作者提出了一种端到端的增量学习方法，旨在解决传统增量学习方法的局限性。他们的实验动机主要包括以下几点：

- **长期增量学习需求**：随着数据不断增长和演化，机器学习系统需要能够持续地适应新的数据和类别。因此，需要一种能够在长期时间内有效学习和演化的增量学习方法。
- **保留先前知识**：在处理新类别的同时，保留先前类别的知识是至关重要的。

一种理想的增量学习方法应该能够在学习新知识的同时保持对先前知识的记忆和利用。

- **有效利用有限资源**：实际应用中，资源通常是有限的，包括内存、计算能力等。因此，一种好的增量学习方法应该能够有效地利用有限的资源，并在学习过程中动态地适应资源的变化。

基于以上动机，作者设计了一系列实验来验证他们提出的端到端增量学习方法在处理长期增量学习任务时的性能表现。这些实验旨在验证方法对遗忘问题的缓解程度、对先前知识的保留能力以及对内存资源的有效利用程度。通过这些实验，他们希望证明所提方法的有效性和实用性，为增量学习领域的进一步研究和应用提供新的思路和方法。

2.4.2 实验方法

首先，作者使用交叉蒸馏损失函数训练深度网络，该损失函数结合了交叉熵损失和蒸馏损失，能够有效地在训练过程中保留旧类别的知识并学习新类别的分类。这个深度网络可以基于大多数用于分类任务的深度模型的架构，因为作者的方法不依赖于特定的属性。

代表性内存：用于存储和管理旧类别中最具代表性的样本。代表性内存执行两个操作：选择要存储的新样本和删除多余的样本。新样本的选择基于与类别平均样本的距离进行排序，选择距离最近的样本。删除操作则是从每个类别的样本集末尾删除多余的样本。

深度网络的架构，包括特征提取器和分类层。特征提取器将输入图像转换为特征向量，而分类层生成一组逻辑，用于计算分类分数。在训练阶段，使用交叉蒸馏损失函数计算更新网络权重的梯度，在测试阶段，损失函数被 softmax 层替换。这些设计和实现细节组合在一起，使作者的方法能够在增量学习任务中取得最先进的结果。

$$L(\omega) = L_C(\omega) + \sum_{f=1}^{F} L_{D_f}(\omega),$$

图 19: 交叉蒸馏损失函数的定义

LC( ) 是应用于新旧类样本的交叉熵损失，LDf 是分类层 f 的蒸馏损失，F 是分类层的总数旧类

作者提出的增量学习方法包括四个主要阶段，每个阶段都在实现不同的目标。

$$L_C(\omega) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} p_{ij}\log q_{ij},$$

图 20: 蒸馏损失的定义

- **训练集的构建**：其中包括新类别的样本和存储在代表性内存中的旧类别的样本。每个样本都会分配两个标签，一个用于分类，另一个用于蒸馏，以帮助网络保留旧知识。
- **训练过程**：在此阶段使用增强的训练集和相应的标签来生成梯度，以优化深度模型。在训练期间，所有权重都会更新，这与其他增量方法不同，这些方法通常会冻结特征提取器并只训练分类层。
- **平衡微调阶段**：以解决旧类别样本和新类别样本之间的不平衡。在这个阶段，使用较小的学习率和平衡的样本子集来微调模型，以确保每个类别都有相同数量的样本。
- **代表性内存的更新阶**：其中删除了旧类别样本，并为新类别的样本腾出空间，然后根据选择算法选择并存储最具代表性的新类别样本。这些阶段的组合使得模型能够在增量学习任务中取得良好的性能，并且能够在连续的增量步骤中保留旧知识，适应新类别的出现。

在作者的研究中，他们使用了 MatConvNet 作为实现端到端增量学习模型的平台。每个增量步骤包括 40 个 epoch 的训练，并额外进行 30 个 epoch 的平衡微调。在前 40 个 epoch 中，学习率从 0.1 开始，每 10 个 epoch 就会除以 10。微调阶段使用相同的减少量，但起始学习率为 0.01。他们采用标准的随机梯度下降法进行网络训练，每个小批量包含 128 个样本，权重衰减为 0.0001，动量为 0.9。为了最大限度地减少过拟合，他们在梯度上应用 L2 正则化和随机噪声。根据数据集的特性，作者使用了特定于数据集的 CNN/深度模型，对于 CIFAR-100 使用 32 层 ResNet，对于 ImageNet 使用 18 层 ResNet。在训练 CIFAR-100 模型时，他们对输入数据进行了归一化处理，而在 ImageNet 的情况下，则只进行了减法操作。

作者采用了分割传统多类数据集的类的标准设置来进行类增量学习。在实验中，作者使用了 iCaRL、hybrid1 和 LwF.MC 等方法作为基准进行比较。数据增强是在训练步骤之前的第二和第三阶段执行的。具体的数据增强操作包括亮度调整、对比度归一化、随机裁剪和图像镜像。这些操作有助于提高模型的泛化能力和鲁棒性。通过这些设置和操作，作者能够在实验中取得良好的结果，并展示了他们的方法在增量学习任务中的有效性和可行性。

2.4.3 实验结果

作者在 CIFAR-100 数据集上进行了实验分析。CIFAR-100 数据集由 100 个类别的 60,000 张 32 × 32 的 RGB 图像组成，每个类别有 600 张图像，其中有 500 张用于训练，100 张用于测试。作者将这 100 个类别随机分为 2、5、10、20 和 50 个类的分组。因此，分别有 50、20、10、5 和 2 个增量训练步骤。在每次增量步骤之后，生成的模型将在由所有已训练类别（即旧类别和新类别）组成的测试数据上进行评估。作者在每个增量步骤的评估指标是标准的多类别准确率。文中以不同的随机类别顺序执行五次实验，报告平均准确率和标准差。此外，文中还报告了平均增量准确率（每个增量步骤的准确率值的平均值）。

在 CIFAR 上，作者遵循文中描述的数据增强步骤，并为每个训练样本生成 11 个新样本：一个亮度归一化，一个对比度归一化，三个随机裁剪（应用于原始图像、亮度和对比度图像），以及六个镜像（应用于先前生成的图像和原始图像）。

作者在 CIFAR-100 数据集上共进行了三类实验。第一类实验中，他们按照文中的实验协议设置了代表性记忆单元的最大存储容量。第二类实验评估了在没有固定内存大小，而是为每个旧类使用了一个固定数量的样本的方法。在这种情况下，随着每次增量步骤的进行，当新类存储在代表性记忆单元中时，内存大小会增加。最后，作者进行了消融研究，以分析这一方法中不同组件对准确性的影响。

- **固定内存大小**

  作者评估了五种不同的类别划分顺序和每次增量步骤为 2、5、10、20 和 50 个类别的情况。为了确保结果的可比性，所有评估方法的类别顺序是相同的。图 21 总结了实验结果，图 22 显示了对于 2 和 5 个类别的增量步骤。

  可以观察到，端到端方法在 2、5、10 和 20 个类别上获得了最佳结果。对于 50 个类别，尽管其与 Hybrid1（使用 CNN 分类器的 iCaRL 变体）得分相同，但比 iCaRL 低 1%。这是由于内存大小的限制，导致训练集严重不平衡，使得新样本的数据量是旧类别的 12.5 倍。为了突出端到端方法相对于 iCaRL 的统计显著性，作者对 CIFAR-100 的结果进行了配对 t 检验。相应的 p 值分别为 0.00005、0.0005、0.003、0.0077 和 0.9886，分别对应 2、5、10、20 和 50 个类别，这表明除 50 个类别外（两种方法表现相似），端到端方法在所有情况下相对于 iCaRL 的改进在统计上显著（$p < 0.01$）。

  还可以观察到，这一方法在增量步长（从每步 2 类到 20 类）上的性能保持稳定，而其他方法的性能依赖于每步添加的类别数量。这是因为在增量学习过程的早期阶段，由于只需分类少量类别，每次增量步骤中的类别数量较少有利于准确性。然而，随着更多步骤的应用来训练所有类别，最终阶段的准确性会下降。

当每次增量步骤添加较多类别时，情况则相反。早期阶段的准确性较低，但在最终阶段得到了更好的补偿。这些效果可以在图 22 中看到，其中可视化了每次增量步骤中不同数量的类别（2 和 5）。图 22 还显示，当每次增量步骤中使用较少数量的类别时，端到端方法显著优于 iCaRL。当每次步骤中类别数量较大时，iCaRL 的表现接近作者的这一方法，但整体仍较低。端到端方法在所有情况下都明显优于 LwF.MC，突出了端到端模型中代表性记忆的重要性。

# classes	2	5	10	20	50
Our-CNN	**63.8 ± 1.9**	**63.4 ± 1.6**	**63.6 ± 1.3**	**63.7 ± 1.1**	60.8 ± 0.3
iCaRL	54.1 ± 2.5	57.8 ± 2.6	60.5 ± 1.6	62.0 ± 1.2	**61.8 ± 0.4**
Hybrid1	34.9 ± 4.5	48.4 ± 2.6	55.8 ± 1.8	60.4 ± 1.0	60.8 ± 0.7
LwF.MC	9.6 ± 1.5	29.5 ± 2.2	40.4 ± 2.0	47.6 ± 1.5	52.9 ± 0.6

(a) CIFAR-100

图 21: 固定内存大小：在 CIFAR-100 上的准确率

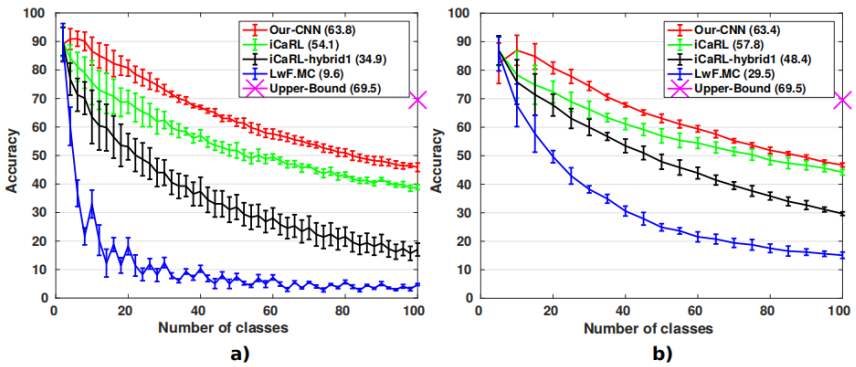


图 22: CIFAR-100 上的准确率

- **固定样本数量**

在这个实验中，作者使用每个旧类固定数量的训练样本来训练模型。这种限制不适用于新类别的样本。因此，允许内存随类别数量成比例增长，与内存大小保持恒定的固定内存实验相反。此外，为了衡量样本数量对准确性的影响，作者也评估了每个类别不同的样本数量：50、75 和 100。这里专注于类别增量步长为 5、10 和 20 的实验。我们考虑 iCaRL 和我们的方法的相同类别顺序，以确保结果具有可比性。在这个实验中，作者将比较重点放在了 iCaRL 和 Hybrid1 上，因为上一个实验已经得出 LwF.MC 的性能低于这两种方法。

图 23 总结了这些实验的结果。表格第一行表示每个增量步骤的类别数量。第二行包含训练期间每个旧类别使用的样本数量。其余行显示了评估的

不同方法的结果。比较 Our-CNN 和 [5] 中开发的方法的结果，我们看到在所有场景中，端到端的方法表现更好。与上一个实验一样，端到端方法在类别增量步长从 5 增长到 20 的情况下实现了类似的平均准确率，例如，每个类别 50 个样本时为 62.4、62.7、63.3，显示了其稳定性。为了衡量每类样本数量对训练的影响，本文还比较了图 21 和图 23 中的结果。在所有情况下，训练使用的样本越多，获得的准确率越高。对于 50 个样本，由于在早期增量步骤中可用的样本数量较少，结果比图 21 中的要差，这些初始模型训练不足。这导致了连锁效应，即使在最后阶段有更多样本可用，获得的模型也比预期的差。

# classes	5			10			20		
# img / cls	50	75	100	50	75	100	50	75	100
Our-CNN	**62.4**	**66.9**	**68.6**	**62.7**	**65.7**	**68.5**	**63.3**	**65.4**	**67.3**
iCaRL	56.5	59.9	62.2	60.0	62.3	63.7	61.9	63.0	64.0
Hybrid1	45.7	49.2	50.9	55.3	56.5	57.4	60.4	61.5	62.2

图 23: 固定样本数量：在 CIFAR-100 上的准确率

- **消融研究**

最后，作者分析该方法的各个组成部分，并展示它们对最终性能的影响。(所有这些消融研究均在固定的内存设置下进行)

作者首先通过一个使用 10 类增量步骤和三种样本选择方法的实验来评估样本选择策略：herding、随机选择和直方图选择。Herding 是文中中介绍的选择方法。随机选择指的是随机选择要存储在内存中的样本。在直方图选择策略中，样本是根据它们与其类别均值的距离来选择的。需要首先计算距离的直方图，分为十个箱，并将每个样本分配到其中一个箱中。然后，根据每个箱所包含的样本比例从每个箱中选择样本。从结果来看（herding：63.6%，随机选择：63.1%，直方图选择：59.1%），herding 和随机选择策略显示了最佳性能。

接下来的消融研究中，作者分析了数据增强和微调的影响。作者首先在没有平衡微调的情况下使用数据增强训练端到端模型（‘Our-CNN-DA’）。在第二个实验中，作者在没有数据增强的情况下使用平衡微调训练（‘Our-CNN-BF’）。最后，作者在没有数据增强和平衡微调的情况下训练端到端的模型（‘Our-CNN-Base’）。在这里，作者重点关注 5、10 和 20 类的增量步长实验。如前面的实验一样，iCaRL 和端到端方法的第一个分割使用相同的类别顺序，以确保结果具有可比性。图 24 和图 25 总结了该研究的结果。基线‘Our-CNN-Base’在所有情况下表现最差。然而，当添加数据增强

（‘Our-CNN-DA’）时，结果在所有情况下均有所改善，并在类别数为 5 时获得最佳结果（59.2）。然而，由于旧类和新类之间样本数量的不平衡，对于较大的增量步长，有必要添加平衡微调（‘Our-CNN-BF’）。当添加平衡微调（‘Our-CNN-BF’）时，结果在所有情况下均有所改善，特别是在较大的增量步长情况下，这突显了平衡训练集的重要性。最后，当将这两个组件都添加到基线时，得到完整模型（‘Our-CNN-Full’），作者观察到最佳结果，并在该数据集上为增量学习建立了新的最先进水平。

# classes	5	10	20
Our-CNN-Base	57.0	53.7	50.1
Our-CNN-DA	59.2	57.9	57.2
Our-CNN-BF	57.9	58.1	57.1
Our-CNN-Full	**63.8**	**64.0**	**63.2**
iCaRL	58.8	60.9	61.2
Hybrid1	48.7	55.1	59.8

图 24: 消融研究

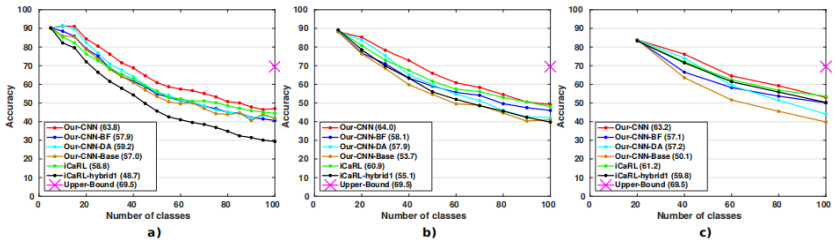


图 25: 在 CIFAR-100 上的消融研究

3 思考与理解：

3.1 联系与区别

3.1.1 联系

三篇论文虽然在实现方法上各有千秋，但殊途同归，他们仍具有不可分割的联系：

- **目标一致**：三篇论文的核心目标都是解决深度学习模型在增量学习中面临的灾难性遗忘问题。iCaRL 通过重放机制存储并重用旧任务样本，LwM 通过引入注意力蒸馏损失保持对旧任务的记忆，而 MAEs 通过自监督学习重建部分遮盖的图像，避免模型在学习新任务时遗忘旧知识。这些方法都旨在增强模型的稳定性，使其在学习新任务时不丧失已经学到的知识。
- **增强模型的适应性**：三篇论文都强调提高模型在动态环境中的适应能力。iCaRL 通过选择性重放旧任务样本，提高模型在新任务中的表现；LwM 通过注意力蒸馏损失，使模型在处理新旧任务时保持一致性；MAEs 通过掩码机制和图像重建过程，提高模型的泛化能力。这些方法共同致力于让模型在面对不断变化的数据分布时，既能学习新知识，又能保持对旧知识的记忆。
- **实验验证有效性**：三篇论文都通过大量的实验验证了各自方法的有效性。iCaRL 在多个图像分类数据集（如 CIFAR-100 和 ImageNet）上展示了其在类别增量学习中的优越性能，LwM 通过多个任务的实验结果证明了其在不存储旧数据的情况下，能有效减少遗忘，而 MAEs 则在图像数据集上展示了其通过掩码和自监督学习，在保持旧知识的同时，能够有效学习新知识。这些实验结果显示了各自方法在应对灾难性遗忘和提高模型适应性方面的成功。

3.1.2 区别

- **基础学习机制**：
  iCaRL：使用的是近似最近邻分类器，结合了分类和表示学习。模型的主要特点是通过维护一个固定大小的样本集（记忆库），来代表所有已经见过的类，以及一个增量训练策略，不断更新这个记忆库并重新训练分类器。
  LwM：不涉及显示的样本存储，而是通过在训练过程中对模型的注意力机制进行蒸馏，以保持对以前学习任务的记忆。该方法通过在损失函数中加入注意力蒸馏项，以保持模型在处理新旧任务时的注意力分布一致性。
  MAEs：利用自监督学习的掩码自动编码器策略，通过随机遮盖图像的部分

区域并要求模型重建这些区域，从而学习到鲁棒的特征表示。这种方法允许模型从大量未标记的数据中学习，同时减少对旧任务数据的依赖。

- **存储需求**：
  iCaRL：需要物理存储代表性样本，这意味着随着类别的增加，所需的存储空间也会增加，尽管有策略尝试最小化这一需求。
  LwM：不存储旧任务的实际样本，而是依靠模型内部的注意力机制来维持对旧知识的记忆，从而减少存储需求。
  MAEs：同样不需要存储旧任务的样本，通过训练模型以重建输入，间接学习和保持对先前任务的记忆。
- **实现策略**：
  iCaRL：主要通过重放旧样本来直接对抗灾难性遗忘，这是一种相对直接且易于理解的方法。
  LwM：使用了更为精细的方法，即通过蒸馏损失保持新旧任务间的知识连贯性，这种方法侧重于通过保持特征级别的连续性来对抗遗忘。
  MAEs：通过自监督学习和生成任务（即图像重建）间接对抗遗忘，这种方法更侧重于通过提升模型的内在泛化能力来防止遗忘。

3.2 尚未解决的问题

- LwM："Learning without Memorizing" 方法在解决灾难性遗忘问题上展示了显著的优势，但在一些方面仍存在尚未解决的问题。LwM 在处理更复杂和多样化的数据集方面还未得到充分验证。现有的实验主要集中在标准数据集如 iCIFAR-100 和 iILSVRC-small 上，而在高维度、动态变化和多模态特征的数据集中，LwM 的表现尚不明确。随着实际应用中数据复杂度的增加，验证 LwM 在这些复杂场景中的有效性显得尤为重要。总体而言，尽管 LwM 在增量学习领域取得了重要进展，但在应对复杂数据集、高效计算和长期稳定性方面，仍有许多挑战需要克服。
- iCaRL："Incremental Classifier and Representation Learning" 介绍了 iCaRL，一种同时学习分类器和特征表示的增量学习策略。在 CIFAR-100 和 ImageNet ILSVRC 2012 数据上的实验表明，iCaRL 能够在较长时间内增量学习，而其他方法很快失败。尽管结果令人鼓舞，但类增量分类问题仍然远未解决。特别是，iCaRL 的性能仍然低于在批处理设置下训练的系统所实现的性能，即在同一时间使用所有类的所有训练示例。iCaRL 在增量学习任务中的性能与在批处理设置下使用所有训练示例的性能之间存在差距。作者并未详细说明这种性能差距的具体原因，但这可能涉及到模型在处理增量学习时的能力限制，以及在增量学习场景下遇到的特定挑战。在未来的工作中，作者计划更详细地分析这一现象的原因，从而缩小剩余性能差距。
- MAEs：

在“Masked Autoencoders are Efficient Class Incremental Learners”这篇论文中，虽然取得了一些进展，但仍有许多未解决的问题和挑战需要进一步研究。

1. **任务间的知识转移**：该方法可能在处理任务间知识转移方面存在局限。不同任务之间的知识如何有效地转移和共享仍然是一个开放问题。
2. **掩码策略的优化**：掩码自动编码器的掩码策略对性能有很大影响。如何设计更加智能和自适应的掩码策略，以更好地捕捉数据的本质特征，仍然需要进一步研究。
3. **长尾分布数据**：类别增量学习中的长尾分布问题，即某些类别的数据量远少于其他类别，仍然是一个挑战。如何在这种不平衡的数据分布中保持高效的学习性能是一个重要的研究方向。

尽管掩码自动编码器在类别增量学习中展现了极大的潜力，但仍有许多方面需要进一步探索和优化。

3.3 未来发展方向

在连续学习领域，大部分的研究都集中在灾难性遗忘问题，但连续学习的最终目标应该是在无限的学习中不断增强模型的表现能力，使模型能够更好地解决各种问题 [6]。这意味着研究应更注重模型的整体性能，而不仅仅是灾难性遗忘。未来的研究应进一步探索如何在不同任务和环境中提升模型的泛化能力和适应性。

3.3.1 内存与计算资源的优化

许多关于灾难性遗忘的研究更关注内存问题，例如内存容量和信息隐私安全等，往往忽略了计算消耗 [7]。重放方法中的存储旧任务缓冲区和正则化方法中的旧参数保存，都涉及到内存的使用。在实际研究中，已经证明无需保存过多旧经验即可实现可塑性与稳定性的平衡 [8]。未来研究应进一步优化内存与计算资源的使用，特别是对于复杂数据带来的高能耗运算问题，如 GPT 等大模型，应当受到广泛关注。

3.3.2 多样化的应用与交叉学科融合

随着连续学习的运用越来越多元化，研究已经扩展到更多计算机视觉应用，如强化学习、自然语言处理和伦理考虑等领域。在交叉学科如机器人、图形学习和生物成像等场景中，连续学习也展现出了无限潜力 [9]。未来研究应更加注重跨学科的融合发展，结合神经科学等领域的研究成果，探索多模态连续学习模型，如

MSCGL 等，进一步提升模型的智能水平和应用广度。

3.3.3 类脑计算与神经科学的启发

神经科学在连续学习的发展中起着重要的引导作用。神经突触的可塑性在调节多个脑区域的稳定性和可塑性平衡机制，以及大脑的互补学习系统，为连续学习的研究带来了重要的生物启发 [10]。未来研究应更加关注类脑计算模型的发展，探索如何通过模拟人脑处理多模态信息的机制，提升连续学习模型的智能性和适应性。

同舟共济扬帆起，乘风破浪万里航。在研究者们的共同努力下，连续学习的前景一定是光明的。

参考文献:

[1] Prithviraj Dhar. Learning without memorizing. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019.

[2] Sylvestre-Alvise Rebuffi. icarl: Incremental classifier and representation learning. Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017.

[3] Jiang-Tian Zhai. Masked autoencoders are efficient class incremental learners. Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023.

[4] Francisco M. Castro. End-to-end incremental learning. European Conference on Computer Vision (ECCV), 2020.

[5] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning. 2017.

[6] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. IEEE transactions on pattern analysis and machine intelligence, 44(7):3366–3385, 2021.

[7] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. Neural networks, 113:54–71, 2019.

[8] Marc Masana, Xialei Liu, Bartosz Twardowski, Michele Menta, Andrew D Bagdanov, and Joost Van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(5):5513–5533, 2022.

[9] Erik Verwimp, Shai Ben-David, Matthias Bethge, Xudong Chen, Zhiyi Chi, Valentin Delmas, Amr Fahmy, Shai Friedman, Max Gabel, Amit Gautam, et al. Continual learning: Applications and the road forward. arxiv preprint arxiv:2311.11908, 2023.

[10] Lin Wang, Xiaopeng Zhang, Hang Su, and Wenjun Li. A comprehensive survey of continual learning: Theory, method and application. arxiv preprint arxiv:2302.00487, 2023.